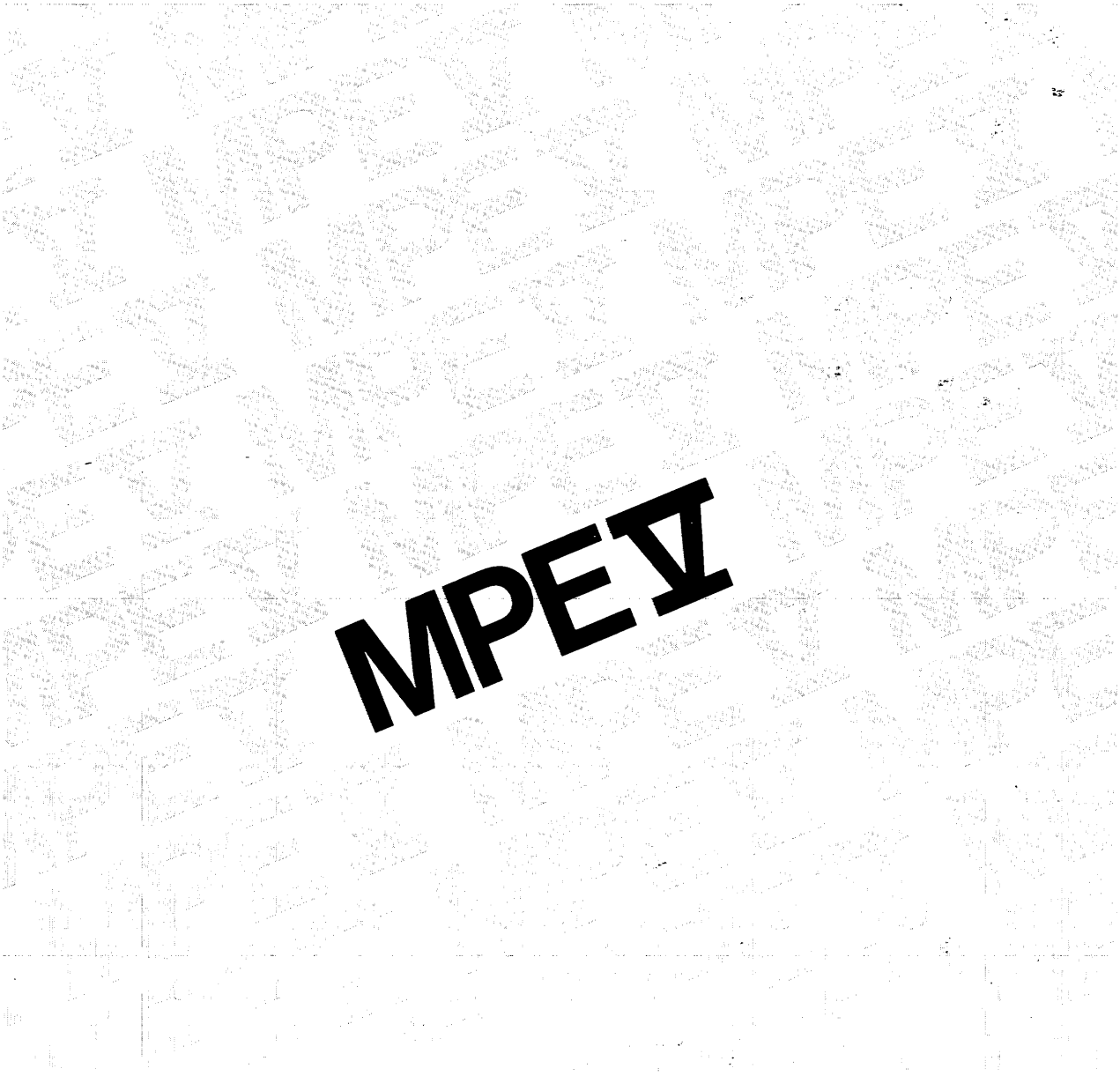
HP 3000 Computer Systems

HEWLETT PACKARD

MPE V COMMANDS
Reference



HP 3000 Computer Systems

MPE V COMMANDS

Reference Manual

HEWLETT PACKARD
19447 PRUNERIDGE AVENUE, CUPERTINO, CA 95014

Part No. 32033-90006
E0286

Printed in U.S.A.

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the most recent version of each page in the manual. To verify that your manual contains the most current information, check the dates printed at the bottom of each page with those listed below. The date on the bottom of each page reflects the edition or subsequent update in which that page was printed.

Effective Pages	Date
All	ORIGINAL

PRINTING HISTORY

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The dates on the title page change only when a new edition or a new update is published. No information is incorporated into a reprinting unless it appears as a prior update; the edition does not change when an update is incorporated.

The software code printed alongside the date indicates the version level of the software product at the time the manual or update was issued. Many product updates and fixes do not require manual changes and, conversely, manual corrections may be done without accompanying product changes. Therefore, do not expect a one to one correspondence between product updates and manual updates.

First Edition . . . . . Feb 1985. . . . . G.02.00

MPE V MANUAL PLAN

INTRODUCTORY LEVEL:

GENERAL INFORMATION Manual 5953-7553

GUIDE FOR THE NEW USER 32033-90009 IN PROGRESS

GUIDE FOR THE NEW OPERATOR 32033-90021 IN PROGRESS

STANDARD USER LEVEL:

MPE V/R COMMANDS Reference Manual 32002-90017

MPE V/R INTRINSICS Reference Manual 32002-90018

MPE V/R UTILITIES Reference Manual 32002-90019

SEGMENTER Reference Manual 30000-90011

DEBUG/STACK DUMP Reference Manual 30000-90012

FILE SYSTEM Reference Manual 30000-90236

ADMINISTRATIVE LEVEL:

MPE V/R SYSTEM OPERATION & RESOURCE MANAGEMENT Reference Manual 32002-90016

SUMMARY LEVEL:

MPE SOFTWARE POCKET GUIDE 30000-90049 IN PROGRESS

There are many more manuals applicable to the HP 3000. A complete list may be found in every issue of the MPE V Communicator. Please contact your System Manager.

CONVENTIONS USED IN THIS MANUAL

NOTATION	DESCRIPTION
COMMAND	Commands are shown in CAPITAL LETTERS. The names must contain no blanks and must be delimited by a nonalphabetic character (usually a blank).
KEYWORDS	Literal keywords, which you enter as you need them but exactly as specified, appear in CAPITAL LETTERS.
parameter	Required parameters, for which you must substitute a value, appear in *standard italics* and stand alone: they are not delimited by brackets or by braces.
[*parameter*]	Optional parameters, for which you may substitute a value, appear in *standard italics* and are delimited by brackets [ ].

An element inside brackets is optional. Several elements stacked inside a pair of brackets means the user may select any one or none of these elements.

Example:
```
[ A ]
[ B ]
```
user may select A or B or neither.

When brackets are nested, parameters in inner brackets can only be specified if parameters in outer brackets or comma place-holders are specified.

Example: [*parm1*[,*parm2*[,*parm3*]]] may be entered as:

parm1,*parm2*,*parm3* or
parm1,,*parm3* or
,,*parm3* , etc.

Shaded delimiters appear in syntax diagrams to emphasize the role of the delimiter. Shading signifies that the delimiter preceding a parameter must be used for one of two reasons.

(1) You have chosen to use the parameter, and the parameter requires the presence of the delimiter.

(2) You have chosen to use more than one of the parameters in a chain of parameters that are connected by commas; and

You have chosen to leave gaps in the chain of parameters by omitting one or more intermediate parameters. The gaps created by the omission of the intermediate parameters must be delimited (separated) to maintain the correct (ordered) position of the remaining parameters.

Example: *itema*[,*itemb* [,*itemc*]]

means that the following are allowed:

itema
itema, *itemb*
itema, *itemb*, *itemc*
itema, *itemc*

{ } When several elements are stacked within braces the user must select one of these elements.
Example:
```
{ A }
{ B }
```
user must select A or B.

... An ellipsis indicates that a previous bracketed element may be repeated, or that elements have been omitted.

user input In examples of interactive dialog, user input is underlined.
Example: NEW NAME? ALPHA1

superscript^{C} Control characters are indicated by a superscript^{C}. Example: Y^{C}. (Press Y and the CNTL key simultaneously.)

(BREAK) (BREAK) indicates a terminal key. The legend appears inside.

<<COMMENT>> Programmer's comments in listings appear within << >>.

** Comment ** Editor's comments appear in this form.

CONTENTS

CONTENTS (Continued)

CONTENTS (Continued)

CONTENTS (Continued)

CONTENTS (Continued)

CONTENTS (Continued)

TABLES

PREFACE

This new edition of the MPE V Commands manual puts all of the MPE commands within easy reach, in a single volume. Whether you are a System Operator, a System Manager, an Account Manager, or you are just getting used to your HP 3000, you will find all of the fundamental commands provided by MPE in this one reference.

The commands are arranged alphabetically. If a particular command requires special capabilities, (SM, OP, AM, etc), those capabilities are mentioned in the "USE" description for that command.

Two new programming language subsets join the list of those already available to users of the Hewlett Packard 3000 Series of Computers: HP Business BASIC and FORTRAN 77, which are available through separate purchase.

You will find references to data communications in the commands that "bridge" into the NS/3000 AdvanceNet subsystem or into the DS/3000 subsytem (:FILE and :RESET). AdvanceNet and DS/3000 are not part of the MPE fundamental operating system and must be purchased separately. The data communications parameters for :FILE and :RESET are presented in order to provide a full and accurate representation of the command syntax. Omitting those parameters will not hinder your use of those commands. Attempting to use those parameters if your system does not support data communications will produce an error.

This release, G.02.00 ("U-MIT"), supports the Series 37 Office Computer and the Series 39, 40, 40SX, 42, 42SX, 44, the new Series 58, and the Series 68 Computers. The Series 58 replaces Series 48 at the top of the Hewlett Packard mid-range line.

New file codes and file mnemonics appear in the :BUILD and :FILE commands. :RESTORE has the new CREATOR default. This release of MPE ensures that the :RESTORE command is backward compatible with the G.00.00 release. If the creator of a file cannot be found on the system, the file will not be restored unless you specify the CREATE option or the CREATOR option.

With the addition of the :CHANGELOG command, you are no longer required to use the :LOG command to change logging files. Instead you may instruct MPE to change logging files on demand or, by using new parameters for :ALTLOG and :GETLOG, you may initiate a sequence of automatic log file changes. The :LISTLOG, and :SHOWLOGSTATUS commands are modified to give you up-to-date information on the process of logging.

Like its predecessors, this edition, is divided into three sections: an introduction, a reference section documenting each user command, and a discussion of user defined commands (UDCs). The "ADDITIONAL DISCUSSION" section of each command may also list additional references should you need further help.

The following documentation may also be helpful:

- MPE V Intrinsics Reference Manual (32033-90007)
- MPE File System Reference Manual (30000-90236)
- MPE V Utilities Reference Manual (32033-90008)
- MPE V System Operator and Resource Management Reference Manual (32033-90005)

HP Advance Net NS/3000 User/Programmer Reference Manual (32344-90001)

"Communicator 3000", Vol. 2, Issues 5 and 6

INTRODUCTION TO COMMANDS

SECTION I

MPE commands allow you to start, control, and stop the processing of programs and to request various other system operations. You generally use them to perform functions external to the source-language programs that you write, although many such functions support your programs. For example, you will use commands to:

- Initiate an interactive session (:HELLO command) or batch job (:JOB command).
- Display status information about sessions and jobs in the system (:SHOWJOB).
- Create, save, and delete files (:BUILD, :SAVE, and :PURGE, respectively); specify and list their characteristics (:FILE and :LISTF); dump them to tape or serial disc and subsequently restore them to the system (:STORE and :RESTORE); and specify security provisions for them (:ALTSEC, :RELEASE and :SECURE).
- Handle programs; compile (:FORTRAN, :FTN77, :COBOL, :COBOLII, :PASCAL, :RPG, :BBASIC, :SPL), prepare those programs (:PREP), and execute them (:RUN).
- Determine the status of devices (:SHOWDEV and :DSTAT).
- Determine the status of devicefiles, which are disc files originating on or destined for nonsharable devices (:SHOWIN and :SHOWOUT).
- Communicate with other users (:TELL) and with the System Operator (:TELLOP).
- Define your own commands (:SETCATALOG).
- Access private disc volumes (:MOUNT and :DISMOUNT).
- Obtain assistance in using the Command Interpreter (:HELP).

Many other commands and functions are available as well. Complete specifications for the MPE commands appear in Section II.

HOW TO USE THIS MANUAL

The reference specifications for all MPE commands available to MPE users appear in Section II. For easy reference, they are presented alphabetically by command name.

For each command, the reference specifications present the following information:

- Command syntax.
- Parameter definitions, including meaning, constraints, and defaults.
- When and how the command may be used, such as; in jobs, sessions, during BREAK, or programmatically.

- Whether the command is interruptible with the [BREAK] key.
- Capability required: System Supervisor (OP), System Manager (SM), or Account Manager (AM), or whether the command must be executed at the Console.
- Operation. A description of how to use the command, and any related special considerations.
- An example.
- Additional discussion, if any, in this or another manual.

Many commands may be invoked under special circumstances during the execution of another operation, provided that the operation itself is "Breakable." For those commands, the "USE" description for the command will note that, "This command may be issued... in BREAK." You may suspend the execution of the current operation by pressing the [BREAK] key on the terminal or by calling the CAUSEBREAK intrinsic.

Similarly, the notation "This command may be issued... from (in) a Program" appearing in the "USE" description indicates that the command is available through the COMMAND intrinsic. The CAUSEBREAK and COMMAND intrinsics are explained in the MPE V Intrinsics Reference Manual (32033-90007).

The "USE" description also indicates whether special capabilities are required to execute the command and, if they are, which ones are required. The special capabilities you will encounter most often are:

OP — System Supervisor. A user whose account and username have been assigned OP capability. The System Supervisor is responsible for optimizing the performance of the system, and may execute commands like :TUNE and :ALLOCATE or logon with ;HIPRI. OP commands should not be confused with Operator commands, which are explained below.

SM — System Manager. A user whose account and username have been assigned SM capability. The System Manager creates the accounting structure and assigns to each account appropriate capabilities and initial security provisions. The System Manager also works with the Hewlett Packard Systems Engineering (SE) and the System Supervisor to develop the MPE configuration which best suits the needs of the users at their installation.

AM — Account Manager. A user, selected by the System Manager, whose username has been assigned AM capability. The Account Manager establishes groups and users within a particular account and assigns capabilities and attributes to them.

CV — Create Volumes capability. A capability assigned to accounts and usernames allows the user to create Private Volumes and access them.

PS — Programmatic Sessions. A user who is allowed to execute the :STARTSESS command and to call to the STARTSESS and ABORTSESS intrinsics. This user is designated by the System Manager during the creation of SYSTART.PUB.SYS for the Startup State Configurator; or by an applications programmer who creates a turnkey system.

NM Node Manager. A user, selected by the System Manager, who is assigned to manage the data communication subsystem at a specified location.

NA Network Administrator. A user, selected by the System Manager, who has been assigned the capability to perform comprehensive diagnostics on an entire communications network.

Executing the commands in this manual does not require NM or NA capability. They are included to present a complete listing of the user capability list. Refer to the HP AdvanceNet NS/3000 User/Programmer Reference Manual (32344-90001).

Many commands are more restricted in their use: they are noted in the "USE" description by the phrase, "...may be executed only from the Console...." These are Operator Commands. They do not require special capabilities, but they are usually executed only by the System Operator, who is logged on at the Console. With five notable exceptions, even Operator Commands may be made available to others users working at standard terminals. They become available only if the Operator specifically assigns (distributes) them to particular users. The five exceptions are =ABORTIO, =ABORTJOB, =LOGON, =LOGOFF, and =SHUTDOWN. These five remain under the exclusive jurisdiction of the Console.

Operator commands may be distributed to specific users, at the Operator's discretion, in two ways: with the :ALLOW command or with the :ASSOCIATE command. The :ALLOW command, as the name implies, gives users the ability to execute specific Operator commands at their own terminals. Any command normally executed at the System Console may be allowed, except those which can be entered only at the "=" prompt. These commands include: =LOGON, =LOGOFF, =ABORTIO, =ABORTJOB, =REPLY, RECALL, and =SHUTDOWN (These commands are console commands, since the System Console is the only device that will accept a A^C and display an "=" prompt in response.) :ALLOW should be used sparingly, because most Operator commands are powerful, and their use implies a high level of responsibility.

The :ASSOCIATE command, the second way to distribute Operator capabilities, is more limited than the :ALLOW command. When a user is associated with a specific device class, he or she is granted operator control of any device belonging to that class. Before a user can be :ASSOCIATEd, the username must be entered into a device/class user association table. This table is created by the System Manager with the ASOCTABL.PUB.SYS utility or the ASOCTABL5.PUB.SYS utility.

Another method used to distribute Operator responsibilities is to transfer Console capabilities to another logical device, or *ldev*. (This is called "moving" the Console. Because the Console is a logical designation rather than a physical device, "moving" it requires only that the Console be assigned, using the :CONSOLE command, to a different *ldev*.) Moving the Console requires OP capability since the Console, and its capabilities, are privileged resources. In this sense, moving the Console is not so much a distribution of responsibility as it is a reassignment of responsibility to another user. The System Console is typically configured as *ldev* 20 and, under normal circumstances, should not be moved to another logical device.

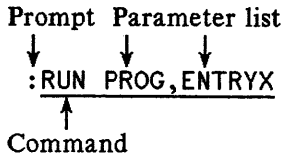
The reference specifications appearing in Section II cover the rules for entering each command. They show the command syntax and format; define the parameters, discuss constraints on and default values for them; provide an overview of the operation requested by the command; and present examples illustrating proper command entries. Parameter definitions common to more than one command are repeated for each applicable command.

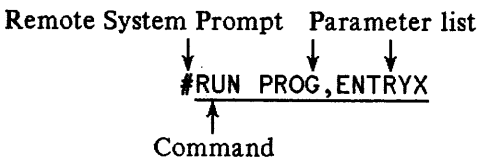
COMMAND ELEMENTS

Each MPE command consists of:

- A colon prompt (supplied by the system during interactive sessions; the user must provide the prompt, usually an exclamation point (!) or another substitute, for batch jobs).
- A command name (required in all cases).
- A parameter list (used in most cases).

A typical command including all three elements appears as follows:





Prompt

The prompt identifies a statement as an MPE command. In an interactive session, MPE displays the colon at the terminal whenever it is ready to accept a command. You respond by entering the remainder of the command after the colon. In a batch job, however, you may enter an exclamation point (!) or other characters to simulate the colon prompt, placing it in column one (1), and following it with the remainder of the command.

Command Name

The command name, which you enter following the colon, requests a specific operation. MPE prohibits embedded blanks within the name, and rejects the command if they appear. MPE interprets the next nonalphabetic character encountered as the end of the command name; typically this character is a blank. Blanks may appear between the colon and the command name except in the `:HELLO`, `:JOB`, and `:DATA` commands.

Parameters

The parameter list contains one or more options for the command. Parameters are required in some commands, but are optional or prohibited in others. Parameter lists can include positional parameters keyword parameter groups, or both, separated from each other by delimiters such as commas, semicolons, equal signs, or other punctuation marks. MPE permits both decimal and octal numbers as command parameters. Octal numbers are distinguished by their preceding percent sign (%).

If a parameter is required, a value for it must be included in the parameter list. MPE substitutes default values for optional parameters that are omitted from the list. These default values are specified in the parameter definitions that accompany each command reference.

Normally, you must separate the parameter list from the command name by one or more blanks. When you omit the first optional positional parameter in a parameter list (refer to "POSITIONAL PARAMETERS" later in this section), you can begin the list with a comma, or other delimiter that normally follows the first parameter, and omit the first blank after the command name. The comma serves as a delimiter in place of blanks. Any delimiter within the parameter list can be surrounded by any number of blanks.

Note that names of user defined commands (UDC's) may be composed of alphanumeric characters, but must begin with an alphabetic character. MPE commands contain only alphabetic characters. A string such as SHOWDEV6 is treated as a UDC, because the last character is numeric. If no UDC by that name exists, then because the last character is numeric, the command is passed to MPE with the numeric value (6 in this case) as a parameter. Normal MPE command syntax requires a space between the command and the parameter. In this case, the parameter is numeric, and SHOWDEV is a valid MPE command; in this case MPE is capable of separating the command from the parameter and will execute it accordingly. For more information, refer to Section III of this manual.

The end of a command is indicated by the last nonblank character of the record in which it appears. For terminal input, a carriage return usually signals termination of command input. However, if the last nonblank character of the record is a continuation character, the command input is continued to the next record. For more information, refer to "Continuation Characters" in this section.

POSITIONAL PARAMETERS. With positional parameters, the meaning of a parameter depends upon its position in the parameter list. For example, in the :FORTRAN command, the parameter list specifies the input file containing the source program, the output file to which the object code is written, and the output file to which the source listing is written, always in that order. In the following :FORTRAN command, the variable names INP, OUT, and *LST indicate the source, object, and listing, respectively:

```
:FORTRAN INP,OUT,*LST
```

(In the above example, the asterisk (*) in *LST is a special character denoting a backreference to a previously defined file. For more information, refer to "Backreferencing" in this section.

Positional parameters are separated (delimited) from one another by commas. When you omit an optional positional parameter from within a list, you must include the comma that would normally follow that parameter. Two adjacent commas indicate a missing optional positional parameter. When you omit an optional positional parameter immediately following a command name, a comma becomes the first character in the parameter list. When you omit positional parameters from the end of the list, you need not include commas to signify this; the terminating RETURN or end-of-record is sufficient. The following examples demonstrate the proper omission of parameters from a command:

```
:FORTRAN INP,,*LST          ** The parameter "OUT" is omitted.  **
:FORTRAN                    ** All parameters omitted.          **
```

KEYWORD PARAMETERS. You can enter keyword groups in any order with respect to each other. A keyword group consists of a keyword that denotes the group's meaning, sometimes followed by one or more subparameters. Each keyword group is preceded by a semicolon (;). When more than one subparameter appears in a group, they are usually separated from each other by commas. All delimiters can optionally be preceded or followed by blanks. The following example shows both positional and keyword parameters. INPT and OUTP are the positional parameters. DL and CAP are keywords. PH, DS, and MR are subparameters of the CAP keyword:

```
:PREP INPT,OUTP;DL=500;CAP=PH,DS,MR
```

When both keyword groups and positional parameters form a list, positional parameters that are not associated with a keyword always precede the keyword groups. When you omit trailing parameters from the positional group, you need not include their delimiters. The occurrence of the first keyword signifies the termination of the positional group. When you omit optional subparameters from a keyword group, follow the same rules that apply to positional parameters.

WILDCARD CHARACTERS. For some commands, you may substitute "wildcard characters" for certain parameters in the list. The wildcard characters count toward the eight character limit for user, group, account, and filenames. These wildcard characters have the following meanings:

@	Specifies one or more alphanumeric characters. When used by itself, @ denotes all members of the set.
#	Specifies one numeric character.
?	Specifies one alphanumeric character.

These characters can be used as follows:

n@	Represents all items starting with the character "n".
@n	Represents all items ending with the character "n".
n@x	Represents all items starting with the character "n" and ending with the character "x".
n#######	Represents all items starting with the character "n" followed by seven digits.
?n@	Represents all items whose second character is "n".
n?	Represents all two-character items starting with the character "n".
?n	Represents all two-character items ending with the character "n".

Syntax Conventions

The syntax boxes which appear in the specifications for each command in Section II are included to illustrate parameter choices. Brackets ([ ]), which enclose separate groups of parameters, indicate the choice of that group is optional.

Several elements may be stacked inside a pair of brackets. The user may select one or none of these elements. An ellipsis with three dots (. . .) signifies that the elements may be repeated. When brackets are nested, parameters in the inner brackets can only be specified if the parameters in the outer brackets or place holders are specified. Either a comma (,) or semicolon (;) may be a place holder, but they are not interchangeable and must be specified in the syntax. For example:

```
:BASICPREP [commandfile][,[progfile][,listfile]]
```

A sample of user choices includes:

```
:BASICPREP
:BASICPREP commandfile
:BASICPREP,progfile
:BASICPREP,,listfile
:BASICPREP,progfile,listfile
```

Braces ({ }) mean the elements contained are required. The user must select one of these elements. For example:

```
:SAVE {$OLDPASS,newfilereference}
      {tempfilereference        }
```

In the example above, the user must enter either ":SAVE $OLDPASS,newfilereference" or ":SAVE tempfilereference".

A combination of brackets and braces indicates mixed optional and required parameters. If the user desires the optional parameter, the user is required to chose only one of the elements in the braces. For example:

```
                        {0 }
                        {1 }
:LISTF [fileset][       {2 }]
                        {-1}
```

In this example, the user must specify 0, 1, 2 or -1, if this optional parameter is chosen.

ENTERING COMMANDS

You can enter commands through any standard input device, usually a terminal (for sessions) or a streamfile (for jobs). Each command is accepted by the MPE Command Interpreter (CI), which passes it to the appropriate system procedure for execution. Following execution, control returns to the Command Interpreter, which is now ready for another command.

NOTE

After completion of the execution of a command, the Command Interpreter prompts the input device for the next command. If this READ fails due to an I/O error, the CI tries to post another READ. If four consecutive READ attempts fail, the CI will log the user/job off the system.

All jobs and sessions have two system defined files, $STDIN and $STDLIST, associated with them. $STDIN is a filename indicating the standard input file from which the job or session is initiated. Commands and data are read from $STDIN by the Command Interpreter. $STDLIST is a filename indicating the standard listing or output file corresponding to the particular job or session. Command output, error messages, etc. are written to $STDLIST. For sessions, both $STDIN and $STDLIST are typically terminal devicefiles. For jobs, $STDIN is often a streamfile or a card reader; $STDLIST is often a spoolfile destined for a line printer. For more information, refer to the :STREAM command in Section II.

$STDLIST is said to be "duplicative" if the data read in from $STDIN is automatically echoed to $STDLIST. An example of this is a session on a CRT terminal. For more information on $STDIN and $STDLIST refer to "System Defined Files" in this section.

In cases where $STDLIST is not duplicative, such as jobs where $STDLIST is a line printer, MPE echoes $STDIN to $STDLIST. For security reasons, MPE will strip all passwords and lockwords from $STDIN before echoing to $STDLIST. Any text following a "/" which could be interpreted as a lockword or password (e.g. eight or fewer alphanumeric characters, beginning with an alphabetic and delimited by a period, comma, semi-colon, or a blank) will not appear on $STDLIST, although MPE uses the full, original line for its command execution.

Continuation Characters

When the length of a command exceeds one record (for instance, one entry line on your terminal, or one source card), you may enter an ampersand (&) as the last nonblank character of the record and continue the command on the next record. The next record must begin with a colon, which is supplied automatically by MPE in interactive processing. In batch processing you must enter a colon or a special character to simulate the colon. You may embed blanks between the colon that begins the continuation record and the first nonblank character in the continuation record. In the following example, the command contains a continuation character at the end of the first line and an embedded blank at the beginning of the second:

```
:RUN PROGB;NOPRIV;LMAP;STACK=501;PARM=5;&
: DL=600;LIB=G
```

Commands may contain a maximum of 278 characters exclusive of prompting colons and continuation ampersands. MPE does not permit a command name, keyword, positional parameter, or keyword subparameter to span more than one line.

MPE does not begin interpretation of a command until the last record of the command is read.

Interactive Session

In interactive processing, users at terminals converse directly with the computer on a real time basis. This type of interaction is called a session. During a session, users are prompted for input.

Executing Command Logon

Another way to issue an MPE command is through the COMMAND LOGON capability. It is used to simplify the logon and logoff procedures when just one MPE command is to be executed. A COMMAND LOGON is performed by entering any MPE command and its parameter list, enclosed in parentheses, followed by the parameter list normally associated with the :HELLO command (user, account and group names). The following example shows how someone with the username MYNAME, and account name MYACCT, can logon to the system, execute a program called MYPROG, and log off, all in a single MPE command:

```
:(RUN MYPROG) MYNAME.MYACCT
```

For more information, refer to ": ( ) COMMAND LOGON" in Section II of this manual.

Batch Jobs

A batch job is a file containing one or more commands, and, optionally, references to any data files needed for the job to successfully execute. The job file is processed independently of your session. It can be submitted from either a session, or another job. This technique, which uses the :STREAM command, is known as "streaming".

Several jobs can be streamed at one time. They are selected for processing according to the input priority assigned when the job was streamed, and the job limit, or if it is a scheduled job, the time that you requested the job be introduced to the system. For more information, refer to :STREAM in Section II, and :STREAMS in the MPE V System Operation and Resource Management Reference Manual (32033-90005).

Sequence Numbers in Job Files

Spooled jobfiles may have sequence line numbers. MPE determines whether a jobfile is sequenced or not by scanning the *JOB record. If the last eight columns of the :JOB record are numeric, the jobfile is considered to be sequenced. MPE procedes to ignore the last eight columns of each record of every file associated with that particular job.

There are certain implications associated with using a sequenced jobfile. Ampersands (&) used as continuation characters must be placed before the sequence fields or they will not be read. Any programs that utilize sequence numbers for any purpose (i.e. continuity check when doing a serial data read) will not function properly as the sequence numbers will not be read. A file used as input to an editor must have no data in the sequence fields as these last eight bytes are not read.

MPE commands in spooled jobs may have sequence line numbers. MPE determines whether a job is sequenced when it scans the :JOB record. If the last eight columns of the :JOB record are numeric, the job is considered sequenced. In sequenced jobs, only the :JOB record is required to have numeric data in the last eight columns. MPE does not check any subsequent records for numeric data, or even proper sequencing in the last eight columns; these columns are simply ignored.

This definition of sequence numbers has certain implications. Ampersands (&) indicate continuationlines, and must be placed before, not after, sequence fields. Secondly, :JOB files may be numbered or unnumbered. If a file is numbered, MPE will strip the last eight bytes of each non-CI command record. The sequence of data records containing non-MPE commands will not be checked, and a program reading these records (data records) from $STDIN will never see the sequence numbers.

Executing Commands Programmatically

MPE allows you to execute nonprogram commands from within many subsystems. Refer to Table 1-1 for a list of nonprogram commands. You do this from the subsystem prompt by preceding the MPE command with a colon (:). For example:

```
>:REPORT
ACCOUNT          FILESPACE-SECTORS        CPU-SECONDS      CONNECT-MINUTES
   /GROUP          COUNT     LIMIT      COUNT     LIMIT      COUNT     LIMIT
TECHPUBS           39868     50000     247000        **     124250        **
   /SALES          11717        **      92182        **      29267        **
>
```

In this case the MPE command :REPORT was accessed from within the subsystem.

Many MPE commands can be executed within programs using the COMMAND intrinsic. This intrinsic invokes the Command Interpreter and passes the MPE command for execution. Complete information on the COMMAND intrinsic appears in the MPE V Intrinsics Reference Manual (32033-90007).

COMMAND ERRORS

If you make an error while entering a command in an interactive session, MPE suppresses execution of the command and attempts to determine the cause of the error. MPE prints a caret (^) under the incorrect part of the command, along with the appropriate message, and control is returned to your terminal.

If you enter an erroneous command in a job file, and the preceding is not a :CONTINUE command, MPE will abort, or flush, the job. MPE prints an error message on the job's standard list device (usually the printer) so that you know which command caused the job to abort. However, if the erroneous command is preceded with the :CONTINUE command, MPE will report the error, and continue processing the job. For further information, refer to the :JOB, :EOJ, and :CONTINUE commands in Section 2.

If a command is continued over several lines, and an error is detected, the offending line will be echoed, preceded by the line number on which the error has occurred. Note that when you use continuation characters, MPE reads all continued portions of the command entry as one line. In the following example, the filename includes a percent sign (%). File names must be alphanumeric, so a Command Interpreter error is generated:

```
:FILE ABC&
:=TAP%&
:;NEW
(1)=TAP%
      ^
UNEXPECTED CHARACTER IN FILE NAME; EXPECTED "." OR "/".  IS
THE DELIMITER BETWEEN PARAMETERS CORRECT?  (CIERR 582)
```

The MPE :REDO command allows you to edit the last line that you entered. To use it, enter: REDO. The previous command will be echoed to your terminal. Using edit symbols, you can insert, delete, or replace a character, or undo a previous edit. If you do not use an edit symbol, MPE assumes that you want to replace one or more characters. For more information, refer to the :REDO command in Section II.

MPE also provides an online HELP facility. To access the HELP subsystem, enter :HELP, or :HELP followed by the name of the command you want information about. The :HELP command has several optional parameters such as parms, operation, and example which can be very useful for obtaining only the desired information. For more information, refer to the :HELP command in Section II of this manual.

INTERRUPTING COMMAND EXECUTION

It is sometimes necessary to interrupt an executing MPE command to execute another command. Command interruption is accomplished by pressing [BREAK] on your terminal (sometimes labeled "INTERRUPT" or "ATTENTION"). In some cases, pressing [BREAK] will cause the executing command to be suspended or aborted. In other cases, pressing [BREAK] will have no effect.

The exact effect that pressing (BREAK) will have on your executing command depends on whether the command is one that executes a subsystem or user program (program command), or whether the command does not execute such a program (nonprogram command). Table 1-1 indicates BREAKable and nonBREAKable nonprogram commands. Table 1-2 indicates BREAKable nonprogram commands.

Table 1-1. Nonprogram Commands

BREAKABLE	NONBREAKABLE		
:BYE	:() COMMAND LOGON	:EOJ	:RENAME
:HELLO (If there is no OPTION LOGON UDC*)	:ABORT	:FILE	:REPLY/=REPLY
:HELP	:ABORTIO/=ABORTIO	:FOREIGN	:RESET
:LISTACCT	:ABORTJOB/=ABORTJOB	:FREERIN	:RESETACCT
:LISTF	:ACCEPT	:GETLOG	:RESETDUMP
:LISTGROUP	:ALLOCATE	:GETRIN	:RESETDUMP
:LISTLOG	:ALLOW	:GIVE	:RESUME
:LISTUSER	:ALTACCT	:HEADOFF	:RESUMEJOB
:LISTVS	:ALTGROUP	:HEADON	:RESUMELOG
:REDO	:ALTJOB	:IF	:RESUMESPOOL
:REPORT	:ALTLOG	:JOB	:SAVE
:RESTORE	:ALTSEC	:JOBFENCE	:SECURE
:RUN	:ALTSPOOLFILE	:JOBPRI	:SETCATALOG
:SEGMENTER	:ALTUSER	:JOBSECURITY	:SETDUMP
:SHOWALLOW	:ALTVEST	:LDISMOUNT	:SETJCW
:SHOWCATALOG	:ASSOCIATE	:LIMIT	:SETMSG
:SHOWDEV	:BUILD	:LMOUNT	:SHOWCOM
:SHOWIN	:CHANGELOG	:LOG	:SHOWLOG
:SHOWJCW	:COMMENT	=LOGOFF	:SHOWLOGSTATUS
:SHOWJOB	:CONSOLE	=LOGON	:SHOWTIME
:SHOWME	:CONTINUE	:MOUNT	=SHUTDOWN
:SHOWOUT	:DATA	:NEWACCT	:SPEED
:SHOWQ	:DEALLOCATE	:NEWGROUP	:STARTSPOOL
:STREAM	:DEBUG	:NEWUSER	:STOPSPOOL
:SYSDUMP	:DELETESPOOLFILE	:NEWVSET	:STREAMS
:TUNE	:DISALLOW	:OUTFENCE	:SUSPENDSPOOL
:VINIT	:SET	:PTAPE	:SWITCHLOG
	:DEBUG	:PURGE	:TAKE
	:DISASSOCIATE	:PURGEACCT	:TELL
	:DISMOUNT	:PURGEGROUP	:TELLOP
	:DOWN	:PURGEUSER	:UP
	:DOWNLOAD	:PURGEVSET	:VMOUNT
	:DSTAT	:RECALL/=RECALL	:VSUSER
	:ELSE	:REFUSE	:WARN
	:ENDIF	:RELEASE	:WELCOME
	:EOD	:RELLOG	

Table 1-2. Program Commands (All Are Breakable)

:BASIC	:COBOLII	:PASCALGO
:BASICGO	:COBOLIIGO	:PASCALPREP
:BASICOMP	:COBOLIIPREP	:PREP
:BASICPREP	:COBOLPREP	:PREPRUN
:BBASIC	:EDITOR	:RPG
:BBASICGO	:FCOPY	:RPGGO
:BBASICOMP	:FORTGO	:RPGPREP
:BBASICPREP	:FORTPREP	:SPL
:COBOL	:FORTRAN	:SPLGO
:COBOLGO	:PASCAL	:SPLPREP
		:STORE

Interrupting Nonprogram Commands

Pressing [BREAK] while a BREAKable nonprogram command is executing will abort the command, and the Command Interpreter will issue a colon prompt.

If you BREAK from a command while waiting for operator intervention, (while a tape request is pending) only three commands will function after the colon prompt is displayed (:RECALL, :RESUME, and :REPLY). If you use any other command a warning message is issued and the command is ignored.

If you log on using the :HELLO command and press [BREAK] during output of the logon message, MPE terminates the message, but you will remain logged on and MPE will prompt you for your next command provided you do not have an OPTION LOGON UDC. If you have an OPTION LOGON UDC, it will display the colon (:) prompt.

If you log on using the :( ) COMMAND LOGON construct and press [BREAK] during output of the logon message, MPE terminates the output and begins executing the command enclosed within the parentheses. If you press [BREAK] after the logon message is output, MPE BREAKs the currently executing command. If this command is a nonprogram command, MPE stops its execution and logs you off immediately.

Interrupting Program Commands

Program commands invoke MPE subsystems or run user programs. All program commands are BREAKable. Note that while you may be able to BREAK the commands themselves, the programs themselves may disable the BREAK facility. Also, if a program is invoked with a nonBREAKable UDC, BREAK will be be disabled for the execution of that program.

When you press [BREAK] to interrupt a program command, the execution of that command is suspended and the Command Interpreter issues a prompt for a new MPE command. If you then enter a nonprogram command (other than :HELLO, :BYE, :JOB, or :DATA), the Command Interpreter performs the requested operation, then allows you to reactivate the suspended program by entering :RESUME. If another program command (such as :FORTRAN, :EDITOR, or :RUN) or one of the nonprogram commands (:HELLO, :BYE, :JOB, or :DATA) is entered, the Command Interpreter prints the following message at your terminal:

```
ABORT?  (YES/NO)
```

If you respond YES to the "ABORT?" message, the Command Interpreter aborts the suspended program and executes the command you entered.

If you log on using :( ) COMMAND LOGON, with a program command inside the parentheses, and then respond YES to the "ABORT?" message, MPE aborts the command and logs you off immediately.

If you respond NO to the "ABORT?" message, the Command Interpreter prints the message "COMMAND NOT ALLOWED IN BREAK. CIWARN (986)" and prompts you for another command. If you now enter :RESUME, the suspended program resumes at the point where it was interrupted.

NOTE

User programs are initiated with the BREAK facility enabled. You may programmatically disable the BREAK facility by using the FCONTROL intrinsic. Application programs can restrict user access to the Command Interpreter by disabling the BREAK function. For more information refer to the MPE V Intrinsics Manual (32033-90007).

Note that (BREAK)/:ABORT and :RESUME are valid only in sessions, not in batch jobs.

Aborting A Program

When a program contains logical errors, such as an infinite loop, you can abort it by following the steps below:

1. Pressing (BREAK). This suspends your program and MPE prompts you for a new command.
2. When you receive a colon prompt, enter :ABORT. This terminates the program, but in no way disrupts the session.

   MPE confirms the termination of the program by displaying the following message and prompting you for a new command:

   ```
   PROGRAM ABORTED PER USER REQUEST (CIERR 989)
   ```

   You can also use the (BREAK)/:ABORT sequence to abort certain MPE subsystem and command operations. This capability is indicated in the Command Specifications, under the "USE" entry.

FILES

The file system has two general classes of files:

- User Defined Files, which you or other users define, create, and make available for your own purposes.
- System Defined Files, which the file system defines and makes available to all users to indicate standard input/output files. Refer to "System Defined Files" in this section for more information.

Files are identified by their filename which optionally may be followed by the group and/or account names that reference them. A fully qualified file identifier is one in which the filename is followed by the group and account names (e.g. myfile.group.account). You may use either the filename or the fully qualified file identifier within a program to identify any given file to the system. Any arbitrary name may also be used as a file identifier (formal file designator) providing the appropriate literal file identifier (actual file designator) is equated at run time. Any file identifier may contain from one to eight alphanumeric characters, but must begin with an alphabetic character. An actual file designator includes the filename, optionally followed by a lockword, the group name, and account name. For example:

```
:FILE MYFILE/MYLOCK.GROUP.ACCOUNT
```

Backreferencing

Once you establish a set of specifications in a :FILE command, you can apply those specifications to other file references in your job or session simply by using the file's formal file designator preceded by an asterisk (*) in those references. For instance, suppose you use a :FILE command to establish the specifications shown below for the file FILEA used by program PROGA. You then run PROGA. You then wish to apply those same specifications to the file FILEB expected by PROGB, and run that program. You can use :FILE to equate the FILEA specifications to cover FILEB, as follows:

```
:FILE FILEA;DEV=TAPE;REC=-80,4,V;BUF=4   ** Establishes specifications.      **
:RUN PROGA                               ** Runs program A.                 **
:FILE FILEB=*FILEA                       ** Backreferences specifications   **
                                         ** for FILEA.                       **
:RUN PROGB                               ** Runs program B.                 **
```

This technique is called backreferencing, and the files to which it applies are known as user predefined files. Whenever you reference a predefined file in an MPE command, you must enter the asterisk before the file. This is done when using EDIT/3000 in session mode if you wish to transmit the EDIT/3000 offline listings to an offline device (such as a magnetic tape or line printer). Define the file in a :FILE command, and then specify the destination filename as a parameter in the :EDITOR command. Next, backreference the definition by using the asterisk in the :EDITOR command, as shown below:

```
:FILE L;DEV=LP          ** Specifies filenamed L as a line printer.        **
:EDITOR *L              ** Runs EDIT/3000, directs offline listing to L,   **
                        ** now established as a line printer.             **
```

When backreferencing in a command, you cannot reference any predefined formal file designator of the subsystem you are using. FORTRAN predefines FTNTEXT as its formal file designator which cannot be backreferenced using the command: FORTRAN;*FTNTEXT.

System Defined Files

MPE reserves certain actual file designators for system defined files. These file designators can be used with file equations. Following is a list of these file designators and a description of the files they reference:

$STDIN	A filename indicating the standard job or session input file (from which the job or session is initiated). For a job, this is typically a spoolfile; for a session this typically indicates a terminal devicefile.

	In the $STDIN file, data (not command) entries should not contain colon or substitute prompts in the character one position. Use the :EOD command to indicate the physical end of a datafile.
$STDINX	This is equivalent to $STDIN, except that MPE command entries (those with a colon in the character one position .) encountered are read without indicating the end of data. (The commands :EOD and ::EOF: are exceptions that always indicate the end of data and are not read.)
$STDLIST	A filename indicating the standard job or session listing file corresponding to the particular input device being used. (For each potential input device, corresponding listing device is defined during system configuration.) The listing device is customarily a printer for a batch job and a terminal for a session.
$NULL	The name of a nonexistent "ghost" file that is always treated as an empty file. When referenced as an input file by a program, that program receives only an end-of-file indication upon first access. When referenced as an output file, the associated write request is accepted by MPE but no physical output is actually performed. $NULL can be used to discard unneeded output from an executing program.
$NEWPASS	A temporary disc file used by MPE to write to during command execution. Only one such file can exist in the job or session at any one time.
$OLDPASS	The name of the temporary file last closed as $NEWPASS.

The actual file designators $NEWPASS and $OLDPASS can be used when compiling and preparing a program. When compilation begins, the compiler issues instructions to the MPE file system to create a filenamed $NEWPASS. As compilation takes place, the compiler writes the translated code (the object code) to this new file. When compilation is complete, $NEWPASS is closed and renamed $OLDPASS. In this example, $OLDPASS is the User Subprogram Library (USL) file.

When you prepare the USL file ($OLDPASS), the prepared code is placed in $NEWPASS, just as the object code was placed in $NEWPASS in the compilation stage. When preparation is complete, $NEWPASS is closed, the USL file ($OLDPASS) is purged, and $NEWPASS is then renamed $OLDPASS. Thus, $OLDPASS is now the program file, and can be executed. $OLDPASS can be made permanent by issuing the :SAVE command. For more information on file system operation refer to the MPE Segmenter Reference Manual (30000-90011) and the MPE File System Reference Manual (30000-90236).

COMMAND DEFINITIONS

SECTION II

The reference specifications for all MPE commands available to users appear in this section. For easy reference, they are presented alphabetically by command name. For each command, the reference specifications contain the following information:

- Command syntax.
- Parameter definitions, including meaning, constraints, and defaults.
- When and how the command may be used, such as: in jobs, sessions, during BREAK, or programmatically.
- Whether the command is interruptible with the (BREAK) key.
- Operation. A description of how to use the command, and any related special considerations.
- An example.
- Additional discussion, if any, in this or another manual.

CAUTION

If you have no prior experience with MPE, you should read Section I before attempting to use the reference specifications in this section.

Some commands require special capabilities. Those capabilities are discussed in Section I, "INTRODUCTION TO COMMANDS". The most frequently cited capabilities are: OP, SM, AM, CV, PS, NM, and NA.

Table 2-1 presents a functional list of the commands found in this manual, along with a notation of any special capability required to execute each command. If no special capability is mentioned, you may assume that the command is available to any user at any level.

Table 2-1. Functional List of Commands Found in this Manual.

FUNCTION	COMMAND	CAPABILITIES REQUIRED (USER Unless Specified)
Running Sessions	:( ) COMMAND LOGON :ABORT :BYE ::EOF: :EOD :HELLO :HELP :RESUME	
Running Jobs	:COMMENT :CONTINUE :ELSE :ENDIF :EOJ :IF :JOB :SET :SETJCW :SHOWJCW :STREAM	
Determining Job or Session Status and Resources	:REPORT :SHOWJOB :SHOWME	AM or SM
Job/Session Commands	=ABORTJOB =LOGOFF =LOGON =SHUTDOWN	May be executed only from the Console.
Job/Session Commands	:ABORTJOB	AM, SM, or from the Console.
Job/Session Commands	:ACCEPT :ALLOW :ALTJOB :BREAKJOB :DISALLOW :JOBFENCE :JOBSECURITY :LIMIT :LOG :REFUSE :RESUMEJOB	May be executed only from the Console, unless distributed to other terminals. OP
Job/Session Commands	:STARTSESS	PS Capability (G.01.00 and later)

Table 2-1. Functional List of Commands Found in this Manual.

FUNCTION	COMMAND	CAPABILITIES REQUIRED (USER Unless Specified)
Managing Files	:ALTSEC :BUILD :DATA :DISMOUNT :FILE :LISTEQ :LISTF :LISTFTEMP :LISTVS :MOUNT :PURGE :RELEASE :RENAME :RESET :RESTORE :SAVE :SECURE :STORE :VSUSER	UV or CV SM, OP or PM for priviledged files. SM, OP or PM for priviledged files.
Running Subsystems and User Programs Commands marked with an asterisk (*) are discussed in detail separate subsystem manuals.	:BASIC* :BASICGO* :BASICOMP* :BASICPREP* :BBASIC* :BBASICGO* :BBASICOMP* :BBASICPREP* :COBOL* :COBOLGO* :COBOLII* :COBOLIIGO* :COBOLIIPREP* :COBOLPREP* :DEBUG* :EDITOR* :FCOPY* :FORTGO* :FORTPREP* :FORTRAN* :FTN* (FORTRAN 77) :FTNGO* :FTNPREP* :PASCAL* :PASCALGO* :PASCALPREP* :PREP :PREPRUN	 PM

Table 2-1. Functional List of Commands Found in this Manual.

FUNCTION	COMMAND	CAPABILITIES REQUIRED (USER Unless Specified)
Running Subsystems and User Programs Commands marked with an asterisk (*) are discussed in detail separate subsystem manuals.	:RESETDUMP* :RPG* :RPGGO* :RPGPREP* :RUN :SEGMENTER* :SETDUMP* :SPL* :SPLGO* :SPLPREP*	
Device and Device File Status	:ASSOCIATE :CONSOLE :DISASSOCIATE :DSTAT :RECALL :SHOWALLOW :SHOWCACHE :SHOWDEV :SHOWIN :SHOWOUT	AM or SM to access other accounts or groups.
Device and Device File Commands	=ABORTIO	May be executed only from the Console.
Device and Device File Commands	:ABORTIO :ALTSPOOLFILE :CONSOLE :DELETESPOOLFILE :DOWN :DOWNLOAD :FOREIGN :GIVE :HEADOFF :HEADON :LDISMOUNT :LMOUNT :OPENQ :OUTFENCE :RESUMESPOOL :SHOWCOM :SHUTQ :STARTSPOOL :STOPSPOOL :STREAMS :SUSPENDSPOOL :TAKE :UP :VMOUNT	May be executed only from the Console, unless distributed to other terminals.

Table 2-1. Functional List of Commands Found in this Manual.

FUNCTION	COMMAND	CAPABILITIES REQUIRED (USER Unless Specified)
Requesting Utility Operations	:FREEIN :GETRIN :PTAPE :REDO :SETCATALOG :SETMSG :SHOWCATALOG :SHOWJCW :SHOWTIME :SPEED :TELL :TELLOP	
Resource Management	:ALTLOG :CHANGELOG :GETLOG :LISTLOG :RELLOG :SHOWLOGSTATUS	LG OP or LG LG LG LG
Creating Private Volumes	:VINIT	SM or OP
Message Commands	:REPLY/=REPLY :WARN :WELCOME	May be executed only from the Console, unless distributed to other terminals.
Managing Files	:ALTVSET :NEWSET :PURGEVSET :RESTORE :STORE	CV CV CV SM, OP or PM for priviledged files. SM, OP or PM for priviledged files.
Program/Procedure Allocation	:ALLOCATE :DEALLOCATE	OP OP
System Modification Backup	:SYSDUMP	OP
System Backup	:PARTBACKUP :FULLBACKUP	May be executed only from the Console, unless distributed to other terminals. OP needed.
Job and System Performance Management	:CACHECONTROL :DISCRPS :JOBPRI :SHOWCACHE :STARTCACHE :STOPCACHE :TUNE	OP or SM Only from the Console. OP OP or SM OP or SM OP or SM

Table 2-1. Functional List of Commands Found in this Manual.

FUNCTION	COMMAND	CAPABILITIES REQUIRED (USER Unless Specified)
System Reports and Information Management	:RESUMELOG	OP or from the Console.
	:SHOWLOG	OP or from the Console.
	:SHOWQ	OP or from the Console
	:SWITCHLOG	OP or from the Console.
Account Management	:ALTACCT	SM
	:ALTGROUP	AM
	:ALTUSER	AM
	:LISTACCT	AM or SM
	:LISTGROUP	AM or SM
	:LISTUSER	AM or SM
	:NEWACCT	SM
	:NEWGROUP	AM
	:NEWUSER	AM
	:PURGEACCT	SM
	:PURGEGROUP	AM
	:PURGEUSER	AM
	:REPORT	AM or SM
	:RESETACCT	SM

:( ) COMMAND LOGON

Begins a session, executes the enclosed MPE command, and ends the session upon completion of the command.

SYNTAX

```
:([:]commandname)[sessionname,]username[/userpass]
  .acctname[/acctpass][,groupname[/grouppass]]

      [;TERM=termtype]

      [;TIME=cpusecs]

                {BS}
      [;PRI=    {CS}]
                {DS}
                {ES}

      [{;INPRI=inputpriority}]
      [{;HIPRI               }]
```

PARAMETERS

commandname — Any MPE or user defined command (UDC), including its parameter list. The prompting colon may be omitted.

sessionname — Arbitrary name used in conjunction with *username* and *acctname* to form a fully qualified session identity. This name must contain from one to eight alphanumeric characters, beginning with an alphabetic character. Default is that no session name is assigned.

username — Username, established by the Account Manager, that allows you to logon under this account. The name must contain from one to eight alphanumeric characters, beginning with an alphabetic character.

userpass — User password, optionally assigned by the Account Manager. This password must contain from one to eight alphanumeric characters, beginning with an alphabetic character. A slash (/) must precede the *userpass* parameter.

acctname — Account name as established by the System Manager. This name must contain from one to eight alphanumeric characters, beginning with an alphabetic character. A period (.) must precede the *acctname* parameter.

acctpass — Account password, optionally assigned by the Account Manager. This password must contain from one to eight alphanumeric characters, beginning with an alphabetic character. A slash must precede the *acctpass* parameter.

groupname Group name to be used for the local file domain and the CPU time charges, as established by the Account Manager, containing from one to eight alphanumeric characters, beginning with an alphabetic character. Default is your home group, if one is assigned. The parameter is required if a home group is not assigned.

grouppass Group password, optionally assigned by the Account Manager. This password must contain from one to eight alphanumeric characters, beginning with an alphabetic character. You do not need to enter the *grouppass* parameter when you logon to your home group. If, you request a logon group that is assigned a password, you must supply the password when you logon to that group. A slash (/) must precede the *grouppass* parameter.

termtype Determines terminal-type characteristics. The value assigned to the *termtype* parameter determines the type of terminal used for input and its characteristics. The default value for *termtype*, a number from 0 to 22, is assigned by the System Supervisor during system configuration. If your terminal is not of the default *termtype*, you must supply this parameter to ensure correct terminal input and output. For more information, refer to Appendix A "TERMINALS SUPPORTED BY MPE".

The *termtype* parameter is the name of the file containing the desired terminal-type characteristics. The file may not have a lockword or reside on a Private Volume.

Users of the Workstation Configurator are allowed to create terminal-type files. The proper and efficient operation of a specific device by a user created terminal-type is the responsibility of the user. The Workstation Configurator utility allows the user to specify the following characteristics of the terminal: data flow control, block mode, read trigger, special characteristics, echo, line feed, parity, and printer control.

If group and/or account names are omitted, the logon group and/or account name is substituted. For further information on the terminal-type characteristics, refer to the Fundamental Communications Handbook (5957-4634) and the Workstation Configurator Reference Manual (30000-90001). Refer also to Appendix A, "TERMINALS SUPPORTED BY MPE", in Section III of the manual.

cpusecs The maximum CPU time that your session can use, entered in seconds. When this limit is reached, session is aborted. It may be any value from 1 to 32767, provided that it does not limit imposed by the System or Account Manager. To specify no limit, specify question mark (?) or UNLIM, or omit this parameter. Default is no limit.

BS, CS, DS, or ES The execution priority queue that the Command Interpreter uses for your session. It is also the default priority for all programs executed within the session. BS is highest priority, ES is lowest. If you specify a priority that exceeds the highest permitted by the system, for your account or username by the system, MPE assigns the highest priority possible below BS. DS and ES are intended primarily for batch jobs. Using them for sessions may produce less than optimum performance. sessions is generally discouraged.

CAUTION

Use care in assigning the BS queue, because processes in this priority class may lock out other processes (MPE uses the BS queue).

For information on the guidelines for use of these priority queues, refer to the :TUNE command in the MPE V System Operation and Resource Management Reference Manual (32033-90005). Default is CS.

inputpriority or HIPRI

Determines the input priority of the job or session. It may be a value from 1 (lowest priority) to 13 (highest priority). The default is 8. If you supply a value less than or equal to the current job fence as set by the Operator, or the job fence is equal to or greater than the default *inputpriority*, the session will be denied access.

The ;HIPRI option has two purposes: to override the system job fence, or to override the session limit. When using the ;HIPRI option to override the job fence, the system will first check to see whether you have either System Manager (SM) or System Supervisor (OP) capability. If you have one or the other, you will be logged on, and your ;INPRI will default to the system's job fence and execution limit. If you do not have either capability, you will receive the following warning:

```
MUST HAVE 'SM' OR 'OP' CAP. TO SPECIFY HIPRI,
 MAXIMUM INPRI OF 13 IS USED (CIWARN 1460)
```

Despite this warning, the system will still allow you to logon if your default input priority (8) exceeds the job fence. If the job fence exceeds your input priority, you will not be able to logon unless you have SM or OP capability.

USE

This command may not be used from a Session or Job. It is not available from BREAK, or a Program. It is not Breakable.

Pressing [BREAK] during the output of the logon message terminates the rest of the message and causes the immediate execution of the command within the parentheses. Pressing [BREAK] during the execution of the command within the parentheses interrupts the execution, if the command is breakable. If there is an option logon UDC that is not a program, the UDC is not breakable and execution is forced at logon time.

OPERATION

This command initiates a session and immediately executes the command contained within the parentheses. Once the command or subsystem specified inside the parentheses is executing, all input, output, BREAK capabilities, and prompts operate as described in the command or subsystem instructions. After the execution of the command or subsystem is completed, normally or through an abort, a logoff is performed automatically.

Note that this command has the following limitations:

- You must enter the :( ) COMMAND LOGON at a terminal. No other device can be used for this command.
- Unlike the :HELLO command, :( ) COMMAND LOGON is not recognized within sessions.
- Use of commands which would normally cause an end-of-file, such as :HELLO, :BYE, :JOB, :DATA, :EOD, ::EOF:, and :EOJ, will cause the session to terminate as soon as as they are encountered by the MPE Command Interpreter.
- Only one MPE command is allowed within the parentheses. Leading and trailing blanks are suppressed within the parentheses.

If you omit any passwords required in the :( ) COMMAND LOGON, MPE prompts you for them individually.

When MPE initiates the session, it displays "HP 3000", the user defined and official (base) *version.update.fix* number (an HP method of distinguishing between software releases), the date, and the time as follows:

```
HP3000 / MPE V  G.02.00 (BASE G.02.00).  MON, MAY 14, 1985,  8:22 AM
```

After the logon message is printed at your terminal, MPE executes the command specified within the parentheses. The information MPE generates as a result of this command is transmitted to your terminal immediately; just as if you had called that command or subsystem during a session. When the command or subsystem is exited, MPE terminates the session at once, and displays the central processor (CPU) time used (in seconds), the connect time (in minutes), and the date and time, as follows:

```
CPU=1. CONNECT=1. MON, MAY 14, 1985,  8:22 AM
```

EXAMPLE

To logon under *username* MAC, *acctname* TECHPUB, and *groupname* XGROUP, and execute the :LISTF command, enter:

```
:(LISTF) MAC.TECHPUB,XGROUP

HP3000 / MPE V  G.02.00 (BASE G.02.00).  MON, MAY 14, 1985,  8:22 AM

FILENAME

FIRST     GP005     TRIAL     XPROG

CPU=1. CONNECT=1. MON, MAY 14, 1985,  8:22 AM
```

:ABORT

Aborts the current program or operation.

SYNTAX

```
:ABORT
```

PARAMETERS

None.

USE

This command may be issued from a Session (in BREAK only). It is not executable from a Job, or a program. It is not Breakable.

OPERATION

After you suspend a program or MPE command operation by pressing (BREAK), the :ABORT command terminates that program or operation. Programs do not terminate while critical system code is executing on their behalf, but will terminate immediately following execution of that code. The :ABORT command is available only from a session and only during a BREAK, but it does not disrupt the session. Some operations abort immediately upon entering BREAK without requiring the :ABORT command. An :ABORT command results in the Job Control Word (JCW) being set to the "SYSTEM 0" state. For a discussion of Job Control Words, refer to the :SETJCW command in this section.

EXAMPLE

To abort the current operation, enter:

```
:ABORT
```

:ABORTIO/=ABORTIO

Aborts one pending I/O request for a device.

SYNTAX

```
:ABORTIO ldev
```

PARAMETERS

ldev	The logical device number of the device for which I/O is being aborted.

USE

This command may be issued from a Session, or Job. It is available from within BREAK, or a program. It is not breakable. /=ABORTIO is executable only from the Console.

* Unless distributed to users with the :ALLOW or :ASSOCIATE command.

OPERATION

This command aborts all pending I/O operations on the specified *ldev*. To delete all queued I/O requests for a device, repeat the :ABORTIO command until the following message is printed on the $STDLIST device:

```
NO I/O TO ABORT FOR DEVICE #ldev
```

Those devices which are :JOB or :DATA accepting always have outstanding read requests pending, due to the auto-recognition feature of MPE. The :ABORTIO command is used to clear these pending input requests.

To completely clear spooled devices, several :ABORTIO commands may be necessary. Issue the :ABORTIO command until there are no queued I/O requests, and the message above appears on the $STDLIST device.

Clearing all outstanding I/O requests is sometimes required in order to allow proper execution of other Console commands. In certain cases the :ABORTJOB, :TAKE, :DOWN, and :REFUSE commands will not execute unless an :ABORTIO command is first issued to clear pending I/O operations on the appropriate device.

NOTE

If :ABORTIO is not effective from the System Console, =ABORTIO may be used. The =ABORTIO command should be used only when the :ABORTIO command cannot be executed, or when the Console is busy.

EXAMPLES

To abort all pending I/O requests for logical device 53 , enter:

```
:ABORTIO 53
11:16/3/NO I/O TO ABORT FOR DEVICE 53
```

It is necessary to issue several :ABORTIO commands to abort all pending I/O operations on a spooled device, as shown below:

```
:STOPSPOOL 5
11:20/31/SP#5/STOPPED
11:20/31/LDEV#5 NOT READY
:REFUSE 5
:ABORTIO 5
:ABORTIO 5
11.21/40/NO I/O TO ABORT FOR DEVICE 5
```

:ABORTJOB/=ABORTJOB

Aborts a job or session.

SYNTAX

```
          {#Jnnn                }
:ABORTJOB {#Snnn                }
          {[jobname,]user.acct  }
```

PARAMETERS

#Jnnn	A job number.
#Snnn	A session number.
jobname	The name of the job, as identified by the :SHOWJOB command.
user	A username.
acct	An account name.

USE

This command may be issued from a Session, Job, Program or in BREAK. It is not Breakable . It is executable only from the Console*, or by a user with Account Manager (AM) or System Supervisor (SM) capabilities. The :ABORTJOB command may be issued only from the Console.

* Unless distributed to users with the :ALLOW command, or with :JOBSECURITY set LOW.

OPERATION

This command terminates the designated job or session, and displays the following message on the job/session list device:

```
SESSION ABORTED BY SYSTEM MANAGEMENT
```

If you use the "[*jobname*,]*user.acct*" form of the command when there is more than one job or session executing under that name, MPE will randomly select which job/session to abort. Therefore, to exercise the more precise control when aborting jobs or sessions, use the *#Jnnn* or *#Snnn* form of the :ABORTJOB command. Although the job/session is abnormally terminated, log records are issued, and CPU and connect times are updated. Any I/O activity (i.e. printing, :STORE , etc) will also be terminated.

The :ABORTJOB command can be applied to waiting and scheduled jobs, as well as to executing jobs. If the spooler input file ($STDIN) for a batch job has been created and not yet opened (i.e. the job is in the WAIT state), the entire file will be deleted. If the :ABORTJOB command is issued before the output spoolfile is complete, only that portion of the file already spooled will be printed, along with an error message indicating that the job was aborted.

If a request is pending at the System Console, it will be terminated automatically by the :ABORTJOB/=ABORTJOB command. The following message will appear on the system console:

time/#S*nnn*/*pin*/REQUEST REQUIRING OPERATOR REPLY FOR PIN #*nn* HAS BEEN ABORTED

When the :ABORTJOB command is successful, a logoff message is displayed, indicating that the job being executed has been aborted, as shown in the example below:

```
:ABORTJOB #S9
11:20/#S9/34/LOGOFF ON LDEV #77
```

Two messages will also be displayed on the user's terminal: the standard error message that appears when a request is manually terminated by a =REPLY of 0 or N and:

SESSION ABORTED BY SYSTEM MANAGEMENT

The =ABORTJOB command may be used at the Console if :ABORTJOB is ineffective. Refer to the "USE" section of this command. busy.

EXAMPLES

To terminate session number 139 , enter:

```
:ABORTJOB #S139
17:10/#S139/34/LOGOFF ON LDEV #62
```

To terminate job number 9 , enter:

```
:ABORTJOB #J9
 20:18/#J9/26/LOGOFF ON LDEV #10
```

Note that in both cases, the execution was successful, and a logoff message was printed.

To terminate session 6 , which has a pending device allocation message, enter:

```
?17:00/#S6/23/LDEV# FOR "SCRTAPE" ON TAPE (NUM)?
:ABORTJOB #S6
17:10/#S6/120/REQUEST REQUIRING OPERATOR REPLY FOR PIN 23
   HAS BEEN ABORTED
17:10/#S6/120/LOGOFF ON LDEV #58
```

:ACCEPT

Permits a designated device to accept jobs/sessions and/or data.

SYNTAX

```
:ACCEPT [JOBS ]
        [DATA ] ldev
```

PARAMETERS

JOBS	Specifies that the designated device will recognize the :JOB and :HELLO commands. The device must be interactive to support sessions.
DATA	Specifies that the designated device will recognize :DATA commands. designated device.
ldev	The logical device number of the device for which the :JOB, :HELLO, and/or :DATA commands are being enabled.

NOTE

If both the JOBS and the DATA parameters are omitted, then both the :JOB and :HELLO commands, and the :DATA command are allowed.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It is executable only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

The Operator or System Supervisor uses this command to designate which devices may be used to initiate jobs or sessions and/or data. When a device is configured as an "accepting" device, MPE will automatically scan the first input record for a valid :JOB, :HELLO, or :DATA command. This feature, called "auto-recognition", allows users to access the device without specifically requesting use of the device with a message to the system console.

If the JOBS parameter is explicitly requested, the :ACCEPT command will not be executed unless the device has been configured with a default output device. Refer to "RECONFIGURING THE SYSTEM" in the MPE V System Operation and Resource Management Reference Manual (32002-90005).

EXAMPLES

To permit logical device 19 to accept jobs and data, enter:

```
:ACCEPT 19
:SHOWDEV 19
LDEV   AVAIL       OWNERSHIP      VOLID     ASSOCIATION

  19 A AVAIL
```

To permit logical device 19 to accept jobs and data, and to allow the device to be spooled, enter:

```
:ACCEPT 19
:STARTSPOOL 19
11:12/31/SP#/SPOOLED IN
11:12/6/LDEV#19 NOT READY
:SHOWDEV 19
 DEV  AVAIL          OWNERSHIP          VOLID          DEN   ASSOCIATION

 19   SPOOLED        SPOOLER OUT
```

:ALLOCATE

Permanently loads a program or procedure into main memory.

SYNTAX

```
:ALLOCATE [PROCEDURE ]
          [PROGRAM   ]  name
```

PARAMETERS

PROCEDURE	Specifies that the procedure in SL.PUB.SYS indicated by *name* is to be allocated. The default is PROGRAM.
PROGRAM	Specifies that the program file indicated by *name* is to be allocated. Default.
name	The name of the program file or procedure to be allocated.

USE

This command may be issued from a Session or Job. It may not be issued from a Program, or in BREAK, and is not Breakable. A user must have System Supervisor (OP) capability to use this command.

OPERATION

A program or procedure is allocated by resolving external references and assigning Code Segment Table (CST) or Extended Code Segment Table (XCST) entries to the program's code segments. Table entries are also allocated for any procedures called by the allocated program or procedure. Allocating a program or procedure will not increase execution speed. However, it will reduce the time taken to load the program for execution.

CAUTION

Use care in deciding which programs or procedures to load with :ALLOCATE. The number of CST table entries is limited. Exceeding the limit runs the risk of losing data.

Segments remain loaded until they are deallocated with the :DEALLOCATE command, or until the system is shut down or a system failure occurs. Programs or procedures must be reallocated with the :ALLOCATE command following any start up.

The user issuing the :ALLOCATE command must have EXECUTE access for any file referenced in the *name* parameter of this command.

Any external procedures referenced by a program being allocated by this command must reside in SL.PUB.SYS.

EXAMPLE

To allocate a procedure identified as PROC1 , that resides in SL.PUB.SYS , enter:

```
:ALLOCATE PROCEDURE,PROC1
```

NOTE

Program files residing in the non-system domain (a private volume) will not be allocated. Attempts to do so will result in a "LOAD ERR 92 " message.

ADDITIONAL DISCUSSION

Refer to the discussion of Program Allocation in "CHANGING OTHER SYSTEM PARAMETERS" in the MPE V System Operation and Resource Management Reference Manual (32033-90005).

:ALLOW

Grants a user access to a specific Operator command.

SYNTAX

```
:ALLOW FILE=formaldesignator[;SHOW]

        {@.@      }
:ALLOW  {user.@   }[;]COMMANDS=command[[,]command,...]
        {@.acct   }
        {user.acct}
```

PARAMETERS

formaldesignator	A formal ASCII filename, which may consist of one to eight alphanumeric characters, beginning with an alphabetic character.
SHOW	Lists input lines on $STDLIST.
@.@	Grants access to all users whether logged on or not.
user.@	Grants access to a specific user in all accounts.
@.acct	Grants access to all users in a specific account.
user.acct	Grants access to a specific user in a specific account.
command	Specifies the names of those commands to which the user is granted access.

USE

This command can be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It is executable only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

The Operator uses the :ALLOW command to distribute specific Operator commands to system users. You indicate at the console which users may execute Operator command(s), then the names of each command they will be allowed to execute.

This command may also be executed in indirect and subsystem modes. You must create a file that contains records identifying the user(s) and account(s) to whom you are allowing Operator commands, followed by the list of commands allowed, such as:

```
:EDITOR
HP32201A.7.16 EDIT/3000  TUES, MAY 29, 1985,  5:08 PM
(C) HEWLETT-PACKARD CO. 1984
/ADD
    1      SUSAN.PAYROLL;COMMANDS=ALTJOB,ALTSPOOLFILE
    2      JOHN.ACCTNG;COMMANDS=ALTSPOOLFILE,DELETESPOOLFILE
    3      //
...
/KEEP ALLOWTMP
/E
```

Then issue the :ALLOW command, using the the ;SHOW parameter to display each command line as it is executed from the file.

```
:ALLOW FILE=ALLOWTMP;SHOW
```

In subsystem mode, enter ":ALLOW", followed by (RETURN). You will be prompted with a "> ". Command parameters are accepted until an end-of-file is received, or until you EXIT.

The Operator may :ALLOW console capabilities only to users who are currently logged on to the system, unless the @.@ option of the :ALLOW command is used. In this case, all users are affected, whether or not they are logged on. Additional capabilities granted to a user last only for the duration of their current session: once the user logs off, any special capabilities previously assigned are no longer applicable.

The user command :SHOWALLOW indicates which Operator commands have been allowed globally and to that specific user. Refer to :SHOWALLOW in this section.

EXAMPLES

To give the user MAC.TECH the ability to execute the :REPLY and :ABORTIO commands, you would enter the following on the system console:

```
:ALLOW MAC.TECH;COMMANDS=REPLY,ABORTIO
```

In subsystem mode, to give the user MGR.MANUALS the ability to execute the BREAKJOB command, you would enter the following on the system console:

```
:ALLOW
>MGR.MANUALS;COMMANDS=BREAKJOB
>EXIT
```

Changes the attributes of an existing account.

SYNTAX

```
:ALTACCT acctname
    [;PASS=[password]]
    [;FILES=[filespace]]
    [;CPU=[cpu]]
    [;CONNECT=[connect]]
    [;CAP=[capabilitylist]]
    [;ACCESS=[(fileaccess)]]
    [;MAXPRI=[subqueuename]]
    [;LOCATTR=[localattribute]]
    [;VS=[volset;{ALT  }]]
                 {SPAN}
```

PARAMETERS

acctname — The name of the account to be altered.

password — Account password (used only for verifying logon access). If this parameter is omitted, no change is made. If the Operator enters ";PASS = RETURN", then the existing password is removed.

filespace — Disc storage limit, in sectors, for the permanent files in the account. The *filespace* limit cannot be less than the the number of sectors currently in use for the account. Default is unlimited file space.

cpu — The limit on cumulative CPU time, in seconds, for the account. This limit is checked only when a job or session is initiated, and therefore never causes the job or session to abort. The maximum value allowed is 2,147,483,647 seconds. Default is unlimited CPU time. The counter may be set to zero with the :RESETACCT command.

connect — The limit on total cumulative job or session connect time, in minutes, allowed the account. This limit is checked at logon and every time a process terminates. The maximum value allowed is 2,147,483,647 minutes. Default is unlimited connect time. The counter may be set to zero with the :RESETACCT command.

capabilitylist List of capabilities, separated by commas, permitted the account. Each capability is denoted by a two-letter mnemonic, as follows.

System Manager	=	SM
Account Manager	=	AM
Account Librarian	=	AL
Group Librarian	=	GL
Diagnostician	=	DI
System Supervisor	=	OP
Network Administrator	=	NA
Node Manager	=	NM
Permanent Files	=	SF
Access of nonsharable I/O devices	=	ND
Use Volumes	=	UV
Create Volumes	=	CV
Use Communications Subsystem	=	CS
Programmatic Sessions	=	PS
User Logging	=	LG
Process Handling	=	PH
Extra Data Segments	=	DS
Multiple RINs	=	MR
Privileged Mode	=	PM
Interactive Access	=	IA
Local Batch Access	=	BA

Default is "AM,AL,GL,SF,ND,IA,BA ", except for the SYS account. This account has no true default: it is assigned the maximum account capabilities when the system is delivered and, under normal circumstances, should not be altered. Note that CV capability, which permits account members to create Private Volumes, automatically gives the account UV capability, allowing account members to use Private Volumes.

fileaccess The restrictions on file access pertinent to this account. Default is R,A,L,W,X:AC, entered as follows:

```
                {R}
                {L}
([{A}[,...]:{ANY} ][;...])
                {AC }
                {W}
                {X}
```

where R , L , A , W , and/or X specify modes of access by types of users (ANY and/or AC) as follows:

R	=	Read
L	=	Lock (allows exclusive access to file)
A	=	Append (implicitly specifies L also)
W	=	Write (implicitly specifies A and L also)
X	=	Execute

The user types are specified as follows.

ANY = Any user
AC = Member of this account only

subqueuename Name of the highest priority subqueue that can be requested by any process of any job/session in the account, specified as AS, BS, CS, DS, or ES. Default is CS.

CAUTION

User processes executing in the AS or BS subqueues can deadlock the system. If you assign these subqueues to non-priority processes, other critical system processes may be prevented from executing. Exercise extreme caution when choosing subqueues.

localattribute Local attribute of the account, as defined at the installation site. This is a double word bit map, of arbitrary meaning, that might be used to further classify accounts. While it is not involved in standard MPE security provisions, it is available to processes through the WHO intrinsic. Programmers may use *localattribute* in their own programs to provide security. Default is double word 0 (null).

volset The volume set or class reference which, when fully qualified, is in the form *vcsid.groupname.acctname* where *vcsid* refers to a previously defined volume set or class definition.

SPAN Indicates that the *acctname* is to be inserted in the accounting directory of the specified volume set (*volset*). The specified volume set must have been previously mounted (via a :MOUNT command) for the SPAN operation to succeed.

ALT Directs the altering of an account or group entry on the specified volume set. This option is useful only if it is necessary to alter account and group file space limits for entries that have already been spanned.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have System Manager (SM) capability to use this command.

OPERATION

The System Manager uses :ALTACCT to change the attributes of an existing account. Multiple keywords may be entered on a single command line as shown in "EXAMPLE". When you change one capability in a *capabilitylist* containing several nondefault values, you must respecify the entire new *capabilitylist*. When an entire keyword parameter group is omitted from the :ALTACCT command, that parameter remains unchanged for the account. When a keyword is included, but the corresponding parameter is omitted (as in ";PASS= "), the default value is assigned.

PARAMETER	DEFAULT VALUES
password	No password
filespace	Unlimited
cpu	Unlimited
connect	Unlimited
capabilitylist	AM,AL,GL,SF,ND,IA,BA (All accounts except SYS)
	SM,AM,AL,GL,DI,OP,SF,ND,PH,DS,MR,PM,IA,BA (SYS account only)
fileaccess	(R,A,W,L,X:AC) (All accounts except SYS)
	(R,X:ANY;A,W,L:AC) (SYS account only)
subqueuename	CS subqueue
localattribute	Double word 0 (null)

Any value changed with :ALTACCT will take effect the next time MPE is requested to check the value. If an attribute is removed from an account while users are logged on, they will not be affected until they end the job or session and logon again. MPE does not automatically generate a message informing users of the change; it is your responsibility to warn account members in advance of any changes. If you take a capability away from an account, all account members and groups within the account also lose the capability.

You cannot remove System Manager (SM) capability from the SYS account. You also cannot take AM capability away from any account. If more than one person is assigned AM capability within the account, you can remove AM capability from all but one (the last) user assigned it.

EXAMPLE

To change an account named AC2 so that its *password* is GLOBALX , and its *filespace* is limited to 50,000 sectors, enter:

```
:ALTACCT AC2;PASS=GLOBALX;FILES=50000
```

:ALTGROUP

Changes one or more attributes of a group.

SYNTAX

```
:ALTGROUP groupname
    [;PASS=[password]]
    [;CAP=[capabilitylist]]
    [;FILES=[filespace]]
    [;CPU=[cpu]]
    [;CONNECT=[connect]]
    [;ACCESS=[(fileaccess)]]
    [;VS=[volset{ALT }][;...]]
                 {SPAN}
```

PARAMETERS

groupname — Specifies the name of the group whose attributes are to be changed.

password — The new password to be assigned to the group (used to verify logon access only). If this parameter is omitted, no change is made. If the Operator enters "; PASS = RETURN " then the existing password is removed.

capabilitylist — Specifies the list of capabilities permitted this group. Each capability is denoted by a two-letter mnemonic, as follows.

Process Handling	=	PH
Extra Data Segments	=	DS
Multiple RINs	=	MR
Privileged Mode	=	PM
Interactive Access	=	IA
Local Batch Access	=	BA

Two or more capabilities may be specified if they are separated by commas.

Default is IA,BA except for the PUB group of the SYS account. This group has no true default. It is assigned the maximum group capabilities when the system is delivered and should not normally be changed.

filespace — Disc storage limit, in sectors, for the permanent files of the group. A group's *filespace* cannot be set to a value greater than the corresponding limits currently defined for the group's account. If an account's *filespace* is later changed, the group limits are automatically changed. Default is unlimited file space.

cpu

The limit on the total cumulative CPU time, in seconds, for the group. This limit is checked only when a job or session is initiated; the limit never causes a job/session to abort. The maximum value you may specify with this command is 2,147,483,647 seconds. If the limit is exceeded, users with Account Manager capability will be warned when logging on; other users will be denied access.

The CPU limit for a group cannot set to a value greater than the corresponding limit currently defined for the account in which that group resides. If the CPU limit for an account is changed later, group limits are automatically changed. Default is unlimited CPU time. The counter may be set to zero with the :RESETACCT command.

connect

The limit on the total cumulative job or session connect time, in minutes, that the group is allowed. This limit is checked at logon, and whenever the job or session initiates a new process. The maximum value you may specify with this command is 2,147,483,647 minutes. If the limit is exceeded, users with Account Manager capability will be warned when logging on; other users will be denied access.

A group's connect limit cannot be specified as greater than the corresponding limit currently defined for the account in which the group resides. However, if the connect limit of the account is changed later, its groups' limits will automatically be changed. Default is unlimited connect time. The counter may be set to zero with the :RESETACCT command.

fileaccess

The restriction on file access pertinent to this group. Default is R,X:ANY;A,W,L,S:AL,GU for the Public Group (PUB); and R,A,W,L,X,S:GU for all other groups, entered as follows:

```
{R}                {ANY}
{L}                {AC }
{A} [;...]:        {GU }[;...]
{W}                {AL }
{X}                {GL }
                   {CR }
```

where R, L, A, W, and X specify modes of access by types of users (ANY, AC, GU, AL, GL, CR) as follows:

R = Read
L = Lock (allows exclusive access to the file)
A = Append (implicitly specifies L also)
W = Write (implicitly specifies A and L also)
X = Execute
S = Save

The user types are specified as follows:

ANY = Any user
AC = Member of this account only
GU = Member of this group only
AL = Account librarian user only
GL = Group librarian user only
CR = Creating user only

Two or more user or access types may be specified if they are separated by commas.

volset — A volume set or class reference which, when fully qualified, is in the following format:

vcsid.groupname.acctname

where *vcsid* refers to a previously defined volume set or class definition.

If *volset* is different from the old volume set, and the old volume set is the system set, then the old volume set must be examined for the presence of files belonging to *groupname*. If no such files are found, the command will succeed, and the group is reassigned to *volset* as its new home volume set.

If *volset* is omitted from the VS= parameter, the group will be reassigned to the system volume set. If the old volume set is already the system volume set, nothing will be altered.

It is permissible to reassign a group to a different volume set despite the presence of files belonging to *groupname*, as long as the old volume set is not the system volume set and the *groupname* is not currently bound to its home volume set, either explicitly via the :MOUNT command, or implicitly, via the FOPEN intrinsic.

SPAN — Indicates that the *groupname* is to be inserted in the accounting directory of the specified volume set (*volset*). The specified volume set must have been previously mounted, via the :MOUNT command, for the SPAN operation to succeed.

ALT — Directs the altering of an account or group entry on the specified volume set. The ALT option is required to alter account and group file space limits for entries that have already been spanned. This option is also required for changing security specifications on a volume set.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have Account Manager (AM) capability to use this command.

OPERATION

This command changes one or more attributes of a group; multiple parameters may be specified on a single command line as shown in "EXAMPLE". When an entire parameter is omitted from an :ALTGROUP command, the corresponding value for the group remains unchanged. When a keyword is included but the corresponding parameter is omitted (as in " ;PASS="), the default value is assigned as follows:

PARAMETERS	DEFAULT VALUES
password	Null (No password)
capabilitylist	IA,BA (except PUB.SYS) PH,DS,MR,PM,IA,BA (PUB.SYS only)
filespace	Unlimited
cpu	Unlimited
connect	Unlimited
fileaccess	R,A,W,L,X,S:GU (All groups except PUB) R,X:ANY;A,W,L,S:AL,G,U (PUB group only)

When a parameter is modified through the :ALTGROUP command, it immediately takes effect in the directory. It does not affect any active users with open files in the group, until they terminate their session and logon to that username and group again. For this reason, it is a good idea to notify all group users of any planned changes in advance.

EXAMPLE

To assign a new password, PASS2 , to a group named GROUPX , enter:

```
:ALTGROUP GROUPX; PASS=PASS2
```

:ALTJOB

Alters the attributes of waiting or scheduled jobs.

SYNTAX

```
:ALTJOB {#Jnnn}
        {#Snnn}

    [;INPRI=inputpriority]
    [;OUTDEV={ldev     }]
             {devclass}
```

PARAMETERS

#Jnnn	A job number.
#Snnn	A session number. (Although syntactically correct, this parameter is rarely used; sessions do not wait.)
inputpriority	Specifies the new input priority (0 = lowest; 14 = highest).
ldev or *devclass*	Specifies the logical device number or device class name of the destination device job's $STDLIST.

USE

This command may be issued from a Session, Job, Program or in BREAK. It is not Breakable. Users may execute :ALTJOB to change their own jobs. To change other users' jobs, the command must be issued from the Console.*

* Unless distributed to users with the :ALLOW command, or if :JOBSECURITY is set LOW.

OPERATION

The :ALTJOB command, in conjunction with the :JOBFENCE command, allows you to control the flow of all jobs on the system with the exception of HIPRI jobs. It can be used to alter only jobs in the INTRO and WAIT, or SCHED state. Jobs with input priority less than or equal to the current JOBFENCE, a numerical value from 0 to 14, will be deferred.

EXAMPLE

Consider three jobs that are submitted by a user and suppose that all of these jobs have an INPRI value of 8 . To change their INPRI values to ensure that JOB1 runs first, JOB2 runs last, and JOB3 runs second with LP allocated as the OUTDEV for JOB3 , enter the following commands:

```
:JOBFENCE 14
15:11/#J1/24/DEFERRED JOB INTRODUCED ON LDEV #53    **Prevents all jobs      **
15:11/#J2/25/DEFERRED JOB INTRODUCED ON LDEV #53    **from running while     **
15:13/#J3/26/DEFERRED JOB INTRODUCED ON LDEV #53    **the user changes       **
                                                    **the INPRI values       **
:SHOWJOB

JOBNUM   STATE  IPRI  JIN   JLIST  INTRODUCED  JOB NAME

#S23     EXEC         20    20     THU  2:15P  OPERATOR.SYS
#J1      WAIT   D 8   10S   12     THU  3:11P  JOB2,OPERATOR.SYS
#J2      WAIT   D 8   10S   12     THU  3:11P  JOB3,SARAH.PAYROLL
#J3      WAIT   D 8   10S   12     THU  3:13P  JOB1,JOHN.ACCTNG

4 JOBS:
    0 INTRO
    3 WAIT; INCL 3 DEFERRED
    1 EXEC; INCL 1 SESSIONS
    0 SUSP
JOBFENCE= 14; JLIMIT= 5; SLIMIT= 16

:ALTJOB #J1;INPRI=10
:ALTJOB #J3;INPRI=9;OUTDEV=LP
:ALTJOB #J2;INPRI=8
:SHOWJOB

JOBNUM   STATE  IPRI  JIN   JLIST  INTRODUCED  JOB NAME

#S23     EXEC         20    20     THU  2:15P  OPERATOR.SYS
#J6      WAIT   D12   10S   12     THU  3:13P  JOB1,JOHN.ACCTNG
#J5      WAIT   D11   10S   LP     THU  3:11P  JOB3,SARAH.PAYROLL
#J4      WAIT   D 9   10S   12     THU  3:11P  JOB2,OPERATOR.SYS

4 JOBS:
    0 INTRO
    3 WAIT; INCL 3 DEFERRED
    1 EXEC; INCL 1 SESSIONS
    0 SUSP
JOBFENCE= 6; JLIMIT= 5; SLIMIT=16          **Resetting the jobfence at 6 **
                                           **permits all jobs with INPRI **
                                           **values of 7 or above to run **
```

:ALTLOG

Alters the attributes of an existing user logging identifier.

SYNTAX

```
:ALTLOG logid

                  {;DISC }
   [;LOG=logfile  {;TAPE }]
                  {;SDISC}
                  {;CTAPE}

   [;PASS=password]

   [{;AUTO  }]
    {;NOAUTO}
```

PARAMETERS

logid	The logging identifier whose attributes are to be changed. This identifier must contain from one to eight alphanumeric characters, beginning with an alphabetic character.
logfile	The name of the file to receive data from the logging procedure. This name must contain from one to eight alphanumeric characters, beginning with an alphabetic character. You must specify the device class on which *logfile* resides, i.e. DISC, TAPE, SDISC, or CTAPE.
password	The new password for the logging identifier. This password must contain from one to eight alphanumeric characters, beginning with an alphabetic character.
AUTO	Initiates an automatic :CHANGELOG if the current logfile becomes full. Refer to :CHANGELOG in this section.
NOAUTO	Prevents the initiation of an automatic :CHANGELOG. A :CHANGELOG will not be performed if the current logfile becomes full. Default.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It requires User Logging (LG) capability.

OPERATION

This command changes the attributes of an existing user logging identifier to those specified in the parameter list. Parameters not included in the :ALTLOG command retain their current values. You must have User Logging (LG) capability to execute :ALTLOG. Only the creator of the logging identifier can alter its attributes.

To use the AUTO parameter, the log process for *logid* must be enabled for changing. You may do this by ending the log filename with the numeric characters "001" (e.g. *fname001*). This name convention works in conjunction with the fileset number to generator sequential filenames automatically.

NOTE

If a new log filename is specified with the :ALTLOG command, the links with any previous log file will be broken.

EXAMPLE

To change the destination *logfile* of the logging identifier KIM to *logfile* C and specify that C resides on disc, enter:

```
:ALTLOG KIM;LOG=C,DISC
```

Since the keyword parameter, "PASS=", was not included, KIM retains any password previously specified.

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005). MPE Intrinsics Reference Manual (32003-90007)

:ALTSEC

Permanently changes a file's security provisions.

SYNTAX

```
:ALTSEC filereference[;([fileaccess[;[fileaccess][;...]]])]
```

PARAMETERS

filereference Actual file designator of the file whose security provisions are to be altered, written in the format:

filename[/*lockword*][.*groupname*[.*acctname*]]

In a batch job if a lockword exists, it must be specified. In a session if a lockword exists and is omitted, MPE will prompt for it.

fileaccess File security specifications, entered as follows:

```
{R}              {ANY}
{L}              {AC }
{A} [;...][;]    {GU }[;...]
{W}              {AL }
{X}              {GL }
                 {CR }
```

AL, GL, CR) as follows:

R = Read
L = Lock (allows opening with dynamic locking option)
A = Append (implicitly specifies L also)
W = Write (implicitly specifies A and L also)
X = Execute

Two or more modes may be specified if they are separated by commas. The user types are specified as follows:

ANY = Any user
AC = Member of this account only
GU = Member of this group only
AL = Account librarian user only
GL = Group librarian user only
CR = Creating user only

Two or more user types may be specified if they are separated by commas. Default is R, L, W, A, X: ANY. The colon (:) separating one or more modes from one or more user types is required punctuation in the specification of *fileaccess*.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

The :ALTSEC command enables you to change the security provisions for any disc file that you have created, by permanently deleting all previous provisions specified for this file at the file level, and replacing them with those defined in the :ALTSEC parameters. Account level and group level security is not affected, nor is the lockword (if one exists).

This command can be used by only the creator of the file, and the file must be a permanent disc file. This command will fail if the group's home volume set is not mounted. When the normal (default) MPE security provisions are in effect, the file must be in your logon account and must belong to your logon or home group.

Even though you may be barred by file access restrictions from accessing a file that you have created, you still can issue an :ALTSEC command for that file. Thus, you can change the security provisions to allow yourself access to it.

EXAMPLE

You have created a filenamed FDATA and you want to change its security provisions to allow write access to yourself only. There will be no default security provisions. Enter:

```
:ALTSEC FDATA;(W:CR)
```

To change the security provisions for a program file (FPROG) so that any group user can execute the program, but only account and group librarian users can read or write to the file, enter:

```
:ALTSEC FPROG;(X:GU;R,W:AL,GL)
```

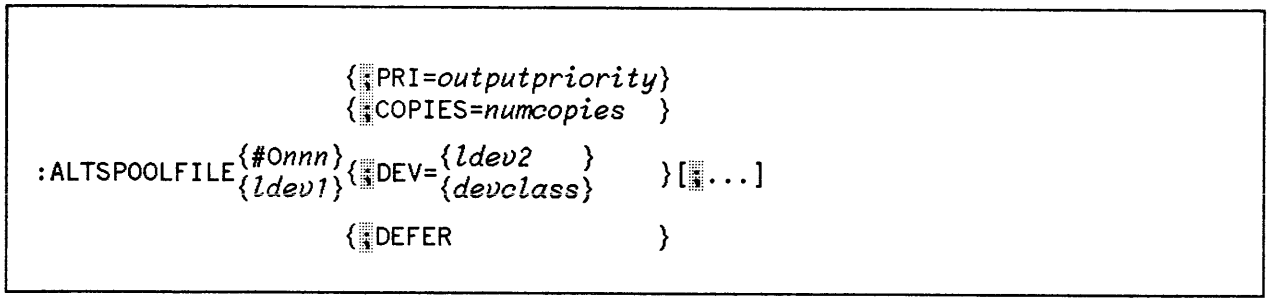
ADDITIONAL DISCUSSION

MPE V Intrinsics Reference Manual (32033-90007)
MPE File System Reference Manual (30000-90236)
MPE V System Operation and Resource Management Reference Manual (32033-90005)

:ALTSPOOLFILE

Alters the characteristics of an output spoolfile.

SYNTAX



PARAMETERS

#Onnn	The output devicefile identification of a spoolfile.
ldev1	The logical device number of the device where an ACTIVE spoolfile currently resides.
outputpriority	Specifies the output priority of the designated devicefile (0 = lowest; 14 = highest).
numcopies	Specifies the number of copies to be produced from the designated devicefile. Range is 1 through 127; default is 1.
ldev2 or *devclass*	Specifies the logical device number or device class name of the spoolfile's destination device. If ACTIVE, the file is returned to the READY state. It may immediately become ACTIVE on *ldev2* if all requirements are met.
DEFER	Immediately changes the output priority of an ACTIVE or READY spoolfile to 0. If ACTIVE, the file is returned to the READY state.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It is executable from the Console by users with OP or SM capability.*

* It may be distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

The Operator uses :ALTSPOOLFILE to change the printing priority of a spoolfile, to increase or decrease the number of copies produced, and/or to change the destination device or class.

When altering an ACTIVE spoolfile, first take the output device offline. This gives you time to enter the command and determine that the ACTIVE spoolfile is the file being printed. When the :ALTSPOOLFILE command has been sent to the spooler process, MPE returns the colon prompt (:). No change to the spoolfile will be made, however, until the output device is returned online.

You may alter the *outputpriority* or the *numcopies* of an ACTIVE spoolfile without interrupting the printing process. If you alter the device or defer the ACTIVE spoolfile with the DEFER parameter, the printer will stop immediately. In both of these cases, the entire file will be printed when printing resumes. Deferring a spoolfile lowers its output priority to zero, the lowest priority possible. To print a deferred spoolfile, you must raise its priority above the current OUTFENCE using the :ALTSPOOLFILE command.

If you intend to print a spoolfile on an HP 2680A Laser Page Printer, you may add an environment file to it before printing. Instructions for this procedure can be found in the MPE V System Utilities Reference Manual (32033-90008) in the discussion of the SPOOK subsystem >COPY command.

EXAMPLES

To defer the ACTIVE spoolfile (#086) on *ldev* 6 take device 6 offline, then enter:

```
:ALTSPOOLFILE #086;DEFER
```

or

```
:ALTSPOOLFILE 6;DEFER
```

To change the priority of deferred spoolfile #0123 from 0 to 3 enter:

```
:ALTSPOOLFILE #0123;PRI=3
```

:ALTUSER

Alters attributes currently defined for a user.

SYNTAX

```
:ALTUSER username
   [;PASS=[password]]
   [;CAP=[capabilitylist]]
   [;MAXPRI=[subqueuename]]
   [;LOCATTR=[localattribute]]
   [;HOME=[homegroupname]]
```

PARAMETERS

username The name assigned to the user within a logon account.

password The new password to be assigned to the user. When ";PASS= RETURN " is entered, any existing password is removed. If this parameter is omitted entirely, the password will remain unchanged.

capabilitylist The list of capabilities, separated by commas, permitted to this user. The capabilities assigned to the user cannot exceed those assigned to the account. Each capability is denoted by a two-letter mnemonic as follows:

System Manager	=	SM
Account Manager	=	AM
Account Librarian	=	AL
Group Librarian	=	GL
Diagnostician	=	DI
System Supervisor	=	OP
Network Administrator	=	NA
Node Manager	=	NM
Permanent Files	=	SF
Access to non-sharable I/O devices	=	ND
Use Volumes	=	UV
Create Volumes	=	CV
Use Communications Subsystem	=	CS
User Logging	=	LG
Process Handling	=	PH
Extra Data Segments	=	DS
Multiple RINs	=	MR
Privileged Mode	=	PM
Interactive Access	=	IA

Local Batch Access	=	BA
Programmatic Sessions	=	PS

Default is SF, ND, IA, and BA. Note that CV automatically gives the user UV capability.

subqueuename The name of the highest priority subqueue that may be requested by any process of any job/session initiated by the user. This parameter is specified as AS, BS, CS, DS, or ES, but cannot be greater than that specified with the :NEWACCT or :ALTACCT commands. The *subqueuename* defined for the user is checked against the *subqueuename* defined for the account at logon, and the lower priority of the two is used as the maximum priority restricting all processes of the job/session. Also, the priority requested by the user when he or she logs on is checked against the *subqueuename* defined for the user, and he or she is granted the lower of these two values. Default is CS.

CAUTION

Processes capable of executing in the AS or BS subqueues can deadlock the system. By assigning non-priority processes to these subqueues, you may prevent critical system processes from executing. Exercise extreme care when assigning processes to the AS or BS subqueue.

localattribute Defined at the installation site, this is an arbitrary double word bit map used to further classify users. When it is not part of standard MPE security provisions, programmers may define it (through the WHO intrinsic) to enhance the security of their own programs. The bit map for the user local attributes must be a subset of the bit map for the account local attributes. The :ALTUSER command checks the local attributes of the user with those of the account. Default is double word 0 (null).

homegroupname The name of an existing group to be assigned as the home group for this user. The first user established when an account is created will by default have PUB assigned as the *homegroup*. Subsequent new users will by default have no *homegroup* assigned. If no *homegroup* is assigned, the user must always specify an existing group when logging on.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have Account Manager (AM) capability to use this command.

OPERATION

The :ALTUSER command allows the Account Manager to change the password, capabilities, processing subqueue, security checking, and home group currently defined for a user. More than one of these attributes may be changed at a time, by entering multiple keyword parameters on a single command line, using the ; delimiter.

To change an attribute, enter the keyword and its new value. If you enter the keyword and no corresponding parameter, the default value is assigned. When an entire keyword parameter group is omitted from the :ALTUSER command, the corresponding value for the user remains unchanged. When a keyword is included, but the corresponding parameter is omitted (as in ;PASS=), a default value is assigned as follows:

PARAMETER	DEFAULT VALUES
password	Null (no password)
capabilitylist	SF , ND , IA , and BA (provided these capabilities have been specified for the account)
subqueuename	CS
localattribute	Null (double word 0)
homegroupname	The first user established when the account is created has *PUB* assigned as home. Subsequent users have no group assigned as home. If a user has no home group assigned, an existing group must be specified when initiating a job or a session.

When a parameter is modified with the :ALTUSER command, it is immediately registered in the directory. However, it will not affect users who are currently logged on to the system. They will be affected the next time they logon to the same user name and account. For this reason, you should warn users in advance of the intended changes.

You should avoid changing the *capabilitylist* or *homegroupname* of the user MANAGER.SYS. SM capability cannot be taken away from MANAGER.SYS.

If more than one user within an account is assigned AM capability, the capability can be removed from all but one (the last) user assigned it.

EXAMPLES

Suppose an account's capabilities are AM , AL , GL , SF , ND , PH , DS , MR , IA , and BA . To change the *capabilitylist* of the user JONES from IA , BA , SF , PH , DS to include Multiple RIN capability (MR), enter:

```
:ALTUSER JONES;CAP=IA,BA,SF,PH,DS,MR
```

To alter two attributes, *password* and *subqueuename*, for user JONES enter:

```
:ALTUSER JONES;PASS=JJ;MAXPRI=DS
```

:ALTVSET

Modifies volume set definitions.

SYNTAX

```
:ALTVSET vsname
    [;ADDCLASS=vcname:vname[,vname[,...]]]
    [;EXPANDCLASS=vcname:vname[,vname[,...]]]
    [;EXPANDSET=vname:type[,vname:type[,...]]]
```

PARAMETERS

vsname — The existing volume set name. MPE will implicitly reference *vsname* as *vsname.groupname.acctname*, where *groupname* and *acctname* are the logon group and account.

vcname:vname — The volume class name, consisting of up to eight alphanumeric characters, beginning with an alphabetic character. Each volume class definition consists of a volume class name and an associated list of volume names, *vname*. The list of volume names must be a subset of the volumes composing the volume set. One of the volume names must be that of the volume set's master volume.

ADDCLASS — Directs MPE to add a volume class and an associated list of volume names to an existing volume set. The list of volume names must be a subset of the volumes comprising the volume set. One of the volume names must be that of the volume set's master volume.

EXPANDCLASS — Directs MPE to add the names of new members in the volume class to an existing volume set. These new members can only be members of the parent volume set.

EXPANDSET — Directs MPE to add the names of new members in the volume set and types of devices required to accommodate the new members to an existing volume set. Total members in a volume set may not exceed eight. The type must be specified with each volume member name to define the type of disc drive. Types of disc drives that may be specified are:

HP 7902	HP 7920
HP 7905	HP 7925
HP 7906	HP 7933
HP 7911	HP 7935
HP 7912	HP 7945
HP 7914	HP 9895

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have Create Volume (CV) capability to use this command.

OPERATION

The :ALTVSET command is used to modify volume set definitions. Volume class names may be added, and the number of member volumes may be increased. The overall size of the volume set, however, cannot be decreased by deleting member volumes.

EXAMPLE

To add member volumes MEM3 and MEM4 to the existing volume set USERSET , enter:

```
:ALTVSET USERSET;EXPANDSET=MEM3:HP7920,MEM4:HP7920
```

ADDITIONAL DISCUSSION

Refer to the discussion of "PRIVATE VOLUMES" in the MPE V System Operation and Resource Management Reference Manual (32033-90005).

:ASSOCIATE

Gives a user Operator control of a device.

SYNTAX

```
:ASSOCIATE devclass
```

PARAMETERS

devclass The name of a logical device class configured during SYSDUMP.

USE

This command may be issued from a Session, Program, or in BREAK. It may not be used from a Job. It is not Breakable.

OPERATION

This command links a device class, such as LP, to an individual user on the system. The user may then execute any valid Operator command for a device in the device class and will receive the status messages for the devices in that device class on $STDLIST. For example, a remote printer may be associated with a terminal, so that messages concerning the printer will go to the terminal, not the System Console.

Before a user can be associated, the System Manager must run a utility program (the version of ASOCTBLE.PUB.SYS that matches your operating system) in order to create a device class/user association table. This table defines which users may be associated with which device classes. At any given time only one user may be associated with a given device class. If the device belongs in several device classes only one of those device classes may be associated.

Certain commands are only on G.00.00 and later releases of MPE. These commands are designated by G.00.00. The Operator commands which are made available to users through the :ASSOCIATE command are:

```
:ABORTIO                    :OPENQ (G.00.00)
:ACCEPT                     :MPLINE
:ALTSPOOLFILE               :REFUSE
:DELETESPOOLFILE            :REPLY
:DISCRPS (G.00.00)          :RESUMESPOOL
:DOWN                       :SHOWCOM
:DOWNLOAD                   :SHUTQ (G.00.00)
:FOREIGN                    :STARTSPOOL
:GIVE                       :STOPSPOOL
:HEADON                     :SUSPENDSPOOL
:HEADOFF                    :TAKE
                            :UP
```

Both the System Supervisor and the user may :DISASSOCIATE a user from a device. In addition, a user implicitly disassociates a device when logging off.

EXAMPLE

To be the controller of the device class "TAPE", enter:

```
:ASSOCIATE TAPE
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)

:BASIC

Interprets a BASIC program. BASIC is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:BASIC [commandfile][,[inputfile][,listfile]]
```

PARAMETERS

commandfile — Actual file designator of the sourcefile or device from which BASIC commands and statements are input. This can be any ASCII input file. Formal file designator is BASCOM. Default is $STDINX.

inputfile — Actual file designator of the file containing data input for a BASIC program. This can be any ASCII input file. Formal file designator is BASIN. Default is $STDINX.

listfile — Actual file designator of the destination file for the program listing and output. This can be any ASCII output file. Formal file designator is BASLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (BASCOM, BASIN, and BASLIST) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the :FILE command.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

The :BASIC command is generally used for online programming in BASIC, but it can also be used to interpret BASIC programs submitted in batch mode. In batch mode, the BASIC >EOD command is required after any data following the BASIC >RUN command, or after the >RUN command itself if there is no data.

EXAMPLES

To enter commands and data from your standard input device, with program listing and output transmitted to the standard output device, enter:

```
:BASIC
```

You may also submit commands and data to the BASIC interpreter through input files that you have stored on disc. Files created using the Editor Subsystem must be kept with the UNN (unnumbered) option of the Editor's Subsystem KEEP command. In this example, BASIC interpreter commands and statements are submitted from the command file MYCOMDS. The data that the program uses is stored in the input file MYDATA. The program listing and output are written to the file MYLIST.

```
:BASIC MYCOMDS,MYDATA,MYLIST
```

ADDITIONAL DISCUSSION

BASIC/3000 Interpreter Manual (32103-90001)
MPE Segmenter Reference Manual (30000-90011)

:BASICGO

Compiles, prepares, and executes a BASIC program. BASIC is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:BASICGO  [commandfile][,listfile]
```

PARAMETERS

commandfile — Actual file designator of the input file from which the BASIC compiler commands are read. This can be any ASCII input file. Formal file designator is BSCTEXT. Default is $STDINX.

listfile — Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is BSCLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (BSCTEXT, and BSCLIST) cannot be backreferenced as actual file designators in the command parameter list. Refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command for further information.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

This command compiles, prepares, and executes a program from a "fastsave" file created by the BASIC interpreter. This enables the program to run faster than it would if it were executed by the interpreter.

To save the program after it is written, use the BASIC interpreter command SAVE *filename*,FAST. The program then can be compiled, prepared, and executed with the :BASICGO command. You must specify the FAST option to compile the program.

EXAMPLE

To compile, prepare, and execute the BASIC program MYPROG, enter:

```
:BASICGO                    ** Calls the BASIC compiler. **
$CONTROL USLINIT            ** Initializes USL. **
$COMPILE MYPROG             ** Compiles program MYPROG. **
$EXIT                       ** Exits from compiler. **
```

ADDITIONAL DISCUSSION

BASIC Compiler Reference Manual (32103-90001)
MPE Segmenter Reference Manual (30000-90011)

:BASICOMP

Compiles a BASIC program. BASIC is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:BASICOMP [commandfile][,[uslfile][,listfile]]
```

PARAMETERS

commandfile Actual file designator of the input file from which the compiler commands are read. This can be any ASCII input file. Formal file designator is BSCTEXT. Default is $STDINX.

uslfile Actual file designator of the User Subprogram Library (USL) file to which the object code is written, which can be any binary output file with file code of USL or 1024. Its formal file designator is BSCUSL. If the *uslfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS. If entered, this parameter specifies that the file was created in one of four ways:

- By using the :SAVE command to save the default USL file $OLDPASS, created by a previous compilation.
- By building the USL with the Segmenter command -BUILDUSL (refer to the MPE Segmenter Reference Manual (30000-90011)).
- By creating a new USL file with the MPE :BUILD command and a file code of USL or 1024.
- By specifying a nonexistent *uslfile* parameter, thereby creating a permanent file of the correct size and type.

listfile Actual file designator of the file on which the program listing is written. This can be any ASCII output file. Formal designator is BSCLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (BSCTEXT, BSCUSL, and BSCLIST) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session, or Job. It may not be used from a Program, or in BREAK. It is Breakable (suspends execution).

OPERATION

The :BASICOMP command compiles a program from a "fastsave" file generated by the BASIC interpreter. If you do not specify a USL file, the BASIC compiler stores the object code in the default system defined temporary file $OLDPASS, as shown in the second example, below. You may, however, build a USL file in the permanent file domain, then direct the BASIC compiler to store the object code in this file by naming the USL file in the :BASICOMP command line. The first example, below, does this.

EXAMPLE

To compile the BASIC program MYPROG onto the USL named OBJECT, enter:

```
:BUILD OBJECT;CODE=USL      ** Builds USL file. **
:BASICOMP,OBJECT            ** Calls BASIC compiler; specifies USL named OBJECT. **
$CONTROL USLINIT            ** Initializes USL. **
$COMPILE MYPROG             ** Compiles fastsave program named MYPROG. **
$EXIT                       ** Exits from compiler. **
```

If you do not choose to build a USL file, the :BASICOMP command compiles your program, storing the object code in the default USL file $OLDPASS.

```
:BASICOMP                   ** Runs BASIC compiler, accepting commands from $STDINX, and
                               specifying $OLDPASS the USL output and $STDLIST for listing
                               output.**
$COMPILE MYRUN              ** Compiles from fastsave filenamed MYRUN into a USL named
                               $OLDPASS.**
$EXIT                       ** Exits from BASIC compiler. **
```

If you now want to run your program, use the :PREPRUN command:

```
:PREPRUN $OLDPASS           ** Prepares and runs program. **
```

ADDITIONAL DISCUSSION

BASIC Compiler Reference Manual (32103-90001)
MPE Segmenter Reference Manual (30000-90011)

:BASICPREP

Compiles and prepares a BASIC program. BASIC is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:BASICPREP [commandfile][,[progfile][,listfile]]
```

PARAMETERS

commandfile Actual file designator of the input file from which the compiler commands are read. This can be any ASCII file. Formal file designator is BSCTEXT. Default is $STDINX.

progfile Actual file designator of the program file on which the prepared program segments are written. You will get the best results by omitting the *progfile* parameter. When *progfile* is omitted, the MPE Segmenter creates the program file, which will reside in the temporary file domain as $OLDPASS. If you do create your own program file, you must do so in one of two ways:

- Using the MPE :BUILD command, and specifying a file code of 1029 or PROG and a *numextents* value of 1. This file is then used by the :PREP command.
- Specifying a nonexistent file in the *progfile* parameter, in which case a temporary job file of the correct size and type is created.

listfile Actual file designator of the file to which the listing is written. This can be any ASCII output file. Formal file designator is BSCLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (BSCTEXT, and BSCLIST) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit:FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

The :BASICPREP command compiles and prepares a program for execution from a "fastsave" file generated by the BASIC interpreter. If you omit the *progfile* parameter, the prepared program segments are stored in the system defined temporary file $OLDPASS. If you want to save the prepared program in a file other than $OLDPASS, you may either create a file and specify its filename on the :BASICPREP command line, or specify a non-existent *progfile*.

A program that you have compiled and prepared with the :BASICPREP command may be executed with the MPE :RUN command.

EXAMPLE

To compile and prepare a program named MYPROG from the BASIC fastsave filenamed MYCOMDS, with listing directed to the standard list device, enter:

```
:BASICPREP,MYCOMDS
```

The file MYPROG is an ASCII file that contains the following BASIC compiler commands:

```
$CONTROL USLINIT,SOURCE        ** Initializes USL and lists program. **
$COMPILE MYPROG                ** Compiles fastsave program. **
$EXIT                          ** Exits from compiler. **
```

ADDITIONAL DISCUSSION

BASIC Compiler Reference Manual (32103-90001)

:BBASIC

Interprets a HP Business BASIC program. HP Business BASIC is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:BBASIC [commandfile][,[inputfile][,listfile]]
```

PARAMETERS

commandfile Actual file designator of the sourcefile or device from which BBASIC commands and statements are input. This can be any ASCII input file. Formal file designator is BASCOM. Default is $STDINX.

inputfile Actual file designator of the file containing data input for a BBASIC program. This can be any ASCII input file. Formal file designator is BASIN. Default is $STDINX.

outfile Actual file designator of the destination file for the program listing and output. This can be any ASCII output file. Formal file designator is BASOUT. Default is $STDLIST.

NOTE

The formal file designators used in this command (BASCOM, BASIN, and BASOUT) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

The :BBASIC command is generally used for online programming in BASIC, but it can also be used to interpret BASIC programs submitted in batch mode. In batch mode, the BASIC >EOD command is required after any data following the BASIC >RUN command, or after the >RUN command itself if there is no data. Business BASIC has its own online HELP facility.

EXAMPLE

To enter commands and data from your standard input device, with program listing and output transmitted to the standard output device, use:

```
:BBASIC
```

You may also submit commands and data to the BBASIC interpreter through input files that you have stored on disc. Files created using the Editor must be kept with the UNN (unnumbered) option of the Editor's KEEP command. In this example, BASIC interpreter commands and statements are submitted from the command file MYCOMDS. The data that the program uses is stored in the input file MYDATA. The program listing and output are written to the file MYLIST.

```
:BBASIC MYCOMDS,MYDATA,MYLIST
```

ADDITIONAL DISCUSSION

HP Business BASIC Reference Manual (32115-90001)
MPE Segmenter Reference Manual (30000-90011)

:BBASICOMP

Compiles an HP Business BASIC program. HP Business BASIC is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:BBASICOMP infile[;[uslfile][;listfile]]
```

PARAMETERS

infile

Actual file designator of the BSAVE file containing the BBASIC program to be compiled. This can be any ASCII input file. Formal file designator is BBCIN.

uslfile

Actual file designator of the User Subprogram Library (USL) file on which the object program is written, which can be any binary output file with file code of USL or 1024. Its formal file designator is BBSCUSL. If the *uslfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS. If entered, this parameter specifies that the file was created in one of four ways:

- By using the :SAVE command to save, the default USL file $OLDPASS created by a previous compilation.
- By building the USL with the Segmenter command -BUILDUSL. Refer to the MPE Segmenter Reference Manual (30000-90011).
- By creating a new USL file with the MPE :BUILD command and specifying a file code of USL or 1024.
- By specifying a nonexistent *uslfile* parameter, thereby creating a permanent file of the correct size and type.

listfile

Actual file designator of the file on which the program listing is written. This can be any ASCII output file. Formal designator is BBCLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (BBCIN, BBCUSL, and BBCLIST) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session, or Job. It may not be used from a Program, or in BREAK. It is Breakable (suspends execution).

OPERATION

The :BBASICOMP command compiles a program from a BSAVE file generated by the BBASIC interpreter. This enables the program to run faster than it would if it were left in the form generated by the interpreter.

A BSAVE program file can be created from within the BBASIC interpreter after it is written, by using the BBASIC interpreter command >SAVE *filename*. The program may be compiled with the :BBASICOMP command, then prepared with the MPE command :PREP, and executed with the MPE command :RUN.

EXAMPLES

To compile the BBASIC program MYPROG onto the USL named OBJECT, enter:

```
:BBASICOMP MYPROG,OBJECT
```

If you do not choose to build a USL file, the :BASICOMP command compiles your program, storing the object code in the default USL file $OLDPASS.

```
:BBASICOMP MYPROG        ** Runs BASIC compiler and accepts MYPROG as a BSAVE program file.
                            $OLDPASS is the default USL, and $STDLIST is the default output
                            listing.**
```

If you now want to run your program, use the :PREPRUN command:

```
:PREPRUN $OLDPASS        ** Prepares and runs program. **
```

ADDITIONAL DISCUSSION

HP Business BASIC Reference Manual (32115-90001)
Basic Compiler Reference Manual (32103-90001)
MPE Segmenter Reference Manual (30000-90011)

:BBASICGO

Compiles, prepares, and executes an HP Business BASIC program. HP Business BASIC is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:BBASICGO infile[,listfile]
```

PARAMETERS

infile Actual file designator of the BSAVE file containing the BBASIC program to be compiled. Formal file designator is BBCIN.

listfile Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is BBCLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (BBCIN, and BBCLIST) cannot be backreferenced as actual file designators in the command parameter list. Refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command in this section.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

This command compiles, prepares, and executes a program from a BSAVE file created by the BBASIC interpreter. This enables the program to run faster than it would if it were executed by the interpreter.

You may create a BSAVE program file within the BBASIC interpreter after it is saved by using the BBASIC interpreter command >SAVE *filename*. The program then can be compiled, prepared, and executed with the :BBASICGO command.

EXAMPLE

To compile, prepare, and execute the BBASIC program MYPROG and send the listing to the disc file LISTFL, enter:

```
:BASICGO MYPROG,LISTFL          ** Compiles, prepares, and runs the program MYPROG. **
```

ADDITIONAL DISCUSSION

HP Business BASIC Reference Manual (32115-90001)
Basic Compiler Reference Manual (32103-90001)
MPE Segmenter Reference Manual (30000-90011)

:BBASICPREP

Compiles and prepares a HP Business BASIC program. HP Business BASIC is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:BBASICPREP infile[,[progfile][,listfile]]
```

PARAMETERS

infile Actual file designator of the BSAVE file containing the BBASIC program to be compiled. Formal file designator is BBCIN.

progfile Actual file designator of the program file on which the prepared program segments are written. You will get the best results by omitting the *progfile* parameter. When *progfile* is omitted, the MPE Segmenter creates the program file, which will reside in the temporary file domain as $OLDPASS. If you do create your own program file, you must do so in one of two ways:

- Using the MPE :BUILD command, and specifying a file code of 1029 or PROG and a *numextents* value of 1. This file is then used by the :PREP command.
- Specifying a nonexistent file in the *progfile* parameter, in which case a temporary job file of the correct size and type is created.

listfile Actual file designator of the file on which program the listing is written. This can be any ASCII output file. Formal file designator is BBCLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (BBCIN, and BBCLIST) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit:FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

The :BBASICPREP command compiles and prepares a program from a BSAVE file generated by the BBASIC interpreter. If you omit the *progfile* parameter, the prepared program segments are stored in the system defined temporary file $OLDPASS. If you want to save the prepared program in a file other than $OLDPASS, you may either create a file and specify its filename on the :BBASICPREP command line, or specify a non-existent *progfile.*

A BSAVE program file can be created from within the BBASIC interpreter after it is written, by using the BBASIC interpreter command >SAVE *filename.* The program may be compiled with the :BBASICOMP command, then prepared with the MPE command PREP, and executed with the MPE command :PREP, and executed with the MPE command :RUN.

EXAMPLE

To compile and prepare a program named MYPROG from the BBASIC BSAVE filenamed MYCOMDS, and send the listing to the standard list device, enter:

```
:BBASICPREP MYCOMDS,MYPROG
```

ADDITIONAL DISCUSSION

HP Business BASIC Reference Manual (32115-90001)
Basic Compiler Reference Manual (32103-90001)
MPE Segmenter Reference Manual (30000-90011)

:BREAKJOB

Suspends an executing job.

SYNTAX

```
:BREAKJOB #Jnnn
```

PARAMETERS

#Jnnn	A job number.

USE

This command may be issued from a Session, Job, or Program, and in BREAK. It is not Breakable. It is executable only from the Console.*

* Unless distributed to users with the :ALLOW command, or if :JOBSECURITY is set LOW.

OPERATION

The Operator can use the :BREAKJOB command to suspend any executing job, including spooled and streamed jobs. A job using a critical system resource will not be suspended until it releases the resource.

When you issue :BREAKJOB for a job that controls a nonsharable device, a console message is displayed listing the device(s) that the job controls. (As many as ten devices may be listed.) You may then decide whether the job should be allowed to run until it releases the device(s), or whether it should be aborted.

All commands that normally affect executing jobs, such as :ABORTJOB, will operate on suspended jobs. The :SHOWJOB command, which lists all jobs, will display "SUSP" for those in the suspended state. To list suspended jobs only, enter :SHOWJOB SUSP.

EXAMPLES

To suspend job number 68, enter:

```
:BREAKJOB #J68
```

To display suspended jobs, enter:

```
:SHOWJOB SUSP
JOBNUM  STATE  INPRI  JIN  JLIST  INTRODUCED  JOB  NAME
#68     SUSP          105    LP   WED. 7:56AM TEST,USER.ACCOUNT
```

Creates and immediately allocates a new empty file on disc.

SYNTAX

```
:BUILD filereference
    [;REC=[recsize][,[blockfactor][,[{F}][,{BINARY}]]]]
                                     {U}  {ASCII }
                                     {V}

    [{;CCTL  }]
    {;NOCCTL}

    [;TEMP]

    [;DEV=[[dsdevice]#][device]]

    [;CODE=filecode]

    [;DISC=[numrec][,[numextents][,[initialloc]]]]

    [{;RIO  }]
    {;NORIO}

     {;STD}
    [{;MSG}]
     {;CIR}
```

PARAMETERS

filereference — Actual name of the file to be created, in the following format:

filename[/*lockword*][.*groupname*[.*acctname*]]

Names must contain from one to eight alphanumeric characters, beginning with an alphabetic character. If *acctname* is specified, it must be that of your logon account (you cannot create a file in another account). The default *groupname* and *acctname* are the logon group and account.

recsize — The record size of a new file. A positive number specifies the size in words while a negative number specifies the size in bytes. For fixed length files, *recsize* is the logical record size. For undefined length files, *recsize* is the maximum record size. For variable length files using a blocking factor of 1, *recsize* is the maximum logical record size. If *blockfactor* is a value other than 1, *recsize* is used to calculate the maximum logical and physical record size.

Since records always begin on word boundaries, the record size is rounded up to the nearest word boundary to calculate block size. If the file format

is binary or variable length ASCII, odd byte lengths are rounded up and the extra byte is not available for data. For example, in a fixed length ASCII file with *recsize* equal to 11 bytes, only 11 bytes will be available for data in each logical record. To determine the block size, *recsize* is rounded up to 12 bytes. If the file format is binary, all 12 bytes, or 6 words, are available to store data.

The default record size is the configured physical record width of the associated device. Disc files will have a record size of 128 works unless you specify a device other than DISC using the ;DEV= parameter.

blockfactor — The number of logical records per physical block in a new file. The default is calculated by dividing the specified *recsize* into the configured block size; this value is rounded downward to an integer that is never less than 1. For variable length record files, *blockfactor* and *recsize* are used to calculate the maximum logical and physical record size. The *blockfactor* is then set to 1. For files containing undefined length records, the *blockfactor* is ignored. The maximum size of *blockfactor* is 255.

F, U, or V — Defines the length of the records of the file. A file may contain fixed length records (F), undefined length records (U), or variable length records (V). For disc files, the default is F.

BINARY or ASCII — Indicates the type of records the file contains. BINARY indicates binary coded records and is the default. ASCII indicates ASCII coded records.

CCTL or NOCCTL — Indicates whether or not carriage control characters are supplied along with data written to an ASCII file. CCTL indicates carriage control characters accompany the data; NOCCTL indicates carriage control characters are not specified. The default is NOCCTL.

TEMP — Indicates that the file will be created as a temporary jobfile and will be saved in the job/session temporary file domain when closed. Refer to the MPE File System Reference Manual (30000-90236) for more information on file domains. The default is that a permanent file is created.

dsdevice — The device class name or logical device number used to open a communications link to a remote computer that contains the source file. The default is the local system, or the computer on which the transfer request originates. A '#' symbol is a delimiter between the filename of the remote computer and the remote devicefilename.

device — Specifies either the *devclass* or *ldev* on which the file is to reside. A device class name, *devclass*, consists of up to eight alphanumeric characters beginning with an alphabetic character, such as DISC or PVDISC1. When you specify *devclass*, the file is allocated to any available device in that class. If you are opening a file destined for a Private Volume, you must specify a device class which includes the drives upon which the home volume set is mounted. The file is then allocated to any of the home volume set's volumes that fall within that device class.

The logical device number (*ldev*) consists of a one- to three-digit number specifying a particular device. Default is the device class name DISC.

filecode

A code indicating a specially formatted file. This code is recorded in the file label and is available to processes accessing the file through the FFILEINFO or FGETINFO intrinsic. Although any user can specify a positive integer ranging from 0 to 32767 or mnemonic name for this parameter, certain reserved integers and mnemonics have particular system defined meanings, as follows:

Integer	Mnemonic	Meaning
1024	USL	User Subprogram Library
1025	BASD	BASIC Data
1026	BASP	BASIC Program
1027	BASFP	BASIC Fast Program
1028	RL	Relocatable library
1029	PROG	Program file
1031	SL	Segmented Library
1035	VFORM	View Form file
1036	VFAST	View Fast Forms file
1037	VREF	View Reformat file
1040	XLSAV	Cross Loader ASCII file (SAVE)
1041	XLBIN	Cross Loader Relocated Binary file
1042	XLDSP	Cross Loader ASCII file (DISPLAY)
1050	EDITQ	Edit Quick file
1051	EDTCQ	Edit KEEPQ file (COBOL)
1052	EDTCT	Edit TEXT file (COBOL)
1054	TDPDT	TDP Diary file
1055	TDPQM	TDP Proof Marked QMARKED
1056	TDPP	TDP Proof Marked non-COBOL file
1057	TDPCP	TDP Proof Marked COBOL file
1058	TDPQ	TDP Workfile
1059	TDPXQ	TDP Workfile (COBOL)
1060	RJEPN	RJE Punch file
1070	QPROC	QUERY Procedure file
1080	KSAMK	KSAM Key file
1083	GRAPH	GRAPH Specification file
1084	SD	Self-describing file
1090	LOG	User Logging logfile
1100	WDOC	HPWORD Document
1101	WDICT	HPWORD Hyphenation dictionary
1102	WCONF	HPWORD Configuration file
1103	W2601	HPWORD Attended Printer Environment
1110	PCELL	IFS/3000 Character Cell file
1111	PFORM	IFS/3000 Form file
1112	PENV	IFS/3000 Environment file
1113	PCCMP	IFS/3000 Compiled Character Cell file
1114	RASTR	Graphics Image in RASTR Format
1130	OPTLF	OPT/3000 logfile
1131	TEPES	TEPE/3000 Script file
1132	TEPEL	TEPE/3000 logfile
1133	SAMPL	APS/3000 logfile

Integer	Mnemonic	Meaning
1139	MPEDL	MPEDCP/DRP logfile
1140	TSR	HPToolset Root file
1141	TSD	HPToolset Data file
1145	DRAW	Drawing File for HPDRAW
1146	FIG	Figure File for HPDRAW
1147	FONT	Reserved
1148	COLOR	Reserved
1149	D48	Reserved
1152	SLATE	Compressed SLATE file
1153	SLATW	Expanded SLATE workfile
1156	DSTOR	Store file for RAPID/3000 Utility DICTDBU
1157	TCODE	Code file for Transact/3000 Compiler
1158	RCODE	Code file for Report/3000 Compiler
1159	ICODE	Code file for Inform/3000 Compiler
1166	MDIST	HPDESK Distribution list
1167	MTEXT	HPDESK Text
1168	MARPA	ARPA Message file
1169	MARPD	ARPA Distribution List
1170	MCMND	HPDESK Abbreviated Commands file
1171	MFRTM	Reserved
1172		Reserved
1173	MEFT	Reserved
1174	MCRPT	Reserved
1175	MSERL	Reserved
1176	UCSF	Reserved
1177	TTYPE	Term Type file
1178	TVFC	Term Vertical Format Control file
1192	NCONF	Network Configuration file
1193	NTRAC	Network Trace file
1194	NLOG	Network logfile
1195	MIDAS	Reserved
1211	ANODE	Reserved
1212	INODE	Reserved
1213	INVRT	Reserved
1214	EXCEP	Reserved
1215	TAXON	Reserved
1216	QUERF	Reserved
1217	DOCDR	Reserved
1226	VC	VC file
1227	DIF	DIF file
1228	LANGD	Language Definition file
1229	CHARD	Character Set Definition file
1230	MGCAT	Formatted Application Message Catalog
1236	BMAP	Reserved
1242	BDATA	Basic Data file
1243	BFORM	Basic Field Order File for VPLUS
1244	BSAVE	Basic Saved Program file
1245	BCNFG	Configuration File for Default Option Basic Program

Integer	Mnemonic	Meaning
1258	PFSTA	Pathflow STATIC file
1259	PFDYN	Pathflow DYNAMIC file
1270	RTDCA	Revisable Form DCA Document (DCA = Document Content Architecture)
1271	FFDCA	Final Form DCA Document (DCA = Document Content Architecture)
1272	DIU	Document Interchange Unit file
1273	PDOC	HPWORD/150 Document
1401	CWPTX	Reserved
1421	MAP	HPMAP/3000 Map Specification file
1422	GAL	Reserved
1425	TTX	Reserved
3333		Reserved

Default is the unreserved file code of 0.

Using 1090 (LOG) as your designated *filecode* may not yield the number of records you specify in the ;DISC= parameter. Most files use the number of records specified in the ;DISC= parameter as the maximum limit; user logging uses this specified number as a minimum.

numrec

The maximum number of logical records in a new file. The theoretical maximum for fixed and undefined length records is 2,147,483,647. Since this maximum will be limited by block size, the actual maximum number of records per file is 2,097,120, which is based on 65535 sectors per extent and 32 extents. An approximate practical limit for *numrec* is 2,097,119 for files with variable and undefined length records, and 267,382,000 for fixed length files. The default is 1023.

NOTE

The file system uses these values to compute other characteristics of the file as well. Therefore, the values (or default values) specified on the :BUILD command may be valid within their respective fields, but may cause overflow errors in the computation of internally needed file specifications. Refer to the MPE File System Reference Manual (30000-90236) for a discussion on calculating file space.

numextents

Maximum number of disc extents. This is a value from 1 to 32. Default is 8.

initialloc

Number of extents to be initially allocated to the file at the time it is opened. This is a value from 1 to 32. Default is 1.

RIO or NORIO

RIO creates a relative I/O file, which is a special file access method primarily used by COBOLII programs. You can, however, access these files from programs written in any language. Specifying RIO implicitly changes the record length parameter to F, or fixed record length. The default, NORIO, creates a non-relative I/O file.

RIO and NORIO specifications affect only the physical characteristics of the file. If NOBUF is specified in the :FILE command, the file will be accessed in non-RIO mode; otherwise RIO access is used with RIO files. Special operations on RIO files, such as replicating a RIO file, uses NOBUF access. Refer to the MPE V Intrinsics Reference Manual (32033-90007) for a discussion of relative I/O.

STD, MSG, or CIR

Defines the type of file. The default, STD, is a standard MPE disc file. A MSG, or message, file allows communication between any set of processes in a first in, first out (FIFO) manner. Records are read from the start of the file and logically deleted and/or are appended to the end of the file. A CIR, or circular, file acts as a normal sequential file until full. When full, the first physical block is deleted when the next record is written, and remaining blocks are logically shifted to the front of the file. A circular file cannot be simultaneously accessed by readers and writers.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

This command builds a new file on disc and immediately allocates space for it. If it is an ASCII file, the file space is initialized to blanks. If it is a binary file, the file space is initialized to zeros.

Unless the TEMP parameter is specified, the file is saved in the permanent file domain. To create a permanent file, you must have SF (Save Files) capability and SAVE access to the group to which the new file will belong. You can only build a file belonging to your logon account.

If specified, the ";DEV=" parameter must be consistent with the group to which the new file will belong. If the group's home volume set is not mounted, :BUILD will implicitly generate a volume set mount request. Refer to the MPE File System Reference Manual (30000-90236) for more information on file domains.

The default characteristics of a file created with the :BUILD command are: fixed length records of 128 words each, a blocking factor of 1, binary formatted, permanent file, a record limit of 1023, and a maximum of 8 extents with 1 extent initially allocated. This is equivalent to entering:

:BUILD *filename*;REC=128,1,F,BINARY;DEV=DISC;DISC=1023,8,1

EXAMPLES

The following example creates a permanent disc filenamed WORKFILE, which can reside on any disc. WORKFILE has fixed length records of 80 bytes each. The records are blocked 3 records per block (which is the *blockfactor*), and are written in ASCII code. The file has a maximum capacity of 2000 records divided into 10 extents with 2 extents allocated immediately.

```
:BUILD WORKFILE;REC=-80,3,F,ASCII;DISC=2000,10,2
```

The second example uses the :BUILD command to create a permanent disc file, named VFILE, belonging to the logon (default) group that resides on the volume set specified by VCLASS1:

```
:BUILD VFILE;DISC=500,10,1;REC=-80;DEV=VCLASS1
                                  ↑
                    Name of existing volume class
```

The following example uses the CODE= parameter to create a logging file called NEWDATA:

```
:BUILD NEWDATA;DISC=3000,1,1;CODE=LOG
```

ADDITIONAL DISCUSSION

MPE File System Reference Manual (30000-90236)
MPE V Intrinsics Reference Manual (32033-90007)

:BYE

Ends an interactive session.

SYNTAX

```
:BYE
```

PARAMETERS

None.

USE

This command may be issued from a Session. It may not be used from a Job, Program, or in BREAK. It is not Breakable.

OPERATION

This command terminates a session and displays the CPU time used (in seconds), connect time (in minutes), the date and time, as follows:

```
CPU=48. CONNECT=35. FRI, MAY 4, 1985, 10:56 PM
```

If you enter a :HELLO command before logging off, MPE terminates your current session and immediately initiates a new one. If you are logged onto the computer with a telephone connection, and you hang up before terminating your session, MPE issues a :BYE command automatically.

EXAMPLE

To terminate a session, enter:

```
:BYE
CPU=1. CONNECT=5. FRI, MAY 4, 1985, 12:20 PM
```

:CACHECONTROL

Tunes the performance of disc caching. Disc caching software is not included in the MPE Fundamental Operating Software and must be purchased separately.

Syntax

```
:CACHECONTROL [SEQUENTIAL=sequentn][;RANDOM=randomn][;BLOCKWRITE={YES}]
                                                                 {NO }
```

PARAMETERS

sequentn — The number of sectors to be read from a sequential access disc file if the information is not found in the cache. The disc read will stop at the extent boundaries and fetch at least the number or sectors requested. The number of sectors must be greater than 1, but less than 96 (inclusive). Default is 96 sectors.

randomn — The number of sectors to be read from a random access disc file if the information is not found in the cache. The disc read stops at the extent boundaries and fetches at least the number of sectors requested. The number of the sectors must be greater than 1, but less than or equal to 96. Default is 16 sectors.

BLOCKONWRITE — Specifies whether MPE will block the process until the posting of cache buffers to the disc is complete. The BLOCKONWRITE must be equal to YES or NO. Default is NO.

USE

This command may be issued from a Session, Job, or Program. It may not be issued in BREAK. It is not Breakable. It requires System Supervisor (OP) or System Manager (SM) capability.

OPERATION

The :CACHECONTROL command tunes the performance of disc caching on your system. You must specify at least one keyword. Any combination of keywords can be used in a single command, but each can be used only once. They can be entered in any order.

When an I/O read request fails to find data currently in memory (a read "miss"), the requested portion of the disc must be fetched into memory. Disc caching uses the information about the file extent size and access type (random or sequential) in its fetch strategy to maximize the "hit" ratio; that is, subsequent read requests may be satisfied by a disc domain fetched as a result of a miss.

The size of the fetch is limited by the extent size of the file. Within this limit, the size of the disc domain fetched will equal the greater of the following:

- The requested size, or
- The largest integer multiple of the requested size that is smaller than the value specified by the *sequentn* or *randomn* parameters.

If the actual request is greater than the value for *sequentn* then caching will fetch the actual request size. If it is less than or equal to the value for *sequentn*, then caching will use the largest multiple of the requested size that is still less than the configured *sequentn/randomn* value. A high hit ratio on read requests will help maximize disc caching performance. A poor choice of values for either or both of the *sequentn* and *randomn* parameters can substantially decrease the hit ratio.

When disc caching is enabled on your system, data residing in the disc domain is transferred from the portion of memory reserved as the disc cache, and not from the disc itself. This way, I/Os complete at processor speed, rather than at disc speed.

When the I/O request is a write operation, physical posting to the disc will occur as a background task while the process continues. In this way, write requests appear completed to the user process before the disc has been updated. By specifying the BLOCKONWRITE=YES parameter, no user write requests will show complete until they are actually posted to the disc. This guarantees that the information is on disc when the request completes. However, the YES option it does eliminate one performance benefit that disc caching normally provides: write operations will complete at disc access speed, rather than at processor speed.

EXAMPLE

To set the fetch values for random access files to 24-sector reads, and the value for sequential access files to 48, enter:

```
:CACHECONTROL SEQUENTIAL=48,RANDOM=24
```

In this example the default value of BLOCKONWRITE, which is NO, will be used.

:CHANGELOG

Changes the user logging file without stopping or interrupting the logging process. This command is available for G.02.00 and later releases of MPE V.

SYNTAX

```
:CHANGELOG  logid [;DEV=device]
```

PARAMETERS

logid — Name of the currently active use logging process. This name may contain from one to eight alphnumeric characters, beginning with an alphabetic character.

device — Name of the device on which the new logging file is to be created. The device may be any one of the following: DISC, TAPE, SDISC, or CTAPE. Default is DISC.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It is available only to the creator of the *logid* and requires System Supervisor (OP) or User Logging (LG) capability.

OPERATION

This command permits the user to change the active logging file without stopping the logging process with the :LOG *logid*, STOP command. By specifying a device, you may switch the logging file from one device to another, regardless of the device on which the logging file was created. If you have enabled automatic logging with the :ALTLOG or :GETLOG command, however, the only device available for logging is the default, DISC.

If the :CHANGELOG command is valid, the system writes a changelog record to the end of the current logging file and closes the file. It then opens a new logging file whose characteristics are identical to those of the preceding file and makes the new file permanent. If the system is unable to open a new file of the same size, it will try to open a new file half the size of the old file. If the system determines that the half-size file is too small to continue logging, user logging terminates.

If the system opens a new logfile, it will immediately write a changelog record to the new file. The changelog record posted to the old logging file contains the fully qualified identifier of the new logging file. A corresponding changelog written to the new file will contain the fully qualified identifier of the old logging file. Changelog records will also contain the devicetype of the logging file to which the changelog refers.

The following messages will be displayed on the $STDLIST to confirm a successful change:

```
Log file for logid AAA has been changed from A001.PUB.SYS to A002.PUB.SYS
(ulogmsg 38)
```

If the new logging file is a serial file, a message will appear on the Console advising the operator to mount the new log file:

```
Mount new {tape/cartridge tape/serial disc} volume for logid (ulogmsg 40).
```

When a user logging file is full, the system terminates the logging process and displays an appropriate message.

However, by specifying the AUTO parameter in a :GETLOG or :ALTLOG command, you will enable an automatic :CHANGELOG, thereby freeing yourself of the need to issue the :CHANGELOG command manually. The :AUTO parameter is available for G.02.00 and later releases of MPE V. Refer to the :ALTLOG and :GETLOG commands in this section.

In order to use :CHANGELOG (manually or automatically), you must end the first user logging filename with the numeric characters "001" (e.g. *fname001*). This establishes a naming convention that works in conjunction with the fileset number to generate sequential filenames independently. New filenames will consist of the filename root (*fname*) plus the next sequential increment of the last three digits:

Current File	Next File
TEST001	TEST002
TEST002	TEST003
...	...
TEST998	TEST999
TEST999	TEST000

The logging process opens files, and automatically names them with the next sequential number, up to a maximum of 999. Thereafter, the numbering sequence will reset to 000 and begin incrementing all over again.

Automatic logging with the :CHANGELOG command is available only for disc files.

NOTE

The logging process specified by *logid* must be currently active. If the logging process is in another state, such as RECOVERING, STOP, INTIALIZING, or if the loggging process has another :CHANGELOG pending, the command will terminate in an error state. The :ALTLOG command permits changing the logfile for an inactive logging process. :ALTLOG, however, does not provide a way to link logfiles into a set.

ADDITIONAL DISCUSSION

Refer to Section III, "Optional Capability" of the MPE V Intrinsics Manual (32033-90007) and the commands :LISTLOG and :SHOWLOGSTATUS in this section.

:COBOL

Compiles a COBOL program on the COBOL 68 compiler. COBOL is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:COBOL  [textfile][,[uslfile][,]listfile][,[masterfile][,newfile]]]
```

PARAMETERS

textfile — Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is COBTEXT. Default is $STDIN.

uslfile — Actual file designator of the User Subprogram Library (USL) file to which the object program is written, which can be any binary output file with a file code of USL or 1024. Formal file designator is COBUSL. If the *uslfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS. If entered, this parameter indicates that the file was created in one of four ways:

- By saving, using the :SAVE command to save the default USL file $OLDPASS created during a previous compilation.
- By building the USL with the Segmenter command -BUILDUSL. Refer the MPE Segmenter Reference Manual (30000-90011).
- By creating a new USL file with the MPE :BUILD command and specifying a file code of USL or 1024.
- By specifying a nonexistent *uslfile* parameter, thereby creating a permanent file of the correct size and type.

listfile — Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is COBLIST. Default is $STDLIST.

masterfile — Actual file designator of the master file with which *textfile* is merged to produce a composite source. This can be any ASCII input file. Formal file designator is COBMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDIN if the *textfile* is not specified.

newfile — Actual file designator of the file created by the *textfile* and *masterfile*. This can be any ASCII output file. Formal file designator is COBNEW. Default is that no file is written.

NOTE

The formal file designators used in this command (COBTEXT, COBUSL, COBLIST, COBMAST, and COBNEW) cannot be backreferenced as literal file identifiers in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

The :COBOL command compiles a COBOL source program into a USL file on disc. If you do not specify a source textfile, MPE expects input from your standard input device. If you create the USL file prior to compilation, you must specify a file code of USL or 1024. If you do not specify a *listfile*, MPE sends the program listing to your standard list device.

EXAMPLE

To compile a COBOL program from a file named SOURCE into an object program stored in the USL file $OLDPASS sending the listing to your standard list device, enter:

```
:BUILD OBJECT;CODE=USL
;COBOL SOURCE,OBJECT
```

ADDITIONAL DISCUSSION

COBOL/3000 Reference Manual (32213-90001)
MPE Segmenter Reference Manual (30000-90011)

:COBOLGO

Compiles, prepares, and executes a COBOL program on the COBOL 68 compiler. COBOL is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:COBOLGO  [textfile][,[listfile][,[masterfile][,newfile]]]
```

PARAMETERS

textfile — Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is COBTEXT. Default is $STDIN.

listfile — Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal designator is COBLIST. Default is $STDLIST.

masterfile — Actual file designator of the master file with which *textfile* is merged to produce a composite source. This can be any ASCII input file. Formal file designator is COBMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified.

newfile — Actual file designator of the file created by the merger *textfile* and *masterfile*. This can be any ASCII output file. Formal file designator is COBNEW. Default is that no file is written.

NOTE

The formal file designators used in this command (COBTEXT, COBLIST, COBMAST, and COBNEW) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

The :COBOLGO command compiles, prepares, and executes a COBOL program. If you do not specify *textfile*, MPE expects input from your standard input device. If you do not specify *listfile*, MPE sends the listing to your standard list device.

The User Subprogram Library (USL) file created during compilation is a system defined temporary file $NEWPASS, which is passed directly to the Segmenter. The Segmenter writes the prepared program to $OLDPASS, which can then be executed.

EXAMPLE

To compile, prepare, and execute a COBOL program entered from your standard input device, with the program listing sent to your standard list device, enter:

```
:COBOLGO
```

To compile, prepare, and execute a COBOL program from the disc file TEXTFL and send the program listing to the disc file LISTFL, enter:

```
:COBOLGO TEXTFL,LISTFL
```

ADDITIONAL DISCUSSION

COBOL/3000 Reference Manual (32213-90001)
MPE Segmenter Reference Manual (30000-90011)

:COBOLPREP

Compiles and prepares a COBOL program on the COBOL 68 compiler. COBOL is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:COBOLPREP[textfile]][,[masterfile][,newfile]]]][,[progfile][,[listfile
```

PARAMETERS

textfile — Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is COBTEXT. Default is $STDIN.

progfile — Actual file designator of the program file to which the prepared program segments are written. You will get the best results by omitting the *progfile* parameter. When you omit *progfile*, the MPE Segmenter creates the program file which will reside in the temporary file domain as $OLDPASS. The formal file designator is COBPROG. If you wish to create your own program file, you must do so in one of two ways:

- By Using the MPE :BUILD command, and specifying a file code of 1029 or PROG, and a *numextents* value of 1. This file is then used by the :PREP command.
- Specifying a nonexistent file in the *progfile* parameter, thereby creating a job file of the correct size and type.

listfile — Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is COBLIST. Default is $STDLIST.

masterfile — Actual file designator of the master file with which *textfile* is merged to produce a composite source. This can be any ASCII input file. Formal file designator is COBMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified.

newfile — Actual file designator of the merged *textfile* and *masterfile*. This can be any ASCII output file. Formal file designator is COBNEW. Default is that no file is written.

NOTE

The formal file designators used in this command (COBTEXT, COBLIST, COBMAST, and COBNEW) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

The :COBOLPREP command compiles and prepares a COBOL program creating a program file on disc. If you do not specify *textfile*, MPE expects the source program input from your standard input device. If you do not specify *listfile*, MPE sends the output to your standard list device.

You will get the best results by allowing the Segmenter to create your program file. You can, however, create your own permanent program file with the :BUILD command. When you do, you must specify a file code of "PROG" or "1029" and a *numextents* parameter of 1. The following BUILD command creates such a file, which is then used by the :PREP command:

```
:BUILD PROGX;CODE=PROG;DISC=,1
```

The USL file created during compilation is a system defined temporary file $OLDPASS, which is passed directly to the Segmenter. The Segmenter also uses the file $OLDPASS. The prepared program segments are written to it, thus overwriting any existing temporary file of that name. Therefore, although it is recommended that you use the default for *progfile*, if you decide that it is important to save the USL file created during the compilation process, you must create your own, permanent program file, referencing it in the :COBOLPREP command line.

EXAMPLE

To compile and prepare a COBOL program entered through your current input device with the listing printed on the current list device, enter:

```
:COBOLPREP
```

To compile and prepare a COBOL source program entered from the *textfile* SFILE into the program file MYPROG, sending the listing to your standard list device, enter:

```
:COBOLPREP SFILE,MYPROG
```

ADDITIONAL DISCUSSION

COBOL/3000 Reference Manual (32213-90001)
MPE Segmenter Reference Manual (30000-90011)

:COBOLII

Compiles a COBOLII program on the COBOL '74 compiler. COBOLII is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:COBOLII [textfile][;[uslfile][,[;listfile][;[masterfile][;newfile]]]]
   [;INFO=quotedstring]
   [;WKSP=workspacename]
```

PARAMETERS

textfile — Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is COBTEXT. Default is $STDIN.

uslfile — Actual file designator of the User Subprogram Library (USL) on which the object program is written. This can be any binary output file with a file code of USL or 1024. Its formal file designator is COBUSL. If the *uslfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS. If the parameter is entered, it indicates that the file was created in one of four ways:

- By using the :SAVE command to save, the default USL file created during a previous compilation.
- By building the USL with the Segmenter command -BUILDUSL. (Refer to the MPE Segmenter Reference Manual (30000-90011).)
- By creating a new USL file with the MPE :BUILD command and specifying a file code of USL or 1024.
- By specifying a nonexistent *uslfile* parameter, thereby creating a permanent file of the correct size and type.

listfile — Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is COBLIST. Default is $STDLIST.

masterfile — Actual file designator of the master file with which *textfile* is merged to produce a composite source. This can be any ASCII input file. The formal designator is COBMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified.

newfile — Actual file designator of the merged *textfile* and *masterfile*. This can be any ASCII output file. Formal file designator is COBNEW. Default is that no file is written.

quotedstring — A sequence of ASCII characters bounded by a pair of single quotation marks (apostrophes) or by double quotation marks. You may use the delimiting character as part of the string so long as it appears twice. Any occurence of two single quotes in a row or two double quotes in a row, is considered part of the string, and, therefore, not the terminating delimiter.

INFO=*quotedstring* is used in the COBOLII programming language to pass compiler options to a program. These options will appear before the first line of source code in the text file. For detailed information, refer to the :RUN command in this section.

workspacename — Actual file designator of an HPToolset workspace used with HPToolset. The formal designator is COBWKSP.

NOTE

The formal file designators used in this command (COBTEXT, COBUSL, COBLIST, COBMAST, COBNEW, and COBWKSP) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or a Job, but not in BREAK of from a Program. It is Breakable (suspends execution).

OPERATION

The :COBOLII command compiles a COBOLII program into a USL file on disc. If you do not specify *textfile*, COBOLII expects the source text to be entered from your standard input device. If you do not specify a *listfile*, COBOLII sends the program listing to the current list device.

EXAMPLE

To compile a COBOLII program stored in the file SOURCE into an object program on the USL file OBJECT and send the listing to the disc file LISTFL, enter:

```
:BUILD OBJECT;CODE=USL
:COBOLII SOURCE,OBJECT,LISTFL
```

ADDITIONAL DISCUSSION

COBOL/II 3000 Reference Manual (32233-90001)
MPE Segmenter Reference Manual (30000-90011)

:COBOLIIGO

Compiles, prepares, and executes a COBOLII program on the COBOL '74 compiler. COBOLII is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:COBOLIIGO [textfile][,[listfile][,[masterfile][,newfile]]]
   [;INFO=quotedstring]
   [;WKSP=workspacename]
```

PARAMETERS

textfile — Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is COBTEXT. Default is $STDIN.

listfile — Actual file designator of the file on which the program listing is written. This can be any ASCII output file. Formal file designator is COBLIST. Default is $STDLIST.

masterfile — Actual file designator of the master file which is merged against *textfile* to produce a composite source. This can be any ASCII input file. Formal file designator is COBMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified.

newfile — Actual file designator of the merged *textfile* and the *masterfile*. This can be any ASCII output file. Formal file designator is COBNEW. Default is that no file is written.

quotedstring — A sequence of ASCII characters bounded by a pair of single quotation marks (apostrophes) or by double quotation marks. You may use the delimiting character as part of the string so long as it appears twice. Any occurence of two single quotes in a row, or two double quotes in a row, is considered part of the string, and, therefore, not the terminating delimiter.

INFO=*quotedstring* is used in the COBOLII programming language to pass compiler options to a program. These options will appear before the first line of source code in the text file. For detailed information, refer to the :RUN command in this section.

workspacename — This parameter is the actual file designator of an HPToolset workspace. The formal file designator created by the compiler is COBWKSP.

NOTE

The formal file designators used in this command (COBTEXT, COBLIST, COBMAST, COBNEW, and COBWKSP) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or Job, but not in BREAK or from a Program. It is Breakable (suspends execution).

OPERATION

The :COBOLIIGO command compiles, prepares, and executes a program using the COBOL '74 compiler. If you do not specify *textfile*, COBOLII expects the source program to be entered from your standard input device. If you do not specify *listfile*, COBOLII sends the output to your standard list device.

The USL file created during compilation is a system defined temporary file, $OLDPASS, which is passed directly to the Segmenter. The Segmenter purges the USL file and writes the prepared program to $OLDPASS if the *progfile* is omitted, which is then executed and may be executed repeatedly.

EXAMPLE

To compile, prepare, and execute a COBOLII program entered from your standard input device and send the program listing to your standard list device, enter:

```
:COBOLIIGO
```

To compile, prepare and execute a COBOLII program from the disc file TEXTFL and send the program listing to the disc file LISTFL, enter:

```
:COBOLIIGO TEXTFL,LISTFL
```

ADDITIONAL DISCUSSION

COBOL/II Reference Manual (32233-90001)
MPE Segmenter Reference Manual (30000-90011)

:COBOLIIPREP

Compiles and prepares a COBOLII program on the COBOL '74 compiler. COBOLII is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:COBOLIIPREP [textfile][;[progfile][;[listfile][;[masterfile][;newfile]]]]
   [;WKSP=workspacename]
   [;INFO=quotedstring]
```

PARAMETERS

textfile Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is COBTEXT. Default is $STDIN.

progfile Actual file designator of the program file to which the prepared program segments are written. You will get the best results by omitting the *progfile* parameter. When you omit *progfile*, the MPE Segmenter creates the program file, which will reside in the temporary file domain as $OLDPASS. Its formal file designator is COBPROG. If entered, this *progfile* indicates that a file was created in one of two ways:

- By using the MPE :BUILD command and specifying a file code of 1029 or PROG, and a *numextents* value of 1. This file is then used by the :PREP command.
- By specifying a nonexistent file in the *progfile* parameter. A temporary job file of the correct size and type is created.

listfile Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is COBLIST. Default is $STDLIST.

masterfile Actual file designator of the file which is merged against *textfile* to produce a composite source. This can be any ASCII input file. Formal file designator is COBMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDLIST if *textfile* is not specified.

newfile Actual file designator of the file created by merging *textfile* and *masterfile*. This can be any ASCII output file. Formal file designator is COBNEW. Default is that no file is written.

quotedstring A sequence of ASCII characters bounded by a pair of single quotation marks (apostrophes) or by double quotation marks. You may use the delimiting character as part of the string so long as it appears twice. Any occurence of two single quotes in a row, or two double quotes in a row, is considered part of the string, and, therefore, not the terminating delimiter.

INFO=*quotedstring* is used in the COBOLII programming language to pass compiler options to a program. These options will appear before the first line of source code in the text file. For detailed information, refer to the :RUN command in this section.

workspacename This parameter is the actual file designator of an HPToolset workspace used with HPToolset. The formal file designator created by the compiler is COBWKSP.

NOTE

The formal file designators used in this command (COBTEXT, COBLIST, COBMAST, COBNEW, and COBWKSP) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or a Job, but not it BREAK or from a Program. It is Breakable (suspends execution).

OPERATION

The :COBOLIIPREP command compiles and prepares a COBOLII program into a program file on disc. If you do not specify *textfile*, COBOLII expects your input from your standard input device. If you do not specify *listfile*, COBOLII sends the listing output to your current list device.

The USL file created during compilation is a system defined temporary file, $OLDPASS, which is passed directly to the Segmenter. The Segmenter overwrites the USL file and writes the prepared program to $OLDPASS if *progfile* is omitted, which can then be executed.

EXAMPLE

To compile and prepare a COBOLII program entered from your standard input device with the listing printed on the standard list device, enter:

```
:COBOLIIPREP
```

To compile and prepare a COBOLII source program input from the *textfile* SFILE into a program filenamed MYPROG, with the resulting listing sent to the current list device, enter:

```
:COBOLIIPREP SFILE,MYPROG
```

ADDITIONAL DISCUSSION

COBOL/II Reference Manual (32233-90001)
MPE Segmenter Reference Manual (30000-90011)

:COMMENT

Inserts a comment into the command stream.

SYNTAX

```
:COMMENT  [text]
```

PARAMETERS

text — Information composing the comment text. If the last nonblank character is an ampersand (&), comment text is continued onto the next line. Default is that a record containing only the string "COMMENT" is inserted in the command stream.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

The :COMMENT command allows you to include comments or notes in job listings produced on hard copy devices to create headings or explain the purpose of commands or logic used. After :COMMENT is entered, it can be followed by a message made up of any ASCII characters.

EXAMPLE

The following example of a job heading employs a comment:

```
!JOB MAC.TECHPUBS
!COMMENT -- THIS IS A SAMPLE JOB
!FORTGO MYPROG
!EOJ
```

:CONSOLE

Changes the System Console from its current device to another job-accepting terminal.

SYNTAX

```
:CONSOLE [ldev]
```

PARAMETERS

ldev	The logical device number of the new Console terminal. If omitted, the :CONSOLE command displays the current logical device number of the Console.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It may be used by any user to determine the location of the console. To change the location of the Console, it must be issued from the Console*, or the user must have SM or OP capability.

* Also may be distributed to users with the :ALLOW command.

OPERATION

The :CONSOLE command has purposes: To display the logical device number of the terminal currently being used as the System Console, and to move the Console to another logical device. Listing the location of the Console requires no special capabilities. Moving the Console requires OP or SM capability.

The Console cannot be moved to a remote terminal, i.e. one that is connected to your system via a Distributed Systems (DS) communications line. Similarly, it should not be moved to a terminal using a Multipoint Terminal Software (MTS) line, or a PAD terminal over a modem.

When you switch the location of the Console with the :CONSOLE command, a message is printed on the former Console and on the logical device receiving the Console to display the new logical device number of the System Console. The old Console is now just another session device and all the Console capabilities are transferred to the newly designated terminal.

Without parameters, the :CONSOLE command will report the current logical device number of the Console. No capabilities are required to execute the command this way.

NOTE

Before transferring the System Console to another terminal, be sure that you can take the Console back when desired (:ALLOW user.account;commands=CONSOLE). Users assigned System Manager (SM) capability can retrieve the Console without having been allowed the use of the :CONSOLE command.

Since the System Console is a session device, a session must be logged on to the Console in order to execute Operator commands.

Control and Maintenance Processor (CMP) and Diagnostic Control Unit (DCU) prompts and messages remain with the configured, e.g. Channel 1, Device 0. This feature cannot be moved to another terminal.

EXAMPLE

To determine the current location of the System Console, enter:

```
:CONSOLE
CONSOLE IS CURRENTLY ASSIGNED TO LDEV 20
```

To transfer the Console to the terminal identified by MPE as logical device 31, enter:

```
:CONSOLE 31
CONSOLE HAS BEEN SWITCHED FROM LDEV 20 TO LDEV 31
```

:CONTINUE

Overrides job error so that the job continues executing.

SYNTAX

```
:CONTINUE
```

PARAMETERS

None.

USE

This command may be issued in a Session, Job, or in a BREAK, but not from a Program. It is not Breakable.

OPERATION

The :CONTINUE command permits a job or session to continue even though the command immediately following :CONTINUE results in an error (with accompanying error message). It is not needed in a session because sessions do not terminate when a command error occurs. The :CONTINUE command is typically used in the line preceding any command suspected of causing the job to abort. If an error does occur, the job will continue to run, and the job output will contain the error message.

EXAMPLE

If you anticipate a possible error resulting from the command "RUN MYPROG", and wish to override this error and allow the job to continue executing, enter:

```
!JOB MAC.PUBS
!CONTINUE
!RUN MYPROG
!RUN MYPROG2
!EOJ
```

ADDITIONAL DISCUSSION

Refer to Section III of this manual.

:DATA

Enters data into the system from any devicefile except $STDIN.

SYNTAX

```
:DATA [jsname,] username[/userpass].acctname[/acctpass][;filename]
```

PARAMETERS

jsname	Name of job or session that is to read data. Default is no job/session name.
username	User name that allows you to access MPE under this account, as established by the Account Manager.
userpass	User password, optionally assigned by the Account Manager.
acctname	Account name under which job/session is running, as established by the System Manager.
acctpass	Account password, optionally assigned by System Manager.
filename	Additional qualifying name for the data that can be used by a job or session to access data. It may be used to distinguish two separate data decks read from different card readers by the same program. Default is that no distinguishing name is assigned.

Note that the *jsname*, *username*, *userpass*, *acctname*, *acctpass*, and *filename* parameters all are names that contain as many as eight alphanumeric characters, beginning with an alphabetic.

USE

This command may be issued from a Session or Job, but may not come from $STDIN. It may not be used from a Program or in BREAK. It is not Breakable.

OPERATION

This command identifies input that is to be read from a devicefile other than the standard input device. The :DATA command would be used to read a file from a spooled input device while running an interactive session. However, if the :DATA command is entered from the terminal during a session, it makes the terminal a data file for some unspecified process, which prevents you from using it until a process issues a request for data from the terminal.

The :DATA command is the only command that can be entered before a job or session is initiated with the :JOB or :HELLO commands. Files identified by :DATA may be input on cards, magnetic tape, paper tape, or from a terminal. To designate a set of data as an auxiliary file for a job or session, enter the :DATA command followed by the data deck or file, and the :EOD command. Begin the job or session using the same identity ([*jsname*,] *username*.*acctname*), used in the :DATA command so that the data can be used by the job or session. If the *filename* parameter is omitted, several data decks or files can be read from any job or session with the same identity.

The :STREAM command must be used to submit data from a disc file. When data is entered from magnetic tape, the file must reside on a single tape volume, with a blocking factor of 1. When the media containing the data file is placed on the input device and that device is placed online, MPE reads the entire file if the input device is spooled. Only the :DATA command is read if the file is input from a nonspooled device. At that point, the job can access the data, which remains available until it is actually read.

EXAMPLE

A session is running and is identified by the session name SESSA, username BROWN, and account name ACCT1. Data on punched cards needs to be made available to that session, to be used by a program named PROGY. This program references the data under the formal file designator DATAFL. Proceed as follows:

1. Arrange the data deck beginning with the :DATA command and terminating with the :EOD command, as follows:

```
:DATA SESSA,BROWN.ACCT1
      .
      .
      .
     data
      .
      .
      .
:EOD
```

2. Load the cards into the card reader and make the reader READY. If the card reader is a spooled device, the data is copied to disc, where it awaits access. If it is not a spooled device, the :DATA command is read, but the subsequent data remains in the card reader until the program accesses it. The :DATA command is used to build an entry in the device directory that identifies the file to the system.

3. Begin your session, making sure to use precisely the same session identity you entered in the :DATA command. (For instance, if the optional session name is omitted SESSA, the session will not be able to access the data.) To logon, enter:

```
:HELLO SESSA,BROWN.ACCT1
```

4. Enter a :FILE command equating the formal file designator DATAFL (used by the program PROGY) with the card reader. For example:

```
:FILE DATAFL;DEV=CARD
```

5. Run the program that requires the data; when the program attempts to open the file DATAFL, it will be available:

```
:RUN PROGY
```

If the data has not been entered through the card reader before this step, the program transmits a message to the System Operator requesting this data and waits for it to become available. It is then the System Operator's option whether to supply the data via the :DATA command or, if the data is not available, to send a message asking you to supply it, or abort the session.

6. Once the data has been read, it is no longer available to the system. If another program requires this data, the data must be entered again.

In the next example, a data file is created on disc and the :STREAM command is used to make the file available to your program:

1. Create the file DATAFL on disc beginning with the :DATA command and ending with the :EOD command.

```
:DATA SESSB,BROWN.ACCT1
      .
      .
      .
     data
      .
      .
      .
:EOD
```

2. Stream the data file, using the :STREAM command as follows:

```
:STREAM DATAFL
```

3. Log onto a session, making sure that precisely the same session identity is used as was entered in the :DATA command:

```
:HELLO SESSB,BROWN.ACCT1
```

4. Enter a :FILE command equating the formal file designator (used by the program) with the streams device (identified by the device class name JOBTAPE):

```
:FILE DATAFL;DEV=JOBTAPE
```

5. Run the program that requires the data:

```
:RUN PROGY
```

:DEALLOCATE

Deallocates a program or procedure previously loaded into memory with the :ALLOCATE command.

SYNTAX

```
:DEALLOCATE [PROGRAM,    ]name
            [PROCEDURE,  ]
```

PARAMETERS

PROGRAM	The program file indicated by *name* is deallocated. Default.
PROCEDURE	The code segment containing the procedure specified by *name* in SL.PUB.SYS is deallocated. Default is PROGRAM.
name	The name of the program file or procedure to be deallocated.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have OP capability to execute this command.

OPERATION

:DEALLOCATE will immediately release table entries belonging to a program file or procedure that has been permanently allocated. If the program is currently executing, the command will take effect once the program or procedure is no longer in use.

EXAMPLE

To deallocate a program filenamed PROGEX, enter:

```
:DEALLOCATE PROGEX
```

:DEALLOCATE does not give back memory, it gives back table entries.

:DEBUG

Invokes the MPE Debug Facility.

SYNTAX

```
:DEBUG
```

PARAMETERS

None.

USE

This command may be issued from a Session, Program, or in BREAK. It may not be issued from a Job. It is not Breakable. A user must have Privilege Mode (PM) capability to use this command.

OPERATION

The :DEBUG command, which can only be executed by a user with Privileged Mode (PM) capability, is an extension of the MPE Debug Facility. It is used primarily by systems programmers to analyze code and trace errors in program logic. :DEBUG is fully explained in the MPE Debug/Stack Dump Reference Manual (30000-90012).

CAUTION

The normal checks and limitations that apply to standard MPE users are bypassed in Privileged Mode. It is possible for a Privileged Mode user to destroy the integrity of the system, including MPE operating system software itself. Hewlett-Packard will investigate and attempt to resolve the problems resulting from the use of Privileged Mode code. This service, which is not provided under the standard Service Contract, is available on a time and materials billing basis. However, Hewlett-Packard will not support, correct, or attend to any modification of the MPE operating system software.

:DELETESPOOLFILE

Deletes a spoolfile from disc.

SYNTAX

```
                    {#Onnn}
:DELETESPOOLFILE    {#Innn}
                    {ldev }
```

PARAMETERS

#Onnn	The identification of a READY or ACTIVE output spoolfile.
#Innn	The identification of a READY, input spooled :DATA file.
ldev	The logical device number on which the spoolfile is ACTIVE.

USE

This command may be issued from a Session, Program, or in BREAK. It may not be issued from a Job. It is not Breakable. This command may be used only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

Before deleting an ACTIVE spoolfile, you should first take the output device offline. This allows you time to enter the command and determine that the ACTIVE spoolfile corresponds to the correct output device. When MPE returns the colon prompt (:), you know that the :DELETESPOOLFILE instruction has been sent to the spooler process. It will not be executed, however, until the output device is put back online.

You may not use the :DELETESPOOLFILE command on the following files:

- System defined standard input spoolfiles ($STDIN). Delete them with the :ABORTJOB command.
- ACTIVE spoolfiles with :DATA input, entered via the :STREAM command. You may delete these only when they are READY. You may also not delete these files when they are OPEN.

:DELETESPOOLFILE will delete ACTIVE :DATA input files which are submitted on a spooled device, such as a card reader. It cannot delete such files while they are being :STREAMED.

EXAMPLE

To delete the ACTIVE spoolfile being printed on *ldev* 6, first take the printer offline. This generates a "NOT READY" message at the Console, after which you may enter the :DELETESPOOLFILE command, as shown below:

```
11:21/7/LDEV#6 NOT READY
:DELETESPOOLFILE 6
```

When you put the device back online, the trailer page will be printed, and the file deleted. If you have suppressed header/trailer output with the :HEADOFF command, no trailer is printed before the spoolfile is deleted. However, the printer will skip to the top of the next physical page. If the device is a page printer, the default environment is reloaded.

:DISALLOW

Prohibits access to a specific Operator command.

SYNTAX

```
:DISALLOW FILE=formaldesignator[;SHOW]
```

```
                { @.@         }
                { user.@      }
:DISALLOW       { @.acct      } ;COMMANDS=command[;command;...]
                { user.acct   }
```

PARAMETERS

formaldesignator	A formal ASCII filename, which may consist of as many as eight alphanumeric characters, beginning with a an alphabetic character.
SHOW	Lists input lines on $STDLIST.
@.@	Prohibits access to all users whether logged on or not.
user.@	Prohibits access to a specific user in all accounts.
@.acct	Prohibits access to all users in a specific account.
user.acct	Prohibits access to a specific user in a specific account.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. This command may be used only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

The Operator uses the :DISALLOW command to prohibit a user from executing specific console commands previously allowed with the :ALLOW command. One form of the command requires you to enter the list of prohibited commands directly at the console. The second form of :DISALLOW is the indirect mode, in which you use EDIT/3000 create a file with EDIT/3000 that contains the username and account of those users who will be prohibited from executing certain Operator commands, and a list of disallowed commands. The procedure is similar to the following example:

```
:EDITOR
HP32201A.7.16 EDIT/3000  TUE, MAY 29, 1984,  5:08 PM
(C) HEWLETT-PACKARD CO. 1984
/ADD
    1      SUSAN.PAYROLL;COMMANDS=ALTJOB,ALTSPOOLFILE
    2      JOHN.ACCTNG;COMMANDS=ALTSPOOLFILE,DELETESPOOLFILE
    3      //
...
/KEEP COMNDTMP
/E
```

Once this file is defined, command input is taken from it rather than from data entered at the console. No continued records are permitted. The :DISALLOW command is then executed by entering the following:

```
:DISALLOW FILE=COMNDTMP;SHOW
```

The ;SHOW parameter directs MPE to display each command line as it is executed from the file.

A third form of the command is the subsystem mode. In this case, you need only type the command name, :DISALLOW, followed by a RETURN. You will then be prompted with a "> ", after which you enter command parameters. Subsystem mode ends when an EOF is received, or until you EXIT.

EXAMPLE

To prohibit the user MAC.TECH from executing the :REPLY and :ABORTIO commands, enter the following on the System Console:

```
:DISALLOW MAC.TECH;COMMANDS=REPLY,ABORTIO
```

To use subsystem mode to prohibit the user MGR.MANUALS from executing the :BREAKJOB command, enter the following on the System Console:

```
:DISALLOW
>MGR.MANUALS;COMMANDS=BREAKJOB
>EXIT
:
```

:DISASSOCIATE

Removes control of a device from the user.

SYNTAX

```
:DISASSOCIATE devclass
```

PARAMETERS

devclass	The name of a device class configured during SYSDUMP.

USE

This command may be issued from a Session, Program, or in BREAK. It is not Breakable.

OPERATION

This command negates a previously issued :ASSOCIATE command by removing control of a device class from a user. The command may be issued by the System Operator or by the user. The user implicitly disassociates a device when logging off.

EXAMPLE

To terminate control of the device class "TAPE", enter:

```
:DISASSOCIATE TAPE
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)

:DISCRPS

Enables or disables the Rotational Position Sensing (RPS) feature on a specified logical device. RPS is supported only on G.00.00 and later releases of the operating system. It requires a special firmware upgrade to CS-80 disc drives.

SYNTAX

```
:DISCRPS ldev {ENABLE }
              {DISABLE}
```

PARAMETERS

ldev	The logical device number of the specified CS-80 disc drive.
ENABLE	Enables Rotational Position Sensing on the device.
DISABLE	Disables Rotational Position Sensing on the device.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It may be executed only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

The :DISCRPS command allows you to enable or disable the Rotational Position Sensing feature for CS-80 disc drives. With RPS enabled, the disc drive signals its availability to do an I/O only when it is a small rotational distance away from the target data. This improves system performance when more than one drive is connected to the same HP-IB channel.

EXAMPLE

To enable the RPS feature on logical device 1 and display the drive's status, enter:

```
:DISCRPS 1,ENABLE
:SHOWDEV 1
LDEV        AVAIL        OWNERSHIP     VOID   DEN ASSOCIATION
1           DISC  (RPS) 50 FILES
```

:DISMOUNT

Causes a volume set that was explicitly mounted by the user to be dismounted.

SYNTAX

```
:DISMOUNT [{  *      } [.groupname[.acctname]]]
          {  vcsname }
```

PARAMETERS

* or *vcsname*	Specifies the volume set. The asterisk (*) specifies the home volume set for the group and account specified, or the logon group and account if they are not specified. The *vcsname* parameter specifies the volume set/class name of a previously defined volume set name or volume class name. Omitting this parameter is equivalent to *.
groupname.acctname	Specifies the group and account which created the volume set. Default is the logon group and account.

USE

This command may be issued from a Session, Job, or in BREAK. It may not be used from a program. It is not Breakable.

OPERATION

The :DISMOUNT command allows you to dismount a volume set that you explicitly mounted using the :MOUNT command. If there are no other users of the volume set, the devices on which it resides are made available to the system. You can request a dismount only for a volume set that you have mounted; you cannot alter the status of the volume set for other users.

EXAMPLE

To dismount the volume set MYSET, previously mounted by you via a :MOUNT command, enter:

```
:DISMOUNT MYSET
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)

:DOWN

Removes a device from normal system use.

SYNTAX

```
:DOWN ldev
```

PARAMETERS

ldev	The logical device number of the device being taken offline.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be used only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

When you issue the :DOWN command for a device that is in use, the request will be satisfied when the process currently accessing it releases it.

Once you :DOWN a device corresponding to the master volume of a Nonsystem Domain disc drive, MPE will reject all further requests to use any volume in the set to which the master volume belongs.

The system console cannot be downed. Any attempt to do so will result in the following error message:

```
DOWN NOT PERFORMED ON CONSOLE DEVICE (CIERR 3150)
```

CAUTION

When any device is powered down without the use of the :DOWN command, subsequent access to that device can result in indefinite waiting, erroneous transfers, or other incorrect operation. Often these failures will occur with no indication to the System Operator or to the user. For this reason, it is very important that every device which is not fully operational (especially those which are powered down) be downed with the :DOWN command. A device that will be inoperable for more than a few hours can be temporarily removed from the I/O configuration at system start up.

EXAMPLE

To take logical device number 7 offline, enter:

```
:DOWN 7
```

To take logical device number 10 (an input spooled, job accepting magnetic tape) offline, enter:

```
:DOWN 10
:STOPSPOOL 10
11:16/31/SP#10/STOPPED
11:16/31/LDEV#10 NOT READY
```

:DOWNLOAD

Downloads format information to a line printer.

SYNTAX

```
:DOWNLOAD ldev [,filename ] [,...]
               [,MARGIN=nn]
```

PARAMETERS

ldev	The logical device number of the output device. This device must be a HP 2608 or HP 2563 line printer. (The HP2563 printer is supported only on the G.00.00 and later releases of MPE V.)
filename	The fully qualified name of a file containing the download control information.
nn	The print position that the first byte of data will assume. This number can be between 1 and 16, inclusive. Note that the HP 2608 Hardware documentation discusses a margin offset which varies from 0-15. This offset is not relevant to the margin parameter of the :DOWNLOAD command, as the software compensates for the hardware offset of *nn* -1.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

The Operator uses the :DOWNLOAD command to transmit format information to system printers only. It cannot be used with remote printers.

The Vertical Format Control (VFC) image file (*filename*) can define the margin setting as well as the VFC image on an HP 2608A or HP 2608S line printer. The number of print lines per form is limited to 127. Although the HP 2608S printer recognizes the :DOWNLOAD command, Hewlett-Packard recommends controlling the HP 2608S with an environment file rather than the :DOWNLOAD command. You cannot download a VFC file to HP 2631B printer, only the MARGIN=*nn* is allowed. (For a discussion of special hardware considerations, refer to Section VIII of the MPE System Operation and Resource Management Manual (32033-90005).

If the MARGIN=nn parameter is specified on an HP 2608A or HP 2608S, and a MARGIN record has also been specified in the VFC file, the MARGIN record in the VFC file overrides the MARGIN parameter of the :DOWNLOAD command. This parameter should only be used in cases where there is no MARGIN record in the VFC file.

When a particular print job has requirements for forms and/or a VFC file, the user will indicate this need via a FORMS message. (Refer to "EXAMPLE".)

CAUTION

Do Not Issue a :Download Command to an HP 2608S White a Spool File is Active. This will make the device UNAVAILABLE, and it will remain so until the system is restarted with a WARMSTART.

EXAMPLE

To respond to a forms message, enter:

```
IO/15:46/22/FORMS: PLEASE MOUNT PAYCHECK FORMS. USE VFC=VFCPAY
IO/15:46/22/SP#11/LDEV# FOR #S93;OUTFILE ON HP 2608 (1)

:DOWNLOAD 11,VFCPAY
```

To reset the VFC to its original state, you must reference a file that contains default specifications (such as VFC6 in this example) by entering:

```
:DOWNLOAD 11,VFC6.PUB.SYS
```

To set the left margin print position to column 4 (the installation defined default) enter:

```
:DOWNLOAD 11,MARGIN=4
```

:DSTAT

Displays the current status of the disc drives on the system.

SYNTAX

```
:DSTAT [ldev]
       [ALL ]
```

PARAMETERS

ldev	An integer specifying the logical device number of the disc drive whose status is desired.
ALL	Displays the status of all disc drives, both system and non system. The default is that if no parameter is included, only the status of non system discs is displayed.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

The :DSTAT command is used to display the current status of one or more disc drives on the system. For example:

```
:DSTAT ALL

LDEV-TYPE      STATUS        VOLUME (VOLUME SET-GEN)
----------     --------      -----------------------
1-7933/7935    SYSTEM        MARKET1
2-7920         SYSTEM        MARKET2
3-7905(R)      AVAIL         (USERVOL1.PUB.SYS-0)
4-7920         DOWNED
5-7920         AVAIL         (SLAVOL1.PUB.SYS-0)
6-7920         AVAIL         (SLAVOL2.PUB.SYS-0)
6-7920         AVAIL         (SLAVOL3.PUB.SYS-0)
```

The status can be:

STATUS	MEANING
AVAIL	A non system disc is physically mounted, but is not being used.
AVAIL *SCRATCH*	A non system disc has one bit modified in the disc label, so that it cannot be used. You can use the VINIT subsystem to reset this bit. The disc then assumes its former state (either serial disc or Private Volume).
DOWNED	The disc is not available for use.
FOREIGN	The disc is unformatted or contains data in a form not recognizable by MPE.
MOUNTED	The disc is mounted and currently is being accessed.
OFF LINE	The disc is spun down.
SYSTEM	The disc is allocated to the operating system.
SERIAL	This is a serial disc.
DOWN-PND	A :DOWN command has been issued for this disc to make it unavailable for use. However, since the disc was still in use when the command was issued, the :DOWN is pending completion.
RESERVED	A non system disc has been reserved for use by a process.

EXAMPLE

To display the status of LDEV 7 enter:

```
:DSTAT 7

LDEV-TYPE      STATUS          VOLUME (VOLUME SET-GEN)
----------      --------        ------------------------
7-7902          FOREIGN         *UNALLOCATED*
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)

Runs EDIT/3000.

SYNTAX

```
:EDITOR [listfile]
```

PARAMETERS

listfile Actual file designator of file to receive any output resulting from EDIT/3000 commands LIST and XPLAIN when the OFFLINE option is specified. It can be any ASCII output file. The formal file designator and default is EDTLIST. If specified with no device parameter, default device is LP.

NOTE

The formal file designator used in this command (EDTLIST) cannot be backreferenced as an actual file designator in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command in this section.

USE

This command may be issued from a Session or Job. It may not be used from a program or in BREAK. It is Breakable (suspends execution).

OPERATION

The :EDITOR command runs the EDIT/3000 subsystem.

EXAMPLE

To runs EDIT/3000 during a session and specify a line printer (device class LP) as the list device for off-line output, enter:

```
:FILE LISTFILE;DEV=LP
:EDITOR *LISTFILE
```

Because the *listfile* is often a line printer, it is often defined in an MPE :FILE command and backreferenced as in the preceding example.

ADDITIONAL DISCUSSION

EDIT/3000 Reference Manual (03000-90012)

:ELSE

Provides an alternate execution sequence for an :IF statement.

SYNTAX

```
:ELSE
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, or in BREAK. It may not be used from a Program. It is not Breakable.

OPERATION

The :ELSE command is used only in conjunction with the :IF command. The :IF command is used with the :ENDIF command, and optionally with the :ELSE command, to control the execution of a job. The :IF, :ENDIF, and optional :ELSE commands constitute an IF block. A logical expression is evaluated, and if true, the IF block is executed; if false, the ELSE block (if one exists) is executed.

EXAMPLE

The following job listing illustrates use of :ELSE command:

```
!CONTINUE
!SPL MYPROG,MYUSL
!IF JCW>=FATAL THEN
!   TELL MAC.TECHPUBS;COMPILE FAILED
!ELSE
!   TELL MAC.TECHPUBS;COMPILE COMPLETED
!ENDIF
```

For more information on Job Control Words (JCWs), refer to :SETJCW and :SHOWJCW in this section.

:ENDIF

Terminates an IF block.

SYNTAX

```
:ENDIF
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, or in BREAK. It may not be used from a Program. It is not Breakable.

OPERATION

The :ENDIF command is used to terminate an IF block. The :IF command, the optional :ELSE command, and :ENDIF command constitute an IF block. They are used to control the execution of a job. A logical expression is evaluated, and if true, the IF block is executed; if false, the :ELSE block (if one exists) is executed. The example on the preceding page illustrates how this command may be used.

Denotes end-of-data on input stream and terminates a set of data initialized by the :DATA command.

SYNTAX

```
:EOD
```

PARAMETERS

None.

USE

This command may be issued from a Session or a Job. It may not be used from a Program or in BREAK. It is not Breakable.

OPERATION

The :EOD command is used to signify the end of a data set entered through a standard input device. It is also used to delimit data entered via the :DATA command submitted on data accepting devices such as tape drives and card readers.

In most cases, programmers use :EOD for delimiting data. Any record beginning with a colon, however, will delimit data. Using a command other than :EOD for this purpose depends upon whether the standard input file is opened with the filename $STDIN or $STDINX. Refer to Table 2-4 on the following page to see which delimiters are permitted with which files. If you enter certain end-of-file indicators such as :EOD or ::EOF: interactively while using MPE or from a subsystem, you may terminate your session.

When using a compiler language that does not provide a convention for terminating compilation (such as END in SPL), you must enter :EOD after the last record of your source program to ensure proper delimiting of your input. The :EOD command is not required when using the BASIC interpreter since the subsystem provides different conventions for delimiting data.

The :EOD command causes the FREAD intrinsic to return the CCG condition code to the calling program, which indicates the end-of-file condition on the terminal.

Table 2-4. End-of-File Indicators

TYPE OF FILE	INDICATORS	
DATA file from Standard Input Device (for sessions)	:EOD ::EOF:	Terminates $STDIN and $STDINX.
	:followed by any other character	Terminates $STDIN.
DATA file from Standard Input Device (for jobs)	:EOJ :JOB :EOD :DATA	Terminates $STDIN.
	:EOD ::EOF:	Terminates $STDINX.
	: followed by any other character	Terminates $STDIN. This record is then interpreted by the Command Interpreter as the next command to be executed.
:DATA files	:EOD :JOB :DATA ::EOF:	

EXAMPLE

To terminate a data file entered on cards for a session identified as SESS1,BLACK.ACCTSP, your data deck would contain :EOD as last record. Enter:

```
:DATA SESS1,BLACK.ACCTSP
        .
        .
        data
        .
        .
:EOD
```

The following FORTRAN program is an example of how the :EOD command is used to terminate a data set data entered through a standard input device:

```
:FORTRAN

PAGE 0001   HP32102B.01.04   (C)  HEWLETT-PACKARD CO. 1985


>$CONTROL USLINIT
>   PROGRAM MONEY
>   INTEGER QUARTERS,DIMES,NICKELS,PENNIES
>   DISPLAY "INPUT MONEY AMOUNT IN DECIMAL FORM "
>   ACCEPT DECIMALFORM
>   CALL CHANGER(DECIMALFORM,QUARTERS,DIMES,NICKELS,PENNIES)
>   DISPLAY QUARTERS," QUARTERS"
>   DISPLAY DIMES," DIMES"
>   DISPLAY NICKELS," NICKELS"
>   DISPLAY PENNIES," PENNIES"
>   STOP
>   END


PROGRAM UNIT MONEY COMPILED
>   SUBROUTINE CHANGER(DECIMALFORM,QUARTERS,DIMES,NICKELS,PENNIES)
>   INTEGER QUARTERS,DIMES,NICKELS,PENNIES
>   DECIMALFORM = DECIMALFORM*100
>   QUARTERS = DECIMALFORM/25
>   REMAINDER = DECIMALFORM-(QUARTERS*25)
>   DIMES=REMAINDER/10
>   REMAINDER=REMAINDER-(DIMES*10)
>   NICKELS=REMAINDER/5
>   PENNIES=REMAINDER-(NICKELS*5)
>   RETURN
>   END


PROGRAM UNIT CHANGER COMPILED
>:EOD


****       GLOBAL STATISTICS        ****
****    NO ERRORS,    NO WARNINGS   ****
TOTAL COMPILATION TIME  0:00:01
TOTAL ELAPSED TIME      0:01:29


END OF COMPILE
```

::EOF:

Simulates physical end-of-file on input stream from any device.

SYNTAX

```
::EOF:
```

PARAMETERS

None.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is not Breakable.

OPERATION

This command denotes the end-of-file when read through an input stream. It is required to simulate a physical end-of-file on HP 2894A Card Reader/Punch when that device is configured to refuse data submitted via a :DATA command. (This device does not provide a true hardware end-of-file indication.) Note that use of this command interactively will terminate your session.

EXAMPLE

To denote the end-of-file read through an input stream, enter:

```
:EOF:       (The last colon in this command must be followed by a blank or RETURN.)
:
```

ADDITIONAL DISCUSSION

For more information on end-of-file indicators refer to the :EOD command in this section.

Ends a batch job.

SYNTAX

```
:EOJ
```

PARAMETERS

None.

USE

This command may be issued from a Job. It may not be used from a Session, Program or in BREAK. It is not Breakable.

OPERATION

The :EOJ command terminates a batch job and displays the CPU time (in seconds) and the elapsed time since the beginning of the job (rounded to the nearest minute) to be displayed. MPE also adds the central processor time and file space used by your job to the resource usage counters maintained for your logon account and group.

If you omit the :EOJ command from a job, the next :JOB command terminates the current job and starts a new one. The end of the first job is indicated by the standard end-of-job display, and the beginning of the next job is denoted by the normal job initiation display.

EXAMPLE

The following example shows how :EOJ is used within a job file to terminate a batch job:

```
!JOB MAC.PUBS
!RUN MYPROG1
!RUN MYPROG2
!EOJ
```

:FCOPY

Runs the FCOPY subsystem.

SYNTAX

```
:FCOPY [fcopycommand]
```

PARAMETERS

fcopycommand An FCOPY/3000 subsystem command. The FCOPY/3000 subsystem enables you to copy files or selected portions of files from any supported input device to any supported output device. There are many commands; only the most common examples will be found in the "EXAMPLE" section of this command. Refer to the FCOPY Reference Manual (03000-90064) for more information.

USE

This command may be issued from a Session or a Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

This command runs the FCOPY subsystem from MPE. If the command is entered with no parameters, :FCOPY prompts (>) the user for subsystem commands until an EXIT command is entered. If the *fcopycommand* parameter is used, :FCOPY executes the FCOPY subsystem command and then returns control to MPE.

EXAMPLE

To access the FCOPY subsystem to execute multiple commands, enter:

```
:FCOPY
HP32212A.3.12 FILE COPIER (C) HEWLETT-PACKARD CO. 1985

>                              ** FCOPY subsystem prompt character. **
```

To access FCOPY to execute a single command and return control to MPE, enter the command as follows:

```
:FCOPY FROM=UDC.TECHPUBS;TO=TEMP;NEW

HP32212A.3.19FILE COPIER @ HEWLETT-PACKARD CO., 1984.

EOF FOUND IN FROMFILE AFTER RECORD 23

24 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
:
```

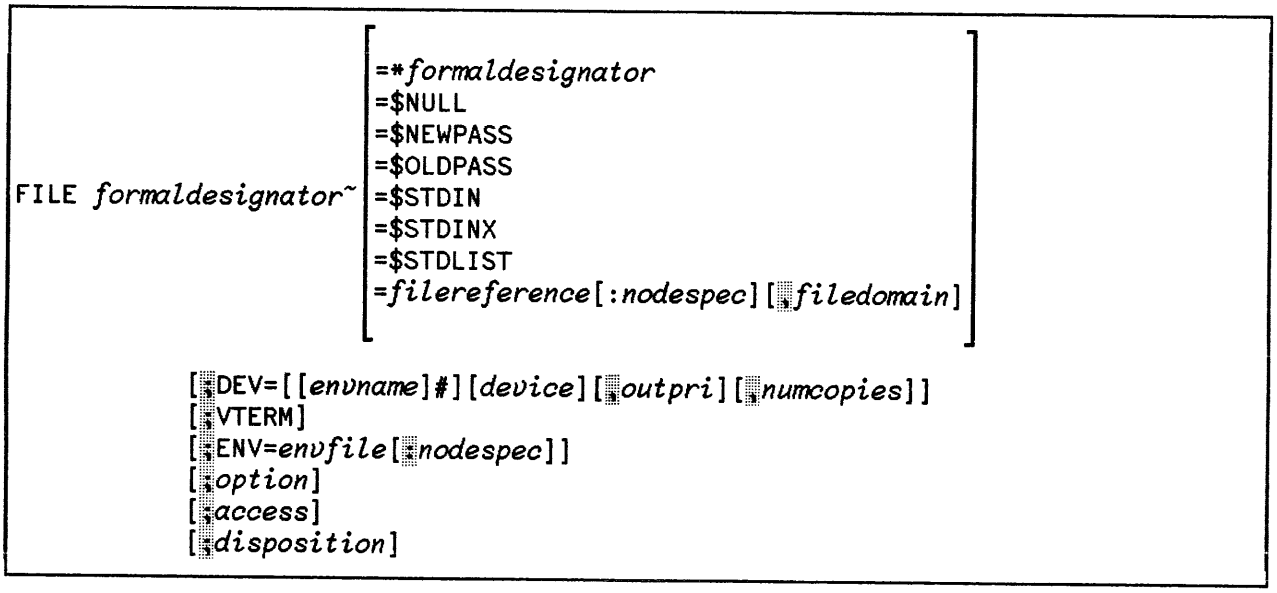
ADDITIONAL DISCUSSION

FCOPY Reference Manual (03000-90064)

:FILE

Declares the file attributes to be used when a file is opened. This declaration, informally known as a file equation, may be used to override programmatic or system default file specifications. With the addition of shared parameters from the NS/3000 AdvanceNet or the DS/3000 subsystem, the declaration may specify a formal file designator that may be used to access a remote file or device in a subsequent command or intrinsic. NS/3000 AdvanceNet and DS/3000 are not part of the HP 3000 Fundamental Operating System and must be purchased separately.

SYNTAX



formaldesignator A formal file designator that has the form

filename[,*groupname*[,*accountname*]] [;*nodespec*]

in which *filename*, *groupname*, and *accountname* are the identifiers that form a fully qualified filename. Each identifier may contain from one to eight alphnumeric characters, beginning with an alphabetic character. This filename may be used to identify the file in subsequent commands or intrinsic calls.

The *nodespec* extension of the formal file designator, explained below, is a parameter shared with the NS/3000 AdvanceNet subsystem and the DS/3000 subsystem. It is not part of the Fundamental Operating System. MPE accepts this extended formal file designator, with a node specification following a colon (:), in :FILE and :RESET and in the FOPEN intrinsic. Refer to the :RESET command in this section and also the MPE V Intrinsics Reference Manual (32033-90006).

You should be aware that the FOPEN intrinsic recognizes only *filename* in the *formalfidesignator*. FOPEN ignores any *groupname* or *accountname* specifications.

If *formaldesignator* is not equated to another file designation, the parameter specifies the name of an actual file. Placing an asterisk ahead of the parameter (**formaldesignator*) establishes a backreference to a formal file designator defined in a :FILE command.

The default is a temporary, nameless file that may be written to but not saved.

$NULL — Actual file designator of a system defined file that is always treated as an empty file. When $NULL is accessed by a program for input, that program receives only an end-of-file indication. When it is accessed by a program for output, the associated write request is accepted by MPE, but no physical output is actually performed.

$NEWPASS — A system defined temporary job file. When $NEWPASS is closed, it is referenced by the name $OLDPASS. Opening $NEWPASS destroys any previous $NEWPASS temporary file.

OLDPASS — The system defined name of the last temporary file that was closed as $NEWPASS.

$STDIN — The system defined name of the standard job input device. A colon (:) as the first character read on this file indicates end-of-file.

$STDINX — The same as $STDIN except that a colon can be read as the first character and received as data (::EOF: and :EOD statements cannot be read as data).

$STDLIST — The system defined name for the standard job or session limit device.

filereference — The actual file designator of the file, which has the form

filename[/*lockword*] [.*groupname*[.accountname]]

Each identifier may contain from one to eight alphanumeric characters, beginning with an alphabetic character. In a batch job, the file will fail to open if the file has a *lockword* that is not specified in *filereference*. In a session, MPE will prompt you for a *lockword* if one exists.

nodespec — An extension of the formal file reference. It may be an environment identification (specified in a previous :DSLINE or :REMOTE command), or it may be $BACK. It may appear in the formal file designator of the file or as an extension of an actual file reference. If an environment identification appears in a file designation and in the DEV= option, an attempt to open the file (with the FOPEN intrinsic, for example) produces an error.

$BACK instructs MPE to "hop backward" one node toward your local system to find the specified file. This works only if the :FILE command is issued in a remote session. If the systems involved are connected in a Local Area Network (LAN), one "hop backward" always means returning to your local system. The $BACK specification is the same as DEV=# without an environment name.

NOTE

The *nodespec* parameter and the :DSLINE and :REMOTE commands are not part of the HP 3000 Fundamental Operating System. The NS/3000 AdvanceNect subsystem or the DS/3000 subsystem must be purchased separately. The *nodespec* parameter is optional: if you do not have NS/3000 AdvanceNet or DS/3000, omitting the *nodespec* parameter will make no difference in the performance of the :FILE command.

However, specifying *nodespec* on a system that does not have NS/3000 or DS/3000 will produce an error. The *nodespec* parameter is controlled by the NS/3000 or DS/3000 subsystem. Refer to the NS/3000 User/Programmer Reference Manual (32344-90001) or the DSN/DS User/Programmer Reference Manual (32189-90001).

filedomain — The domain of the file, which may be OLD, NEW, or OLDTEMP.

- NEW — Creates a new file, which becomes the default. The NEW file may be permanent or temporary, depending on how the file was created. You must use either the :BUILD command or the FOPEN intrinsic to create the file. Refer to the :BUILD command in this section.
- OLD — Specifies an existing permanent file that was saved in the system or in the Private Volume domain. The file continues to exist after the current job or session ends.
- OLDTEMP — Specifies a temporary file that already exists in the temporary session or job file domain. The file is deleted at the end of the current job or session.

envname — This may be a *nodespec*, a DS device class name, logical device number, or an X.25 node name. Over the DS compatible links, the logical device number must not be preceded by the DSLDEV prefix. Instead, use just the logical device number. The parameter *envname* may consist of as many as eight alphanumeric characters, beginning with an alphabetic character.

NOTE

The *envname* parameter is not part of the HP 3000 Fundamental Operating System. The NS/3000 AdvanceNet subsystem or the DS/3000 subsystem must be purchased separately. The *envname* parameter is optional: If you do not have NS/3000 AdvanceNet or DS/3000, omitting the *envname* parameter will make no difference in the performance of the :FILE command.

However, specifying *envname* on a system that does not have NS/3000 or DS/3000 will produce an error. The *envname* parameter is controlled and by the NS/3000 or DS/3000 subsystem. Refer to the NS/3000 User/Programmer Reference Manual (32189-90001) or the DSN/DS User/Programmer Reference Manual (32344-90001).

device The logical device name or logical device number of a device, such as a disc, tape, printer, or a terminal. The default is DISC. If the DEV= option appears, it must be followed by at least one parameter (the parameter can be simply #).

If you are opening a file that is to reside on a Private Volume, you must specify a device class that includes the drives upon which the home volume set is mounted. The files is then allocated to any of the volume set's volumes that fall within that class. The default device class is DISC. A previously defined environment identifier is permitted in the DEV= option, but the *domain* and *organization* qualifiers are not permitted.

outpri The output priority requested for a spooled device. This may have a value of 1 (the lowest priority) to 13 (the highest).

numcopies The number of copies requested for a spooled output devicefile. The maximum number is 127.

VTERM Instructs MPE to use the Reverse Virtual Terminal service instead of the Remote File Access. You may use VTERM only if the designated device is a remote terminal. Using VTERM allows a local application program to perform I/O to remote terminals located on systems which support Reverse Virtual Terminal. Refer to "Communicator 3000", Volume 2, Issue 6 (Version G.002.00 of MPE V/E "U-MIT").

envfile The name of a file containing laser printer environment information, which controls the printing output formats on the laser printer. This name may be an actual file designator, or it may be a formal file designator preceded by an asterisk (*).

The information in this file contains such specifications as page size, the character font, forms, and other printer requirements to be used with the HP Laser Printing System, entered with IDS/3000 and IFS/3000. The file must be in a form suitable for downloading to the HP2680A, 2688, 2563A, or 2608S.

To specify an environment file called TRADMIN.HPENV.HP2680A for a form printed on the HP2680A, enter:

```
:FILE LP;DEV=LASER;ENV=TRADMIN.HPENV.HP2680A
```

Refer to the IFS/3000 Reference Guide (36580-90001) for discussions on creating environment files.

The ENV= parameter in a :FILE command overrides the environment specified in the FOPEN intrinsic. If the ENV= parameter is used and the **formalfiledesignator* sor *filereference* is omitted, any environment file specification in a subsequent FOPEN of the devicefile is ignored.

option Any valid option for the :FILE command.

SYNTAX FOR OPTION

```
;REC=[recsize][,[blockfactor][,{F}[,{BINARY}]]]]
                               {U}  {ASCII }
                               {V}
;DEN=[density]]

;DISC=[numrec][,numextents][,initialloc]]

;CODE=[filecode]]

[{;RIO  }]
{;NORIO}

{;STD}
{;MSG}]
{;CIR}
```

PARAMETERS FOR OPTION

recsize Record size. A positive number indicates words, while a negative number indicates bytes for new files only. For fixed length files, this is the logical record size. For undefined length files, this is the maximum record size. For variable length files, this is the maximum logical record size if *blockfactor* is 1. If not, this is used to calculate the maximum logical record size and physical record size.

Records always begin on word boundaries. Therefore, the record size is rounded up to the nearest word boundary for block size calculations. For a binary file or a variable length ASCII file, odd byte lengths are rounded up and the extra byte is available for data.

However, if an odd byte length record size is specified for a fixed or undefined length record file, the extra byte is not available for data. Default is the configured physical record width of the associated device. If you do not use the DEV= parameter, the default will be DISC with 128 records.

For example, a fixed length ASCII file with a record size specified as 11 bytes will have only 11 bytes available for data in each logical record. However, to determine actual block size, 12 bytes will be used for the record size (block size = 12 bytes multiplied by the *blockfactor*). If the file was specified as a binary file, the 11 bytes would be rounded up to 12 bytes (6 words), all of which are available for each logical record.

blockfactor — Number of logical records per physical block, for new files only. Default is calculated by dividing the specified *recsize* into the configured block size; this value is rounded downward to an integer that is never less than 1. For variable length record files, *blockfactor* is always set to 1 after using the original value along with *recsize* to calculate maximum logical record size and physical record size. The *blockfactor* is ignored for undefined length records. Maximum size is 255.

F, U, or V — Defines the format of the records of the file. A file may contain fixed length records (F), undefined length records (U), or variable length records (V). Default is F for disc files.

BINARY or ASCII — Indicates the type of records. BINARY indicates binary coded records and is the default. ASCII indicates ASCII coded records.

density — Corresponds to tape densities in BPI (bits-per-inch) for new files only. This parameter is only applicable when writing to a tape mounted on the HP 7976A or HP 7978A, variable density tape drive.

The density value from a file equation takes precedence over the density specified in FOPEN. The supported densities are 800, 1600 and 6250. For details on the operation of density selection, refer to FOPEN in the MPE V Intrinsics Reference Manual (32022-90007).

numrec — Maximum number of logical records, for new files only. For fixed and undefined length files the maximum value allowed for this field is 2,147,483,647. However, the maximum sectors per file is 2,097,120 based on the maximum of 65535 sectors per extent, 32 extents maximum. Thus the actual maximum number of records will be limited by block size (determined by *recsize* and *blockfactor*). An approximate practical limit for *numrec* is 2,097,119 for files with variable undefined lengths, and 267,382,000 for fixed length files. Default is 1023.

NOTE

The file system uses these values to compute other characteristics of the file as well. Therefore, the values (or default values) specified in the :FILE command may be valid within their respective fields, but may cause overflow errors in the computation of internally needed file specifications.

Refer to the MPE File System Reference Manual (30000-90236) for a discussion on calculating file space.

numextents Maximum number of disc extents. This is a value from 1 to 32. Default is 8.

initialloc Number of extents to be initially allocated to the file at the time it is opened. This is a value from 1 to 32. Default is 1.

filecode Code indicating a specially formatted file. This code is recorded in the file label and is available to processes accessing the file through the FGETINFO intrinsic. For this parameter, any user can specify a positive integer ranging from 0 to 32767 or a mnemonic name. Certain integers and mnemonics have been reserved for particular HP defined meanings, as follows:

Integer	Mnemonic	Meaning
1024	USL	User Subprogram Library
1025	BASD	BASIC Data
1026	BASP	BASIC Program
1027	BASFP	BASIC Fast Program
1028	RL	Relocatable library
1029	PROG	Program file
1031	SL	Segmented library
1035	VFORM	View Form file
1036	VFAST	View Fast Forms file
1037	VREF	View Reformat file
1040	XLSAV	Cross Loader ASCII file (SAVE)
1041	XLBIN	Cross Loader Relocated Binary file
1042	XLDSP	Cross Loader ASCII file (DISPLAY)
1050	EDITQ	Edit Quick file
1051	EDTCQ	Edit KEEPQ file (COBOL)
1052	EDTCT	Edit TEXT file (COBOL)
1054	TDPDT	TDP Diary file
1055	TDPQM	TDP Proof Marked QMARKED
1056	TDPP	TDP Proof Marked non-COBOL file
1057	TDPCP	TDP Proof Marked COBOL file
1058	TDPQ	TDP Workfile
1059	TDPXQ	TDP Workfile (COBOL)
1060	RJEPN	RJE Punch file
1070	QPROC	QUERY Procedure file
1080	KSAMK	KSAM Key file
1083	GRAPH	GRAPH Specification file

1084	SD	User Logging Log file
1090	LOG	Self-describing file
1100	WDOC	HPWORD Document
1101	WDICT	HPWORD Hyphenation dictionary
1102	WCONF	HPWORD Configuration file
1103	W2601	HP 2601 Environment file
1110	PCELL	IDS/3000 Character Cell file
1111	PFORM	IDS/3000 Form file
1112	PENV	IFS/3000 Environment file
1114	RASTR	Graphics Image in RASTR Format
1130	OPTLF	OPT/3000 Log file
1131	TEPES	TEPE/3000 Script file
1132	TEPEL	TEPE/3000 Log file
1133	SAMPL	APS/3000 Log file
1139	MPEDL	MPEDCP/DRP Log file
1140	TSR	HPToolset Root file
1141	TSD	HPToolset Data file
1145	DRAW	Drawing File for HPDRAW
1146	FIG	Figure File for HPDRAW
1147	FONT	Reserved
1148	COLOR	Reserved
1149	D48	Reserved
1152	SLATE	Compressed SLATE file
1153	SLATW	Expanded SLATE Workfile
1156	DSTOR	Store file for RAPID/3000 Utility DICTDBU
1157	TCODE	Code file for Transact/3000 Compiler
1158	RCODE	Code file for Report/3000 Compiler
1159	ICODE	Code file for Inform/3000 Compiler
1166	MDIST	HPDESK Distribution list
1167	MTEXT	HPDESK Text
1177	TTYPE	Term Type file
1178	TVFC	Term Vertical Format Control file
1192	NCONF	Network Configuration file
1193	NTRAC	Network Trace file
1194	NLOG	Network Log file
1211	ANODE	
1212	INODE	
1226	VC	VC file
1227	DIF	DIF file
1228	LANGD	Language Definition file
1229	CHARD	Character Set Definition file
1230	MGCAT	Formatted Application Message Catalog
1235	ATLAS	Reserved
1236	BMAP	Reserved
1258	PFSTA	Pathflow STATIC file
1259	PFDYN	Pathflow DYNAMIC file

Default is the unreserved file code of 0.

RIO or NORIO

Creates a relative or nonrelative I/O file. RIO creates a relative I/O file. The record length parameter will be implicitly changed to fixed record length. RIO is a special file access method primarily intended for use by COBOLII programs; however, you can access these files by programs written in any language. NORIO creates a nonrelative I/O file. Default is NORIO.

RIO and NORIO specifications affect only the physical characteristics of the file. If NOBUF is specified in the :FILE command, the file will be accessed in non-RIO mode; otherwise RIO access is used with RIO files. NOBUF access is provided for special operations on RIO files such as replicating RIO file. NOBUF is not normally used by the RIO user. Refer to the MPE V Intrinsics Reference Manual (32033-90007) for a discussion of relative I/O.

STD, MSG, or CIR

Defines the type of file. STD is a standard MPE disc file. MSG (message) file allows communication between any set of processes. MSG acts like a FIFO (first in, first out) queue where records are read from the start of the file and logically deleted and/or are appended to the end of file. CIR acts as normal sequential file until full. When full, the first physical block will be deleted when the next record is written, and remaining blocks will be logically shifted to front of file. CIR cannot be simultaneously accessed by readers and writers. Default is STD.

SYNTAX FOR ACCESS

```
[{;NOCCTL}]                                    {;NOMULTI}
[{;CCTL  }]                                    [{;MULTI  }]
                                               {;GMULTI }

       {IN     }                               [{;NOMR   }]
       {OUT    }                               [{;MR     }]
[;ACC=[{UPDATE }]]
       {OUTKEEP}                               [{;WAIT   }]
       {APPEND }                               [{;NOWAIT }]
       {INOUT  }

                                               [{;LOCK   }]
[{;BUF=[numbuffers]}]                          [{;NOLOCK }]
[{;NOBUF           }]
                                               [{;COPY   }]
 {;EXC }                                       [{;NOCOPY }]
[{;SHR }]
[{;EAR }]                                      [;FORMS=formsmsg]
 {;SEMI}

[{;NOLABEL
[{;LABEL=[volid ][;[{IBM}] [;[expdate][;seq]]]}]
                   {ANS}
```

PARAMETERS FOR ACCESS

NOCCTL or CCTL

Indicates whether carriage control characters are or are not specified. NOCCTL indicates that carriage control characters are not being specified in writes to the file. CCTL indicates that carriage control characters are being supplied in writes to the file. Default is NOCCTL.

IN, OUT, UPDATE, OUTKEEP, APPEND, or INOUT	Defines the type of file access. IN only permits READ access to the file and is the default for all input devices. OUT only permits WRITE access to the file and is the default for output devices. UPDATE permits any type of access to the file. OUTKEEP only permits WRITE access to the file, except previous data is not deleted. APPEND only permits APPEND access to any file. INOUT only permits input/output access; any file intrinsic except FUPDATE can be issued against the file.
BUF=*numbuffers* or NOBUF	Specifies whether buffers are to be allocated to the file. The *numbuffers* parameter is the number of buffers (1 to 16) to be allocated for the file. The *numbuffers* parameter is ignored for terminals. The default is BUF=2 buffers. NOBUF specifies that no buffers are allocated for the file.
EXC, SHR, EAR, or SEMI	Specifies shared or exclusive file access. EXC is exclusive access; after the file is opened no other accessors are permitted. For message and circular files, EXC means one writer and one reader. SHR is share access; after the file is opened other accessors are permitted. EAR is exclusive for one writer, allowing multiple readers. SEMI is intended for use with message files; it allows allows one exclusive reader, multiple writers. If the file is not a message file, SEMI acts like EAR (one exclusive writer, multiple readers). Default is EXC except with read only file access (IN).
NOLABEL or LABEL	Specifies if this tape is labeled or unlabeled. NOLABEL specifies that this is not a labeled tape. LABEL specifies that this is a labeled tape. Default is NOLABEL.
volid	Up to six alphanumeric characters identifying a labeled magnetic tape volume.
ANS or IBM	Type of standard label. ANS is ANSI-standard label. IBM is IBM-standard label. Default is ANS.
expdate	Month, day, year, written in the format *mm/dd/yy*. This specifies the expiration date of the file, or the date after which information contained in the file is no longer useful. The file can be overwritten without operator reconfirmation after this date. Default is 00/00/00: the file can be overwritten immediately.
seq	Either an absolute file number between 1 and 9999 (inclusive), or one of the following, which specifies the position of the file relative to other files on the tape:

0	Causes a search of all volumes until the file is found.
ADDF	ADDF positions the tape to add a new file on the end of the volume (or last volume in a multivolume set). Note that ADDF should not be used to add to a new labeled tape volume.

NEXT	NEXT positions the tape at the next file on the tape. If this is the first FOPEN, then NEXT will cause the tape to be positioned to the first file on the tape. If the previous FCLOSE specified REWIND, the tape backspaces to the last file, and the position is as it was, on the previous file. This is the Default. Refer to FOPEN in Section II of the MPE V Intrinsics Reference Manual (32033-90007) for more information.
NOMULTI, MULTI, or GMULTI	Specifies the sharing of files in jobs and sessions. NOMULTI prohibits sharing files in MULTI mode and is the default. MULTI allows concurrent accessors of the file and may regard the file as if no buffering is taking place. Access control information can be shared by the processes of the same CI process tree (i.e. father-to-son processes) with MULTI. GMULTI is the same as MULTI except it allows accessors to be in different jobs/sessions.
NOMR or MR	Specifies whether multirecord access is permitted. NOMR specifies that no multirecord access is permitted. MR allows multirecord access to the file. Default is NOMR.
WAIT or NOWAIT	Specifies whether I/O requests are to be completed or queued before control returns to the program. WAIT completes I/O requests to the file before control is returned to the program. NOWAIT returns control to the program as soon as I/O requests are queued by MPE; only Privileged Mode programs are allowed. In this way, the program does not have to wait for the physical I/O to be complete before resuming execution and also implies NOBUFF.
NOLOCK or LOCK	Specifies whether dynamic locking/unlocking is to be permitted. NOLOCK prohibits dynamic locking/unlocking of file through the FLOCK/FUNLOCK intrinsics. LOCK allows dynamic locking/unlocking through FLOCK/FUNLOCK intrinsics. Default is NOLOCK.
COPY or NOCOPY	Specifies whether files can be copied. COPY allows MSG, KSAM, and CIR files to be either copied (logical data record read) or replicated (block read and write completely duplicating file) to another file. NOCOPY accesses the file in its natural mode, i.e. as a MSG file. Default is NOCOPY.
formsmsg	A message to the operator requesting that certain forms be mounted. The message must be displayed and verified before the output data can be printed on a line printer. The message is a string of no more than 49 ASCII characters terminated by a period. Control characters for bells and inverse video may be sent to the System Console using this parameter. Attempts to send other control characters, however, will result in a display of blanks and the associated control character letter when the forms message appears on the System Console. Default is no forms message sent.

SYNTAX FOR DISPOSITION

```
 {;DEL  }
[{;TEMP}]
 {;SAVE}
```

PARAMETERS FOR DISPOSITION

DEL	The file is deleted when closed.
SAVE	The file is saved in the permanent file domain when closed.
TEMP	The file is saved in the job/session temporary domain when closed.

If none of these parameters are supplied, the disposition of the file is as it was when opened, or as specified by the FCLOSE intrinsic call issued by the user program. Refer to the MPE File System Reference Manual (30000-90236) for more information on file domains.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

This command allows you to change the specifications for files at run time, including the devices on which they reside, overriding specifications supplied via the FOPEN intrinsic. The :FILE command remains in effect for the entire job or session unless revoked by the :RESET command or superseded by another :FILE command.

To use the :FILE command for a file, you must have a valid, formal file designator (the name by which your program recognizes the file). The formal file designator provides a way for commands and code outside your program to reference the file.

NOTE

When you are programming in a language other than SPL, and you did not write the FOPEN intrinsic calls for files used by your program, the :FILE command is the only way you can control or change the programmatic file specifications.

EXAMPLES

A program, MYPROG, references two files by the filenames (*formaldesignators*) SOURCE and DEST, but you wish to use two existing disc files INX and OUTX as the actual files for the program. Enter:

```
:FILE SOURCE=INX
:FILE DEST=OUTX
:RUN MYPROG
```

Now you wish to run the program using input from a card reader which has 80-byte records and has the device class name CARD:

```
:FILE SOURCE,OLD;REC=-80;DEV=CARD
```

Output is to be sent to a new file, FILEX, with 64-word fixed length records, blocked two records per block in ASCII code. FILEX is limited to 800 records among 10 extents, two of which are to be immediately allocated. The file is to be permanently saved when MYPROG closes it. Enter:

```
:FILE DEST=FILEX,NEW;REC=64,2,F,ASCII;DISC=800,10,2;SAVE
:RUN MYPROG
```

Note that the file equation only modifies those items specified. All other attributes used will come from the parameters specified in the FOPEN call (or the defaults where parameters are omitted) for the file DEST.

Implicit :FILE Commands for Subsystems

When an actual file designator appears as a command parameter, it is automatically equated to a formal file designator. This is then used within the subsystem by an implicit :FILE command issued by the command executor. For instance, within the FORTRAN compiler the formal file designator for the text file input is FTNTEXT. Suppose you specify a file named ALSFILE for text file input as shown below:

```
:FORTRAN ALSFILE
```

MPE implicitly issues the following :FILE command, invisible to you:

```
:FILE FTNTEXT=ALSFILE
```

You cannot backreference any of the formal file designators associated with the command as actual file designators. Therefore, do not, use the formal file designators FTNTEXT, FTNUSL, FTNLIST, FTNMAST or FTNNEW as actual filenames. The use of FTNTEXT as a filename, as in the following example, is invalid because the implicit :FILE command issued by the FORTRAN compiler will then backreference itself:

```
:FORTRAN;*FTNTEXT
:FILE FTNTEXT=*FTNTEXT
```

The following is an example of a correct use of the **formaldesignator*, in this case specifying a file on magnetic tape used as a source file during a FORTRAN compilation:

```
:FILE SOURCE=TAPE1,OLD;DEV=TAPE;REC=-80
:FORTRAN;*SOURCE
```

Implicitly, the command executor issues the following :FILE command, backreferencing your previous :FILE command:

```
:FILE FTNTEXT=*SOURCE
```

Implicit :FILE commands, like explicit :FILE commands, cancel any previous :FILE commands that reference the same formal file designators. Formal file designators are described in each compiler command description throughout this manual.

ADDITIONAL DISCUSSION

MPE V Intrinsics Reference Manual (32033-90007)
MPE File System Reference Manual (30000-90236)

:FOREIGN

Directs MPE to treat the disc volume mounted on the specified *ldev* as foreign.

SYNTAX

```
:FOREIGN ldev
```

PARAMETERS

ldev	The logical device number of a disc drive.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

This command forces MPE to treat the volume currently mounted on the logical device as a foreign disc. The device class associated with the logical device must be foreign and the drive must be online, but the volume cannot be in use.

CAUTION

If the restrictions stated above are met, the system will treat the logical device as a foreign disc. As long as the volume is not a System Domain disc, MPE performs no security checking. Therefore, any disc (system, private volume, etc) could be mounted and inadvertently treated as a foreign disc. Since the Foreign Disc Facility may write on the MPE label area, file labels could be destroyed and important information lost.

EXAMPLES

To direct MPE to treat the disc mounted on device 3 as foreign, first verify the status of the disc by entering:

```
:DSTAT 3

  LDEV-TYPE          STATUS       VOLUME   (VOLUME SET-GEN)
------------         --------     ----------------------------

    3-7902           SERIAL              *UNALLOCATED*
```

Then enter:

```
:FOREIGN 3
```

To verify the change, enter:

```
:DSTAT 3

  LDEV-TYPE          STATUS       VOLUME   (VOLUME SET-GEN)
------------         --------     ----------------------------

    3-7902           FOREIGN             *UNALLOCATED*
```

:FORTGO

Compiles, prepares, and executes a FORTRAN/3000 program. FORTRAN/3000 is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:FORTGO [textfile][,[listfile][,[masterfile][,newfile]]]
```

PARAMETERS

textfile — Actual file designator of the input file from which the source program is read. This can be any ASCII input file. The formal file designator is FTNTEXT. Default is $STDIN.

listfile — Actual file designator of the file to which the program listing is written. This can be any ASCII output file. The formal file designator is FTNLIST. Default is $STDLIST.

masterfile — Actual file designator of the master file with which *textfile* is merged to produce a composite source. This can be any ASCII input file. The formal file designator is FTNMAST. Default is that the file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified.

newfile — Actual file designator of the file resulting from merging of *textfile* and *masterfile*. This can be any ASCII output file. The formal file designator is FTNNEW. Default is that the file is not written.

NOTE

The formal file designators used in this command (FTNTEXT, FTNLIST, FTNMAST, and FTNNEW) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

The :FORTGO command compiles, prepares, and executes a FORTRAN/3000 program. If you do not specify a source file, MPE expects input from your standard input device. If you do not specify *listfile*, MPE writes the listing to your standard output device.

The USL file created during the compilation is a system defined temporary file $OLDPASS, which is passed directly to the Segmenter, and cannot be accessed.

EXAMPLES

To compile, prepare, and execute a FORTRAN/3000 program entered from the disc file SOURCE and transmit the resulting program listing to the disc file LISTFL, enter:

```
:FORTGO SOURCE,LISTFL
```

To enter your source input from a device other than your standard input device, and/or direct the listing to a device other than your standard list device, simply name the input and listing files as command parameters. In the example below, the source listing is read from magnetic tape, formally identified by the filename MTAPE. Output is sent to the printer, identified by the filename PRTR.

```
:FILE MTAPE;DEV=TAPE
:FILE PRTR;DEV=FASTLP
```

MTAPE and PRTR are then backreferenced in the :FORTGO command, as shown here:

```
:FORTGO *MTAPE,*PRTR
```

ADDITIONAL DISCUSSION

FORTRAN/3000 Reference Manual (30000-90040)
MPE Segmenter Reference Manual (30000-90011)

:FORTPREP

Compiles and prepares a FORTRAN/3000 program. FORTRAN/3000 is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:FORTPREP [textfile][,[progfile][,[listfile][,[masterfile][,newfile]]]]
```

PARAMETERS

textfile — Actual file designator of the input file from which the source program is read. This can be any ASCII input file. The formal file designator is FTNTEXT. Default is $STDIN.

progfile — Actual file designator of the program file to which the prepared program segments are written. You will get the best results by omitting the *progfile* parameter. When you omit *progfile*, the MPE Segmenter will create the program file, which will reside in the temporary file domain as $OLDPASS. If you create your own program file, you must do so in one of two ways:

- Use the MPE :BUILD command, and specify a file code of 1029 or PROG, and a *numextents* value of 1. This file is then used by the :PREP command.
- Specify a nonexistent file in the *progfile* parameter, resulting in the creation of job/session temporary file of the correct type.

listfile — Actual file designator of the file to which the program listing is written. This can be any ASCII output file. The formal file designator is FTNLIST. Default is $STDLIST.

masterfile — Actual file designator of the master file with which *textfile* is merged to produce a composite source. This can be any ASCII input file. The formal file designator is FTNMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified.

newfile — Actual file designator of the file resulting from the merger of *textfile* and *masterfile*. This can be any ASCII output file. The formal file designator is FTNNEW. Default is that the file is not written.

NOTE

The formal file designators used in this command (FTNTEXT, FTNLIST, FTNMAST, and FTNNEW) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or a Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

This command compiles and prepares a FORTRAN/3000 program into a program file stored on disc. If you do not specify a source file, MPE expects the input from your standard input device. If you do not specify a *listfile*, MPE sends the output to your standard list device.

The USL file created during compilation is a system defined temporary file $OLDPASS, which is passed directly to the Segmenter. The Segmenter also uses the file OLDPASS. The prepared program segments are written to it, thus overwriting any existing temporary file of that name.

If you have no need to examine the USL file, use the default for *progfile*. This way, MPE will create a program file for you, ensuring the best results. If, on the other hand, you want to store the USL file and the program file as separate entities, create a permanent program file with the :BUILD command, and reference it in the :FORTPREP command line. When you create the file, you must specify a file code of "PROG" or "1029" and a *numextents* parameter value of 1, as shown in the example below.

```
:BUILD PROGX;CODE=PROG;DISC=1
```

EXAMPLES

To compile and prepare a FORTRAN/3000 program entered from your standard input device, into the standard default file $OLDPASS, with the listing printed on your standard list device, enter:

```
:FORTPREP
```

To compile and prepare a FORTRAN/3000 source program from a textfile named TEXTX into a program filenamed PROGX, with the resulting listing sent to the list file LISTX, enter:

```
:FORTPREP TEXTX,PROGX,LISTX
```

The :FORTPREP command combines the compilation and preparation steps. The compiled program segments, stored in the file $OLDPASS, are prepared and stored in the program file PROGX. Therefore, it is equivalent to:

```
:FORTRAN TEXTX, LISTX
:PREP $OLDPASS,PROGX
```

ADDITIONAL DISCUSSION

FORTRAN/3000 Reference Manual (30000-90040)
MPE Segmenter Reference Manual (30000-90011)

:FORTRAN

Compiles a FORTRAN/3000 program. FORTRAN/3000 is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:FORTRAN [textfile][,[uslfile][,[listfile][,[masterfile][,newfile]]]]
```

PARAMETERS

textfile — Actual file designator of the input file from which the source program is read. This can be any ASCII input file. The formal file designator is FTNTEXT. Default is $STDIN.

uslfile — Actual file designator of the User Subprogram Library (USL) file to which the object program is written, which can be any binary output file with file code of USL or 1024. The formal file designator is FTNUSL. If the *uslfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS. If entered, the file must have been created in one of four ways:

- By using the MPE :SAVE command to save default USL file $OLDPASS created during a previous compilation.
- By building the USL with the Segmenter -BUILDUSL command. (Refer to the MPE Segmenter Reference Manual (30000-90011).)
- By creating a new USL file with the MPE :BUILD command and specifying with a file code of USL or 1024.
- By specifying a nonexistent *uslfile* parameter, thereby creating a permanent file of the correct size and type.

listfile — Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is FTNLIST. Default is $STDLIST.

masterfile — Actual file designator of the master file with which *textfile* is merged to produce a composite source. This can be any ASCII input file. Formal file designator is FTNMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified.

newfile — Actual file designator of the merged *textfile* and *masterfile*. This can be any ASCII output file. Formal file designator is FTNNEW. Default is that no file is written.

NOTE

The formal file designators used in this command (FTNTEXT, FTNUSL, FTNLIST, FTNMAST, and FTNNEW) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

This command compiles a FORTRAN/3000 program into a USL file on disc. If you do not specify *textfile*, MPE expects input from your standard input device. If you do not specify *listfile*, MPE sends the listing to your standard list device.

If you create the USL file (using the MPE :BUILD command) before compiling the program, you must assign it a file code of USL or 1024. If you omit this parameter, the compiled program segments will be stored in the temporary file $OLDPASS.

EXAMPLES

To compile a FORTRAN/3000 program entered from your standard input device into an object program in the USL file $OLDPASS, and write the listing to your standard list device, enter:

```
:FORTRAN
```

The following example compiles a program from the source file MYSOURCE and stores the object code into the USL file MYUSL. The program listing will be stored in the disc file MYLIST:

```
:FORTRAN MYSOURCE,MYUSL,MYLIST;INFO= "$CONTROL BOUNDS"
```

To compile a FORTRAN/3000 program and store the object code into a USL file you create with the :BUILD command, enter:

```
:BUILD OBJECT;CODE=USL          **You must specify the CODE parameter as choose to
:FORTRAN SOURCE,OBJECT,LISTFL   create a USLfile with the :BUILD command.**
```

ADDITIONAL DISCUSSION

FORTRAN/3000 Reference Manual (30000-90040)
MPE Segmenter Reference Manual (30000-90011)

Releases a global Resource Identification Number (RIN).

SYNTAX

```
:FREERIN rin
```

PARAMETERS

rin	The Resource Identification Number (RIN) to be released. It must be a number from 1 to the configured maximum.

USE

This command may be issued from a Session or Job. It may not be used from a Program or in BREAK. It is not Breakable.

OPERATION

A Resource Identification Number is used to manage a resource shared among two or more jobs or sessions so that only one job or session at a time can access that resource.

The user acquires a RIN from the system by entering the :GETRIN command. When all users are finished with the RIN, the user who acquired it returns it to the system by entering the :FREERIN command. To free a RIN, you must be the original owner of that RIN, i.e. the user who actually issued the :GETRIN command that allocated the RIN and assigned it a password.

EXAMPLE

To release RIN 1, enter:

```
:FREERIN 1
```

ADDITIONAL DISCUSSION

MPE V Intrinsics Reference Manual (32033-90007)

:FTN

Compiles a FORTRAN/77 program. FORTRAN/77 is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:FTN [textfile][,[uslfile][,listfile]]
 [;INFO=quotedstring]
```

PARAMETERS

textfile Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is FTNTEXT. Default is $STDIN.

uslfile Actual file designator of the USL file to which the object code is stored, which can be any binary output file with a file code of USL or 1024. Its formal file designator is FTNUSL. If the *uslfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS. If entered, this parameter indicates that the USL file was created in one of four ways:

- By using the MPE :SAVE command to save, the default USL file $OLDPASS, created during a previous compilation.
- By building the USL with the MPE Segmenter -BUILDUSL command. Refer to the MPE Segmenter Reference Manual (30000-90011).
- By creating a new USL file specifying the MPE :BUILD command and specifying a file code of USL or 1024.
- By specifying a nonexistent *uslfile* parameter, thereby creating a permanent file of the correct size and type.

listfile Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is FTNLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (FTNTEXT, FTNUSL, and FTNLIST) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

quotedstring A sequence of characters between two single quotation marks (apostrophes) or between two double quotation marks. You may use the delimiter as part of the string so long as the delimiter appears twice. Any occurrence of two single quotation marks in a row, or two double quotes in a row, is considered part of the string, and, therefore, not the terminating delimiter.

INFO=*quotedstring* is used in the FORTRAN/77 programming language to pass initial compiler options to a program. For more information, refer to the :RUN command in this section.

USE

This command may be issued from a Session or a Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

The :FTN command compiles a FORTRAN/77 program and stores the object code in a User Subprogram Library (USL) file on disc. If *textfile* is not specified, MPE expects the source program to be entered from your standard input device. If you do not specify a *listfile*, MPE sends the program listing to your standard list device and identifies it by the formal file designator, FTNLIST.

If you create the USL prior to compilation, you must specify a file code of USL or 1024. If you omit the *uslfile* parameter, the object code is saved in the temporary file domain as $OLDPASS. To keep it as a permanent file, you must save $OLDPASS under another name.

EXAMPLES

The following example compiles a FORTRAN/77 program entered from your standard input device and stores the object program in the USL file $OLDPASS. The listing is then sent to your standard list device.

```
:FTN
```

The next example compiles a FORTRAN 77 program contained in the disc file FORTSRC, and stores the object program in the USL file FORTOBJ. The program listing is stored in the disc file LISTFILE.

```
:BUILD  FORTOBJ;CODE=USL

:FTN FORTSRC,FORTOBJ,LISTFILE
```

ADDITIONAL DISCUSSION

FORTRAN/77 Reference Manual (5957-4685)
MPE Segmenter Reference Manual (30000-90011)

:FTNGO

Compiles, prepares, and executes a FORTRAN/77 program. FORTRAN/77 is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:FTNGO [textfile][,listfile]
   [;INFO=quotedstring]
```

PARAMETERS

textfile — Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is FTNTEXT. Default is $STDLIST.

listfile — Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is FTNLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (FTNTEXT, and FTNLIST) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

quotedstring — A sequence of characters between two single quotation marks (apostrophes) or between two double quotation marks. You may use the delimiter as part of the string so long as the delimiter appears twice. Any occurrence of two single quotation marks in a row, or two double quotes in a row, is considered part of the string, and, therefore, not the terminating delimiter.

INFO=*quotedstring* is used in the FORTRAN/77 programming language to pass initial compiler options to a program. For more information, refer to the :RUN command in this section.

USE

This command may be issued from a Session or Job. It may not be used from a Program, or in BREAK. It is Breakable (suspends execution).

OPERATION

The :FTNGO command compiles, prepares, and executes a FORTRAN/77 program. If *textfile* is omitted, MPE expects input from your standard input device. If you do not specify *listfile*, MPE sends the program listing to the formal file designator FTNLIST (default is $STDLIST).

The USL file created during the compilation is the system defined temporary file $OLDPASS, which is passed directly to the Segmenter. It cannot be accessed because the Segmenter also uses $OLDPASS to store the prepared program segments, overwriting any existing temporary file of the same name.

EXAMPLE

To compile, prepare, and execute a FORTRAN/77 program entered from your standard input device, with the program listing sent to your standard list device, enter:

```
:FTNGO
```

To compile, prepare, and execute a FORTRAN/77 program from the disc file FORTSRC and send the program listing to the file LISTFILE, enter:

```
:FTNGO FORTSRC,LISTFILE
```

ADDITIONAL DISCUSSION

FORTRAN/77 Reference Manual (5957-4685)
MPE Segmenter Reference Manual (30000-90011)

:FTNPREP

Compiles and prepares a FORTRAN/77 program. FORTRAN/77 is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:FTNPREP [textfile][,[progfile][,listfile]]
   [;INFO=quotedstring]
```

PARAMETERS

textfile

Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is FTNTEXT. Default is $STDIN.

progfile

Actual file designator of the program file to which the prepared program segments are written. You will get the best results by omitting the *progfile* parameter. When you omit *progfile*, the MPE Segmenter creates the program file, which will be stored in the temporary file domain as $OLDPASS. If you do create your own program file, you must do so in one of two ways:

- By using the MPE :BUILD command and specifying a file code of 1029, or PROG, and a *numextents* value of 1. This file is then used by the :PREP command.
- By specifying a nonexistent file in the *progfile* parameter, in which case a job/session temporary file of the correct size and type is created.

listfile

Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is FTNLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (FTNTEXT, FTNPROG, and FTNLIST) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

quotedstring

A sequence of characters between two single quotation marks (apostrophes) or between two double quotation marks. You may use the delimiter as part of the string so long as the delimiter appears twice. Any occurrence of two single quotation marks in a row, or two double quotes in a row, is considered part of the string, and, therefore, not the terminating delimiter.

INFO=*quotedstring* is used in the FORTRAN/77 programming language to pass initial compiler options to a program. PASCAL/3000 brackets the *quotedstring* with dollar signs and places it before the first line of source code in the text file. For more information, refer to the :RUN command in this section.

USE

This command may be issued from a Session or Job. It may not be used from a Program, or in BREAK. It is Breakable (suspends execution).

OPERATION

The :FTNPREP command compiles and prepares a FORTRAN/77 program into a program file on disc. If you do not specify *textfile*, MPE expects input from the current input device. If you do not specify *listfile*, MPE sends the listing output to the formal file designator FTNLIST (default $STDLIST). The USL file $OLDPASS, created during compilation, is a temporary file passed directly to the Segmenter. You may access it only if you do not use the default for *progfile*. This is because the Segmenter also uses $OLDPASS to store the prepared program segments, overwriting any existing temporary file of the same name.

EXAMPLES

The following example compiles and prepares a FORTRAN/77 program entered through your standard input device and stores the prepared program segments in the file $OLDPASS. The listing will be printed on your standard list device.

```
:FTNPREP
```

To compile and prepare a FORTRAN/77 source program from the source file FORTSRC, storing it in FORTPROG, and sending the listing to your standard list device, enter:

```
:FTNPREP FORTSRC,FORTPROG
```

ADDITIONAL DISCUSSION

FORTRAN 77 Reference Manual (5957-4685)
MPE Segmenter Reference Manual (30000-90011)

:FULLBACKUP

Performs a backup of MPE and all of the files on the system.

SYNTAX

```
:FULLBACKUP [dumptape][,auxlistfile]
```

PARAMETERS

dumptape Specifies the magnetic tape destination for the backup. It *dumptape* is specified, it must be backreferenced to a previous :FILE command or the file equates to the formal file designator DUMPTAPE. The default is TAPE. If there is not a device class TAPE, the device class CTAPE will be used.

auxlistfile Actual file designator of the output file (device) to which all listings requested during the execution of the backup process are written. The default is $STDLIST.

USE

This command may be issued from a Session, Job, Program, but not in BREAK. It is Breakable (suspends execution). It requires System Supervisor (OP) capability.

OPERATION

This command starts the backup process and copies MPE and all of the files on the system. The two commands :PARTBACKUP and :FULLBACKUP provide an alternative to the :SYSDUMP dialog. The only user interaction required with these commands is mounting tapes or serial discs and replying to any mount request. These replies are unnecessary, however, if the devices are configured as AUTO REPLY.

During the backup, FULLBACKUP will display messages at one minute intervals to report its progress:

```
STORE OPERATION IS 27% COMPLETE
```

EXAMPLES

In the following example the first :FILE command defines DUMP as a magnetic tape file. The second :FILE command defines LIST as a line printer. The TAPE and LP parameters are device class names arbitrarily defined during the previous system configuration.

```
:FILE DUMP;DEV=TAPE
:FILE LIST;DEV=LP
:FULLBACKUP *DUMP,*LIST
```

Establishes a logging identifier on the system.

SYNTAX

```
                                    {,DISC }
:GETLOG logid,LOG=logfile[          {,TAPE }]
                                    {,SDISC}
                                    {,CTAPE}

  [,PASS=password]

  [{,AUTO  }]
  [{,NOAUTO}]
```

PARAMETERS

logid	The logging identifier to be established. This must contain from one to eight alphanumeric characters beginning with an alphabetic character.
logfile	The name of the file to receive data from the logging procedure. It must contain from one to eight alphanumeric characters, beginning with an alphabetic character. You must also specify the device class on which the *logfile* resides, i.e. DISC, TAPE, SDISC, or CTAPE.
password	Logging identifier password, optionally assigned by the creator for protection against illegal use of a particular identifier. The password must contain from one to eight alphanumeric characters, beginning with an alphabetic character.
AUTO	Initiates an automatice :CHANGELOG if the log file becomes full. Refer to :CHANGELOG will not be performed if the log file becomes full.
NOAUTO	Prevents initiation of an automatic :CHANGELOG. A :CHANGELOG will not be performed if the log file becomes full.

USE

This command may be issued form a Session, Job, Program, or in BREAK. It is not Breakable. It requires User Logging (LG) capability.

OPERATION

The :GETLOG command specifies a logging identifier to be used each time a particular logging process is used. Frequently the :GETLOG command is used with data bases, so that each test task runs will write to a logging file. Therefore, data recovery is easier because you know where the task failed.

The creator of the logging identifier must have User Logging (LG) capability to execute this command. Other users can be allowed access to this logging identifier by notifying them of the identifier and password. Note that if a password is specified, it will be required whenever the logging process is accessed. Users accessing the logging system with this identifier must supply the identifier and password in the OPENLOG intrinsic.

To use the AUTO parameter, the log process for *logid* must be enabled for changing. You may do this by ending the log file name with the numeric characters "001" (e.g. *fname001*). This naming convention works in conjuction with the file set number to generate sequential filenames automatically.

NOTE

If a new log filename is specified with the :ALTLOG command, the links with any previous log file will be broken.

There cannot be two logging identifiers with the same name on the system at the same time. The :LISTLOG command can be used to find out what logging identifiers currently exist.

EXAMPLE

To create the logging identifier FINANCE and associate it with the disc log file A, enter:

```
:GETLOG FINANCE;LOG=A,DISC
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)
MPE V Intrinsics Reference Manual (32033-90007)

:GETRIN

Acquires a global Resource Identification Number (RIN) and assigns a password to it.

SYNTAX

```
:GETRIN rinpassword
```

PARAMETERS

rinpassword	Password of the intrinsic that locks the RIN. The password must contain from one to eight alphanumeric characters, beginning with an alphabetic character.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

The :GETRIN command acquires a global RIN from the MPE RIN pool, typically during a session. You must assign an arbitrary password for the RIN, which aids in restricting its use to authorized users. You can then give this RIN and the associated password to cooperating users so that it can be locked and unlocked by them. For instructions on how to lock and unlock a RIN, and how to pass a RIN and its password as intrinsic parameters, refer to the MPE V Intrinsics Reference Manual (32033-90007).

Users who know the RIN and its password can use it in their programs (in jobs or sessions) until the user who acquired the RIN releases it with the :FREERIN command. The RIN acquired is always a unique, positive integer. The total number of RINs MPE can allocate is specified when the system is configured, but cannot exceed 1024. If all currently available RINs have been acquired by other users, MPE will reject your request and issue the message:

```
RIN TABLE FULL
```

In this case, you must wait until one of the RINs becomes available, or request that your System Manager raise the maximum number of RINs that can be assigned.

EXAMPLE

To acquire a global RIN and assign to it the password MYRIN, enter:

```
:GETRIN MYRIN
```

MPE responds with the RIN number assigned, for example:

```
RIN: 1
```

ADDITIONAL DISCUSSION

MPE V Intrinsics Reference Manual (32033-90007)

Assigns a downed device to diagnostics.

SYNTAX

```
:GIVE ldev
```

PARAMETERS

ldev	The logical device number of the downed device.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It is executable only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

This command gives control of the device specified by *ldev* to diagnostics so that no other process can access the device while diagnostic tests are in progress. Before the device can be given to diagnostics, it must first be taken offline with the :DOWN command, as shown in the example below.

EXAMPLES

To assign a downed device with logical device number 81 to diagnostics, enter:

```
:DOWN 81
:GIVE 81
11:20/3/LDEV#81 IN USE BY DIAGNOSTICS
```

To assign a spooled job accepting card reader to diagnostics, enter:

```
:STOPSPOOL 5                          ** Stops spooling
                                         process for ldev 5. **
11:16/31SP#5/STOPPED
11:16/31/LDEV#5 NOT READY

:REFUSE 5                             ** Prevents jobs and
                                         sessions on the device. **

:ABORTIO 5                            ** Aborts pending I/O
                                         for the device. **
11:16/3/LDEV#5 NO I/O PENDING

:DOWN 5                               ** Takes ldev
                                         5 offline. **

:GIVE 5                               ** Gives ldev
                                         5 to diagnostics. **

11:16/3/LDEV#5 IN USE BY DIAGNOSTICS
```

:HEADOFF

Stops header/trailer output to a device.

SYNTAX

```
:HEADOFF ldev
```

PARAMETERS

ldev	The logical device number of the printer or card punch affected by the command.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be executed only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

Header and trailer information is the data which appears before and after a file, but which is not part of the file's text. It identifies the file by session number, output spoolfile number, session name (if any), user and account. It also lists the date and time the file was printed.

When the header/trailer facility is enabled and output is directed to a card punch, MPE automatically punches a header card and a trailer card identifying the job. If output is directed to a line printer, MPE automatically prints header and trailer pages identifying the job that produced the file.

If the device is in use and a header has already been printed when you issue the :HEADOFF command, your request to suppress header/trailer output will take effect after the corresponding trailer is printed.

EXAMPLE

To stop header/trailer output to logical device number 6 , enter:

```
:HEADOFF 6
```

:HEADON

Resumes header/trailer output to a device.

SYNTAX

```
:HEADON ldev
```

PARAMETERS

ldev	The logical device number of the printer or card punch affected by the command.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be executed only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

Header and trailer information is the data which appears before and after a file, but which is not part of the file's text. It identifies the file by session number, output spoolfile number, session name (if any), user and account. It also lists the date and time the file was produced.

When the header/trailer facility is enabled and output is directed to a card punch, MPE automatically punches a header card and a trailer card identifying the job. If output is directed to a line printer, MPE automatically prints header and trailer pages identifying the job that produced the file.

If the device is in use, your request to resume header/trailer output will take effect after the the current output is complete.

The header/trailer facility is enabled on all cold loads.

EXAMPLE

To resume header/trailer output to logical device number 6 enter:

```
:HEADON 6
```

Initiates an interactive session.

SYNTAX

```
:HELLO [sessionname,]username[/userpass].acctname[/acctpass]
       [,groupname[/grouppass]]

   [;TERM={termtype }]
          {termname}

   [;TIME=cpusecs ]

         {BS}
   [;PRI={CS}]
         {DS}
         {ES}

   [{;INPRI=inputpriority}]
   [{;HIPRI               }]
```

PARAMETERS

sessionname — Arbitrary name used in conjunction with *username* and *acctname* parameters to form a fully qualified session identity. The name must contain from one to eight alphanumeric characters, beginning with an alphabetic character. Default is that no session name is assigned.

username — Username, established by the Account Manager that allows you to logon under this account. The name must contain from one to eight alphanumeric characters, beginning with an alphabetic character.

userpass — User password, optionally assigned by the Account Manager. The password must contain from one to eight alphanumeric characters, beginning with an alphabetic character.

acctname — Account name as established by the System Manager. The name must contain from one to eight alphanumeric characters, beginning with an alphabetic character. The *acctname* parameter must be preceded by a period (.).

acctpass — Account password, optionally assigned by the System Manager. The password must contain from one to eight alphanumeric characters, beginning with an alphabetic character. The *acctpass* must be preceded by a slash (/).

groupname — Group name to be used for the local file domain and the CPU time charges as established by the Account Manager. The name must contain from one to eight alphanumeric characters, beginning with an alphabetic character.

Default is your home group if you are assigned one by the Account Manager. (Required if a home group is not assigned.)

grouppass

Group password optionally assigned by the Account Manager. The password must contain from one to eight alphanumeric characters, beginning with an alphabetic character. The *grouppass* parameter is not needed when you logon to your home group. The *grouppass* parameter must be preceded by a slash (/).

termtype or *termname*

Determines terminal-type characteristics. The *termtype* parameter whose assigned value determines the type of terminal used for input. MPE uses this parameter to determine device-dependent characteristics such as delay factors for carriage returns. It must be a number from 0 to 22. For more information refer to Appendix A "TERMINALS SUPPORTED BY MPE". The default value for *termtype* is assigned by the System Supervisor during system configuration. This is a required parameter to ensure correct listings if your terminal is not the default *termtype.*

The *termname* parameter is the name of the file containing the desired terminal-type characteristics. The file cannot have a lockword or reside on a Private Volume.

Users of the Workstation Configurator are allowed to create terminal-type files. The proper and efficient operation of a specific device by a user created terminal-type is the responsibility of the user. The Workstation Configurator utility allows the user to specify the following characteristics of your terminal: data flow control, block mode, read trigger, special characteristics, echo, line feed, parity, and printer control.

If group and/or account names are omitted, the proposed logon group and/or account name is substituted. For further information on the terminal-type characteristics, refer to the Fundamental Communications Handbook (5957-4634) and the Workstation Configurator Reference Manual (30000-90001).

cpusecs

Maximum CPU time that session can use, entered in seconds. When the limit is reached, session is aborted. It must be a value from 1 to 32767, but cannot exceed any limit imposed by the System or Account Manager. To specify no limit, enter a question mark (?), UNLIM , or omit the parameter. Default is no limit.

BS, CS, DS, or ES

The execution priority queue that the Command Interpreter uses for your session, and also the default priority for all programs executed within the session. BS is highest priority, ES is lowest. If you specify a priority that exceeds the highest permitted for your account or username by the system, MPE assigns the highest priority possible below BS. DS and ES are intended primarily for batch jobs; their use for sessions is generally discouraged. Care should be use in assigning the BS queue because processes in this priority class may lock out other processes. For information on the guidelines for these priority queues, refer to the :TUNE command in the MPE V System Operation and Resource Management Reference Manual (32033-90005). Default is CS.

inputpriority or HIPRI

Determines the input priority of the job or session. The *inputpriority* option is the relative input priority used in checking against access restrictions imposed by the job fence. The *inputpriority* option takes effect at logon time and must be from 1 (lowest priority) to 13 (highest priority). If you supply a value less than or equal to current job fence set by System Operator, session is denied access. Default is 8.

The HIPRI option is used for two different purposes when logging on. It can be used to override the system job fence or it can be used to override the session limit. When using the HIPRI option to override the job fence, the system will first check to see if you have System Manager (SM) or System Operator (OP) capability. If you have either of these capabilities, you will be logged on and your INPRI will default to the system's job fence and execution limit. If you do not have either of these capabilities, the system will attempt to log you on using INPRI=13 and will succeed if the job fence is 12 or less, and if the session limit is not exceeded. In attempting to override the session limit (to logon after the maximum number of sessions set by the operator has been reached), you can specify HIPRI, but to do so you must have either SM or OP capability. The system will not override the the session limit automatically. Use of the HIPRI option without SM or OP capability causes the following warning to be displayed:

```
MUST HAVE 'SM' OR 'OP' CAP. TO SPECIFY HIPRI,
 MAXIMUM INPRI OF 13 IS USED (CIWARN 1460)
```

USE

This command may be issued from a Session. It may not be used from a Job, Program, or in BREAK. It is Breakable.*

* Pressing [BREAK] during the output of the logon message terminates the rest of the message. You will remain logged on, and MPE will prompt you for your next command, provided you do not have an OPTION LOGON UDC. If there is an OPTION LOGON UDC that is not a program, the UDC is not BREAKable and is forced at logon time.

OPERATION

The :HELLO command initiates an interactive session and must be entered from a terminal; no other device can be used for this command. You must supply both a valid username and account name in your logon command. Otherwise, MPE rejects your logon attempt and prints an error message to that effect. If your logon attempt is accepted, however, MPE verifies this by printing specific logon information and prompting you for your next MPE command. This information appears in the following example, where a user has logged on under the username MAC and the account name TECHPUBS:

```
:HELLO MAC.TECHPUBS
HP3000 / MPE V  G.02.00 (BASE G.02.00).  MON, MAY 14, 1985,  8:22 AM
:                                          ** MPE prompts for next command. **
```

Sometimes a welcome message from the System Operator will appear following the MPE verification of your logon.

The session number assigned by MPE uniquely identifies your session to MPE and to other users. MPE assigns such numbers to sessions in sequential order as they are logged on. If you are on a modem and do not logon within the system configured time, the line is dropped. You must redial and press RETURN again. If you are already logged on and you issue the :HELLO command, you will be logged off your current session and logged onto a new session except in the following cases:

- Spaces precede the command.
- The command is executed using a :REDO command.
- The command is issued in a user defined command (UDC).

In certain instances, you may be required to furnish information in addition to the user and account names in your :HELLO command or :( ) COMMAND LOGON. This information includes:

- Group Name.
- One or more passwords.
- Terminal type code.

Group Name

The group you select at logon for your local file domain is known as your logon group. If your Account Manager has associated a home group with your username, and if you desire this group as a logon group, you need not specify this. MPE automatically assigns the home group as your logon group when you logon. But if you desire to use some other group as your logon group, you must specify that group's name in your logon command in this way:

```
:HELLO MAC.TECHPUBS,YGROUP
                    ↑
            ** Group name **
```

If your username is not related to a home group, you must enter a group name in your :HELLO command or :( ) COMMAND LOGON, or your logon attempt will be rejected.

Once you logon, if the normal (default) file security provisions of MPE are in force, you have unlimited access to all files in your logon and home groups. Furthermore, you can read files and execute programs stored in the PUB (public) group of your account and the PUB (public) group of the SYS (system) account. You cannot, however, access any other files in any way. Further information about files and file security can be found in the MPE File System Reference Manual (30000-90236).

Passwords

To enhance the security of an account, and to prevent unauthorized accumulation of charges against the account, the System Manager may assign a password. Similarly, an Account Manager may associate passwords with the usernames and groups belonging to his account. If you are using an account, username, or group (other than your home group) that has a password, you must furnish that password when you logon. Include the password after the name of the protected entity, separated from that name by a slash mark (/). (In MPE, slash marks denote security.)

For instance, if the group XGROUP requires a password and if you wish this group as your logon group, you could enter the password in this fashion:

```
                          ** Group password **
                                    ↓
:HELLO MAC.TECHPUBS,XGROUP/XPASS
```

Note that when you specify your home group as your logon group, you need not enter a password, even if that group has such a password.

Sometimes, when logging onto the system, it is more convenient to have MPE prompt you for any required passwords. You do this by omitting the passwords from the logon command. When you logon, the command is printed in the normal way; MPE prompts you for the password, then turns the echo off so that the password will not be printed. If you enter the password incorrectly, the prompt will reappear and you will have two more chances to enter the password correctly. After the third incorrect entry, the line will be dropped, and you must press RETURN to receive another prompt. Echo is turned on after all passwords are read.

Terminal Types

MPE must be able to determine certain characteristics about your terminal, such as input and output speed, in order to conduct a session. If you logon using a different type of terminal than the type the System Manager has configured, you must specify your terminal type when you logon. For example, if you are logging on at an HP 2621A, but the default type at your site is a HP 2645A you would enter:

```
:HELLO MAC.TECHPUBS;TERM=10
                        ↑
                  ** Terminal type **
```

Terminal types may also be specified as a file containing terminal-type characteristics. Users of the Workstation Configurator are allowed to create terminal type files without any restrictions. The proper and efficient operation of a specific device by a user-created terminal type is the responsibility of the user. For example, a terminal type that specifies no flow control mechanism will generally create data overruns on the device. Users may control each of the following terminal-type characteristics: echo, line feed during input, form feed during output, backspace response, printer control, and parity. Refer to the Workstation Configurator Reference Manual (30239-90001) and the Fundamental Communications Handbook (5957-4634) for more information. Refer also to Appendix A, "Terminals Supported by MPE", in Section III of this manual.

EXAMPLE

To start a session named ALPHA, with the user MAC, the account TECHPUBS, the group XGROUP, and the grouppass word XPASS, enter:

```
:HELLO ALPHA,MAC.TECHPUBS,XGROUP/XPASS
HP3000 / MPE V  G.02.00 (BASE G.02.00).  MON, MAY 14, 1985,  8:22 AM
:                                  ** MPE then prompts for next command. **
```

Accesses the HELP subsystem.

SYNTAX

```
:HELP
   [HELP            ]
   [tablecontents   ]
   [command[keyword]]
   [ALL             ]
   [EXIT            ]
```

PARAMETERS

HELP — Displays information for the :HELP command. This is treated the same as entering HELP with any other command, because :HELP is a command.

tablecontents — Any of several items for which information may be obtained. These items are:

```
SESSIONS
JOBS
PROGRAMS
FILES
MANAGE
OPERATOR
SPOOLER
UTILITY
```

If you want information on running sessions, you would enter:

```
:HELP SESSIONS
```

command — Any MPE command. MPE displays the command name and syntax. In addition, a list of keywords for that command is displayed.

Any UDC can also be the *command*. In this case MPE will list out the lines of the UDC from the UDC catalog file and reference the UDC as a "user defined command".

keyword One of the keywords described under the command parameter. All commands have the following keywords:

PARMS	Lists all parameters of the specified command.
OPERATION	Describes the use of the specified command.
EXAMPLE	Displays an example showing usage of the specified command.
ALL	Displays all parameters, operation, and an example of the command.

The :FILE command has the following additional keywords:

NAME
DEVICE
ACCESS
DISP

Entering the :HELP command, followed by a command name and the ALL parameter causes MPE to display all information for that command (syntax, parameters, operation, and example). The keyword parameter can be entered with a command, as in :HELP HELLO PARMS. In this case, PARMS is a keyword and must be separated from the command with a space or comma. The keyword parameter also can be entered alone, if you are running the HELP subsystem in subsystem mode, in response to the (>) prompt.

ALL Displays entire table of contents and the contents of each keyword for the :HELP command.

EXIT Exits the HELP subsystem.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is Breakable (aborts execution).

OPERATION

The :HELP command accesses the HELP subsystem. If entered with no parameters, as in

```
:HELP
```

HELP enters the subsystem mode, displays a table of contents and greater-than (>) prompt, and awaits your input. Entering any table of contents item such as SESSIONS produces a listing of all the the commands used in running sessions. Entering any command name produces the syntax for that command and a list of the keywords. Entering a keyword such as PARMS produces a listing of all the items for that keyword (all parameters in this case).

Entering E or EXIT or pressing (BREAK) terminates the HELP subsystem. Entering Y^C terminates the display, and returns you to a system prompt. Entering S^C temporarily stops the display, and entering Q^C resumes the display (useful if the screen is full as when using a CRT terminal). Pressing (RETURN) in subsystem mode causes MPE to display information up to the next keyword or command. For example, after you enter STORE with the :HELP command, MPE displays STORE syntax, the keyword list (PARMS, OPERATION, EXAMPLE) and provides a prompt (>). Pressing (RETURN) again causes MPE to display all parameter information for the :STORE command. Pressing (RETURN) again at the next prompt causes MPE to display all OPERATION information for the :STORE command. (This is similar to turning the pages of a manual.)

Entering :HELP with a parameter causes HELP to enter the immediate mode. Information pertaining to that parameter is displayed immediately. For example:

```
:HELP ABORT
```

Causes HELP to display:

```
    :ABORT

Aborts current program or operation.

SYNTAX
    :ABORT

KEYWORDS: PARMS,OPERATION,EXAMPLE
:
```

EXAMPLE

To obtain information concerning MPE utility functions, enter:

```
:HELP UTILITY
```

MPE displays:

```
Utility Functions.  Following are the commands used:
   CONSOLE        SHOWALLOW
   PTAPE          SHOWCATALOG
   SETCATALOG     SHOWME
   SETMSG         SHOWTIME
   SPEED          TELL
   RECALL         TELLOP
   REDO

You can use any command name as a keyword.


KEYWORDS: SESSIONS,JOBS,PROGRAMS,FILES,MANAGE,OPERATOR,SPOOLER,
          UTILITY
:
```

:IF

Used to control the execution sequence of a job.

SYNTAX

```
:IF [(]logexpr[)] THEN
```

PARAMETERS

logexpr Logical expression, consisting of the operands and relational operators. The relational operators allowed are:

> greater than
< less than
>= greater than or equal
<= less than or equal
= equal
<> not equal

The allowed operands are: JCW or CIERROR; any Job Control Word values saved in a name with the :SETJCW command; OK, WARN, FATAL, or SYSTEM (as defined under the :SETJCW command), or integer constants.

Compound logical expressions can be formed using the AND and OR logical operators, and nested within parentheses.

USE

This command may be issued from a Session, Job, or in BREAK. It may not be used from a Program. It is not Breakable.

OPERATION

This command begins an IF block consisting of all the commands after the :IF command up to, but not including, the next :ELSE or :ENDIF statement. The :ELSE or :ENDIF must have the same nesting level as the IF statement. Another similar block can follow the :ELSE statement. Nesting of the blocks is allowed to 15 levels. The :ENDIF statement ends the IF block.

The logical expression is evaluated and, if the expression evaluates to true, the IF block is executed; if false, the ELSE block (if one exists) is executed.

EXAMPLE

The following job listing illustrates the use of an :IF statement with :ELSE and :ENDIF statements:

```
!CONTINUE
!SPL MYPROG,MYUSL
!IF JCW>=FATAL THEN
!   TELL MAC.TECHPUBS;COMPILE FAILED
!ELSE
!   TELL MAC.TECHPUBS;COMPILE COMPLETED
!ENDIF
```

:JOB

Initiates a batch job.

SYNTAX

```
:JOB [jobname,]username[/userpass].acctname
   [/acctpass][,groupname[/grouppass]]

   [;TIME=cpusecs]

         {BS}
   [;PRI={CS} ]
         {DS}
         {ES}

   [{;INPRI=inputpriority}]
   [{;HIPRI               }]

   [;RESTART]

   [;OUTCLASS=[device][,outputpriority[,numcopies]]]

   [;TERM={termtype}]
          {termname}
```

PARAMETERS

jobname — Arbitrary name used with *username* and *acctname* parameters to form a job identity. The name must contain from one to eight alphanumeric characters, beginning with an alphabetic character. Default is that no job name is assigned.

username — Username, established by the Account Manager, that allows you to logon under this account. The name must contain from one to eight alphanumeric characters, beginning with an alphabetic character.

userpass — User password, optionally assigned by Account Manager. The password must contain from one to eight alphanumeric characters, beginning with an alphabetic character.

acctname — Account name as established by the System Manager. The name must contain from one to eight alphanumeric characters, beginning with an alphabetic character. The *acctname* parameter must be preceded by a period (.).

acctpass — Account password, optionally assigned by the System Manager. The password must contain from one to eight alphanumeric characters, beginning with a alphabetic character. The *acctpass* parameter must be preceded by a slash (/).

groupname — Group name to be used for the local file domain and for CPU time charges, as established by Account Manager. The name must contain from one to eight alphanumeric characters, beginning with an alphabetic character. Default is home group if one is assigned. (Required if a home group is not assigned.)

grouppass — Group password, optionally assigned by Account Manager. The password must contain from one to eight alphanumeric characters, beginning with an alphabetic character. The *grouppass* parameter is not needed when you logon to your home group. The *grouppass* parameter must by preceded by a slash (/).

cpusecs — Maximum CPU time allowed job, in seconds. When this limit is reached, job is aborted. This must be value from 1 to 32767. To specify no limit, enter a question mark or UNLIM, or omit this parameter. Default is system configured job limit.

BS, CS, DS, or ES — The execution priority queue that the Command Interpreter uses for your session. This is also the default priority for all programs executed within the session. BS is highest priority, ES is lowest. If you specify a priority that exceeds the highest permitted for your account or username by the system, MPE assigns the highest priority possible below BS. DS and ES are intended primarily for batch jobs; their use for sessions is generally discouraged.

CAUTION

Use care in assigning the BS queue. Processes in this priority class can lock out other processes.

For information on the guidelines for these priority queues, refer to the :TUNE command in the MPE V System Operation and Resource Management Reference Manual (32033-90005). Default is CS.

inputpriority or HIPRI — Determines the input priority of the job or session. The *inputpriority* option is the relative input priority used in checking against access restrictions imposed by the job fence. The *inputpriority* option takes effect at logon time and must be from 1 (lowest priority) to 13 (highest priority). If you supply a value less than or equal to current job fence set by System Operator, session is denied access. Default is 8.

HIPRI is the request for maximum input priority, causing job to be scheduled regardless of current job fence or execution limit for jobs. This parameter overrides the *inputpriority*. You can specify this parameter only if you have System Manager or System Supervisor capability. If not, the system tries to log you on with INPRI=13.

RESTART — Request to restart a spooled job that has been interrupted by system termination/restart. This parameter takes effect automatically when system is subsequently restarted with the WARMSTART option.

This parameter applies only to jobs initiated on spooled input devices. It is ignored for other jobs. Default is that spooled jobs are not restarted after system termination/restart.

device Class name or logical device number (*ldev*) of device to receive listing output. You cannot specify a magnetic tape unit. If the parameter is not a valid *ldev* or class name an error will be generated. Default is defined in the system configuration.

NOTE

Nonsharable device (ND) file access capability is required in order to use this parameter.

outputpriority The output priority for job list file, if destined for spooled line printer or card punch. This parameter is used to select next spooled devicefile (on disc) for output, among all those contending for a specific printer or punch. Must be a value from 1 (lowest priority) to 13 (highest priority). Note that when the *outputpriority* is 1, output is always deferred. Thus, you should use an *outputpriority* of 2 or greater if you wish output printed or punched from disc.

This parameter applies only to output destined for spooled output devices, and is ignored for other output. Default is 8.

numcopies Number of copies of job listing to be produced. This parameter applies only when listing is directed to a spooled device, and is ignored in other cases. If the number of copies is less than 1 a warning will be issued. The command will still execute with the default value of 1. If it is greater than 127, an error message will be printed and 127 copies will be printed. Default is 1.

termtype or *termname* Determines the terminal-type characteristics. The *termtype* parameter determines the type of terminal used for input. MPE uses this parameter to determine device dependent characteristics such as delay factors for carriage returns. It must be a number from 0 to 22. Refer to Appendix A "TERMINALS SUPPORTED BY MPE V" for a list of terminal-type numbers. The default value for *termtype* is assigned by the System Supervisor during system configuration. This is a required parameter to ensure correct listings if your terminal is not the default *termtype*.

The *termname* parameter is the name of the file containing the desired terminal-type characteristics. The file may not have a lockword or reside on a Private Volume.

Users of the Workstation Configurator are allowed to create terminal-type files. The proper and efficient operation of a specific device by a user-created terminal-type is the responsibility of the user. The Workstation Configurator utility allows the user to specify the following characteristics of your terminal: data flow control, block mode, read trigger, special characteristics, echo, line feed, parity, and printer control.

If group or account names are omitted, the proposed logon group and/or account name is substituted. For further information on the terminal-type characteristics, refer to the Fundamental Communications Handbook (5957-4634) and the Workstation Configurator Reference Manual (30000-90001).

USE

This command may be issued from a Session or Job. It may be used from a Program or in BREAK. It is not Breakable.

OPERATION

Initiates a batch job by establishing contact with MPE. If the job is initiated through a session and is in WAIT state, your terminal is locked until the job enters EXEC state. When MPE begins the job, it displays the following information on the list device:

- Job Number, as assigned by MPE, to identify the job.
- Date and time.
- "HP 3000", and the modified and base MPE *version.update.fix* numbers.

In the :JOB command, as in the :HELLO command, you must always include your username and account name, which you obtain from your Account Manager. If you omit either of these names, or enter them incorrectly, MPE rejects your job and prints an error message on the standard listing device. If your job is accepted, MPE begins job processing. If the job is entered through a spooled input device, the job is copied to a disc file and initiated from that file rather than the originating device.

If the standard listing file is a line printer, MPE prints a a header page prior to listing the :JOB command. (The System Operator can disable the printing of this page via the console command :HEADOFF.)

The job number assigned by MPE always uniquely identifies your job to MPE and other users. MPE assigns such numbers in sequential order as jobs are accepted.

Sometimes, the job acceptance information includes a message from the System Operator following the standard display. When present, this is the same message output in the logon information for sessions.

The minimum information needed for job initiation is the user and account name. Under certain circumstances, however, you may need to furnish the following additional information in your :JOB command:

- File Group Name.
- User, Account, and/or Group Passwords.

The cases in which this information is required, and the rules for supplying it, are the same as those for the :HELLO command for sessions, except that:

- When you omit a required password from the :JOB command, MPE rejects your access attempt without prompting you for the password.

- When you enter the :JOB command through a device other than a terminal and the standard input device is different from the standard listing device, MPE does not echo passwords.

- When the standard listing device is a line printer, and you do not specify a file group name, central processor time limit, execution priority, and/or input priority in the :JOB command, the default values assigned by MPE for the omitted parameters appear on the job listing.

A job will execute user defined commands (UDCs) whether the job is interactive or streamed. UDCs will also execute in any subsystem or in MPE itself. You or your stream files must anticipate actions, prompts, or questions incorporated in any UDC you use, whether you use it interactively or in a streamed job.

The :EOJ command is used to terminate or end a job. The only alternative exits from a job are an error flush or an ABORT. An error flush stops the completion of the job at the point the error occurred.

EXAMPLE

To start a job named BETA, with the *username* MAC, *acctname* TECHPUBS, *groupname* XGROUP, and *group-pass* XPASS, enter:

```
:JOB BETA,MAC.TECHPUBS,XGROUP/XPASS

JOB NUMBER = #J1
MON, MAY 14, 1985, 8:22 AM
HP3000 / MPE V  G.02.00 (BASE G.02.00)
:
```

ADDITIONAL DISCUSSION

MPE File System Reference Manual (30000-90236)
MPE V Intrinsics Reference Manual (32033-90007)

:JOBFENCE

Defines the minimum input priority that a job or session must have in order to execute.

SYNTAX

```
:JOBFENCE priorityfence
```

PARAMETERS

priorityfence A number between 0 and 14, inclusive. Within this range, smaller numbers are less limiting; larger numbers more limiting.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

MPE will not dispatch jobs or sessions with an input priority less than or equal to the *priorityfence* until their input priority is raised with the :ALTJOB command, or until the JOBFENCE is lowered. System Managers and System Supervisors may override the JOBFENCE setting by logging on with the ;HIPRI parameter of the :JOB or :HELLO commands. Or, they may logon with an input priority greater than the JOBFENCE as reported by the :SHOWJOB command.

EXAMPLES

To defer all non-HIPRI jobs and sessions, first set the JOBFENCE to 14, as shown below.

```
:JOBFENCE 14
16:18/#J7/34/DEFERRED JOB INTRODUCED ON LDEV #10
16:18/#J8/35/DEFERRED JOB INTRODUCED ON LDEV #10
```

Then enter the :SHOWJOB command to display the effect of the new JOBFENCE.

```
:SHOWJOB

JOBNUM   STATE   IPRI   JIN   JLIST    INTRODUCED   JOB NAME

#S26     EXEC           20    20       THU  4:17P   OPERATOR.SYS
#J7      WAIT    D 8    10S   12       THU  4:18P   JOB1,FIELD.SUPPORT
#J8      WAIT    D 8    10S   12       THU  4:18P   JOB2,FIELD.SUPPORT
```

```
3 JOBS:
   0 INTRO
   2 WAIT; INCL 2 DEFERRED
   1 EXEC; INCL 1 SESSIONS
   0 SUSP

JOBFENCE= 14; JLIMIT= 5; SLIMIT=16
```

Finally, reset the JOBFENCE to 6, which will allow waiting jobs to logon.

```
:JOBFENCE 0

16:21/#J7/34/LOGON FOR: JOB1,FIELD.SUPPORT ON LDEV #10
16:21/#J8/35/LOGON FOR: JOB2,FIELD.SUPPORT ON LDEV #10
```

:JOBPRI

Sets or changes the default execution priority for batch jobs and sets a maximum execution priority for batch jobs.

SYNTAX

```
:JOBPRI [maxsubqueue][,defaultsubqueue]
```

PARAMETERS

maxsubqueue — The maximum priority at which batch jobs will be allowed to run. This overrides of any job priority a user may have requested with the :JOB command. This parameter may be ES, DS, CS, or 0. If 0 is specified, no limit is imposed on batch jobs. Default is no change in maximum priority.

defaultsubqueue — The default execution priority for batch jobs, which may be ES, DS, or CS. This takes effect if a user does not specify an execution priority in the :JOB command. Default is no change in execution priority.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. Users must have OP capability to execute this command.

OPERATION

The *maxsubqueue* parameter specified in the :JOBPRI command takes precedence over *defaultsubqueue*. Therefore, selecting a default parameter greater than the value of *maxsubqueue* parameter will not affect job execution: jobs will still be initiated with the maximum priority parameter.

EXAMPLE

To raise the maximum execution priority so that batch jobs can run in any subqueue requested, enter:

```
:JOBPRI 0
```

:JOBSECURITY

Designates what level of user may request resources and control the execution of jobs.

SYNTAX

```
:JOBSECURITY {HIGH}
             {LOW }
```

PARAMETERS

HIGH	Permits only the Operator logged on at the Console to use job control commands.
LOW	Allows individual users to exercise control over their own jobs.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be executed only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

:JOBSECURITY controls the use of the :ABORTJOB, :ALTJOB, BREAKJOB, and :RESUMEJOB commands. The HIGH parameter permits only the Operator to issue these commands. JOBSECURITY LOW allows any user to issue these commands for their own jobs; the job's username and account must match that of the user. Account Managers do not need a matching username, so they may control the execution of any job in their account.

EXAMPLE

To allow any user to abort, alter, BREAK or resume their own jobs, enter:

```
:JOBSECURITY LOW
```

:LDISMOUNT

Performs the logical dismount of a private volume set or class.

SYNTAX

```
:LDISMOUNT vcsname.group.account
```

PARAMETERS

vcsname	Volume class/set name for which you are requesting a logical dismount.
group.account	The group and account under which the volume set was created.

USE

This command may be issued from a Session, Job, Program or in BREAK. It is not Breakable. It may be executed only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

:LDISMOUNT is used to dismount Nonsystem Domain discs while the disc system is online. It informs MPE that the volume class/set is no longer needed and that member volumes should be logically dismounted, as a unit, from the drive(s) on which they reside. The dismount request will remain pending until any files being accessed on the disc volume are closed; at that point the command will execute. If the specified volume class/set is not mounted when you attempt to logically dismount it, you will receive an error message.

EXAMPLE

To dismount a private volume set named DATABASE, which was created in the PAYROLL group of the ACCTNG account, enter:

```
:LDISMOUNT DATABASE.PAYROLL.ACCTNG
```

ADDITIONAL DISCUSSION

Refer to the discussion of "PRIVATE VOLUMES" in Section of the MPE V System Operation and Resource Management Reference Manual (32033-90005).

:LIMIT

Limits the number of concurrently running jobs/sessions.

SYNTAX

```
        {numberjobs                    }
:LIMIT  {,numbersessions               }
        {numberjobs,numbersessions     }
```

PARAMETERS

numberjobs The number of jobs.

numbersessions The number of sessions.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

Maximum job and session limits are established by the System Supervisor during system configuration. Within these limits, the Operator may redefine the job and session limit with the :LIMIT command. When the system is restarted from disc in a WARMSTART, the Operator defined limits are retained. When any other startup option is used, the values configured by the Supervisor take effect.

At least one parameter must be included in the :LIMIT command. If you execute the :LIMIT command without parameters, the following message is displayed on the Console:

```
EXPECTED ONE OF JOB OR SESSION LIMIT. (CIERR 3071)
```

If you enter one parameter and omit the other, the limit of omitted parameter's remain unchanged.

No new jobs or sessions will be dispatched that would cause either of these limits to be exceeded, unless they are initiated with the ;HIPRI parameter of the :JOB or :HELLO commands. Non-HIPRI jobs can still be introduced when the limit is achieved, but they will not execute.

If you attempt to logon to a non-HIPRI session after the limit has been reached, you will receive the message:

```
CAN'T INITIATE NEW SESSIONS NOW
```

The specified limits may be exceeded at the time the command is issued. This will not cause jobs or sessions executing at the time to abort. They will continue to execute, but no new jobs will be allowed to enter the executing state, and no new sessions will be initiated.

EXAMPLES

To limit the number of jobs to 2 and the number of sessions to 15 , enter:

```
:LIMIT 2,15
:SHOWJOB

JOBNUM  STATE IPRI JIN  JLIST    INTRODUCED  JOB NAME

#S24    EXEC        20  20       TUE  1:54A  OPERATOR.SYS
#S26    EXEC       177  177      TUE  5:01A  CHEWY,RSPOOL.SYS
#S96    EXEC QUIET  35  35       TUE  8:31A  SLIDES.SIMON

3 JOBS:
    0 INTRO
    0 WAIT; INCL 0 DEFERRED
    3 EXEC; INCL 3 SESSIONS
    0 SUSP
JOBFENCE= 6; JLIMIT= 2; SLIMIT= 15
```

To limit the number of sessions to 13, but retain the current job limit, enter:

```
:LIMIT ,13
```

:LISTACCT

Lists the attributes for one account (Account Manager capability) or for any or all accounts (System Manager capability).

SYNTAX

```
:LISTACCT [{ @ }] [listfile]
          [{acct}]
```

PARAMETERS

@	Specifies all accounts. This is the default for users with SM capability, which is required to list all accounts.
acct	The name of the account whose attributes are to be listed. Account Managers may specify only their own account. Default is the logon account.

Note that the characters @, #, and ? can be used as "wildcard" characters, but will count toward the eight character limit. These "wildcard" characters have the following meanings:

@	Specifies zero or more alphanumeric characters and denotes all members of the set.
#	Specifies one numeric character.
?	Specifies one alphanumeric character.

The characters can be used as follows:

n@	List all accounts starting with the character "n".
@n	Lists all accounts ending with the character "n".
n@x	Lists all accounts starting with the character "n" and ending with the character "x ".
n##...#	Lists all accounts starting with the character "n" followed by up to seven digits.
?n@	Lists all accounts whose second character is "n ".
n?	List all two-character accounts starting with the character "n ".
?n	List all two-character accounts ending with the character "n ".

listfile	Destination of the attribute listing. The default is $STDLIST, but the output may be redirected with the :FILE command.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. Users must have System Manager (SM) or Account Manager (AM) capability to execute this command.

OPERATION

This command lists the attributes for one account (AM capability) or for one or all accounts (SM capability).

The listing appears as an octal dump of the account entry; each is headed by "A = " and the account name. When the dump contains alphanumeric data, a translation into ASCII appears to the right of each line. The following is a sample listing of an octal dump:

```
A  =  SYS
051531 051440 020040 020040 000006 000007 177607 000713 SYS.............
000000 000000 020040 020040 020040 020040 000010 052150 ..............Th
077777 177777 000010 113257 077777 177777 000037 147337 ................
077777 177777 004551 140036 000123 000217               .....i...S..
```

Each octal word in the preceding listing is displayed in the following decimal representation of the dump. The table on the next page decodes the information.

```
--00-- --01-- --02-- --03-- --04-- --05-- --06-- --07--
--08-- --09-- --10-- --11-- --12-- --13-- --14-- --15--
--16-- --17-- --18-- --19-- --20-- --21-- --22-- --23--
--24-- --25-- --26-- --27-- --28-- --29--
```

WORDS	CONTENT
0-3	Account name.
4-5	Account's group and user index pointers.
6-7	Account attributes.
8-9	Local attributes.
10-13	Password.
14-15	Permanent file space usage count (in sectors).
16-17	Permanent file space limit (in sectors).
18-19	Central processor time usage count (in seconds).
20-21	Central processor time limit (in seconds).
22-23	Connect-time count (in minutes).
24-25	Connect-time limit (in minutes).
26	Purge and account-security flags.
27	Maximum job/session priority (numerical).
28	Command file location of account UDCs.
29	Command file location of SYS account UDCs.

EXAMPLE

To list the attributes of the account, SYS ,enter:

```
:LISTACCT SYS
A  =  SYS
051531 051440 020040 020040 000006 000007 177607 000713 SYS.............
000000 000000 020040 020040 020040 020040 000010 052150 ..............Th
077777 177777 000010 113257 077777 177777 000037 147337 ................
077777 177777 004551 140036 000123 000217               .....i...S..
:
```

Displays all active :FILE equations for a job or session.

SYNTAX

```
LISTEQ [listfile]
```

PARAMETERS

listfile	The file to which output is directed. Default is $STDLIST.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is Breakable.

OPERATION

The :LISTEQ command displays all the active :FILE equations for a job or session.

EXAMPLE

An example of :LISTEQ is given below.

```
:LISTEQ


***FILE EQUATIONS

FILE TAPE1;DEV=ATAPE
FILE PP;ENV=LP2.ENV.OSE;DEV=EPOC
```

:LISTF

Lists descriptions of one or more permanent disc files.

SYNTAX

```
                    {,0 }
                    {,1 }
:LISTF [fileset][   {,2 }][,listfile]
                    {,-1}
```

PARAMETERS

fileset Specifies the set of files to be listed. This positional parameter has the form:

filename[.*groupname*[.*acctname*]]

The characters @, #, and ? can be used as "wildcard" characters in any position of the *fileset* parameter. The wildcard characters count toward the eight character limit for group, account, and filenames. These wildcard characters have the following meanings:

@	Specifies one or more alphanumeric characters. When used by itself, @ denotes "all members of the set".
#	Specifies one numeric character.
?	Specifies one alphanumeric character.

These characters can be used with files as follows:

n@	Lists all files starting with the character "n".
@n	Lists all files ending with the character "n".
n@x	Lists all files starting with the character "n" and ending with the character "x".
n#######	Lists all files starting with the character "n" followed by seven digits.
?n@	Lists all files whose second character is "n".
n?	Represents all two-character files starting with the character "n".
?n	Represents all two-character files ending with the character "n".

Note that depending upon the placement of these characters in the *fileset* parameter, they can also be used to indicate groups and accounts. If you specify nothing for the group or account, MPE assumes the logon group and account. Default is @ (lists all files in logon group).

0, 1, 2, or -1 — Displays information about the file. Displays only the filename. Default is 0 if no listlevel is specified.

1 — Displays the filename, file code, record size, format, the current end-of-file location, and the maximum number of records allowed in the file. It also shows whether the file is ASCII or binary; CCTL or NOCCTL; CIR, MSG, or STD).

2 — Displays all of the information found with the 1 option, plus the blocking factor, the number of disc sectors in use (including those in use for file labels and user headers), the number of extents currently allocated, and the maximum number of extents allowed.

-1 — Displays the octal listing of the file label. The first line of this listing is the directory entry for the file being listed. The remainder is the file label. The -1 option is available only to users who have System Manger (SM) or Account Manager (AM) capability.

listfile — The name of the output file to which the descriptions are written. It is automatically specified as an ASCII file with variable length records and these characteristics: closed in the temporary doamin, CCTL (user supplied carriage-control characters), OUT access mode, and EXC (exclusive access) option. The remainder of its characteristics are those obtained with the :FILE command default specifications. This file is temporary and cannot be overwritten by a :BUILD command. The default is $STDLIST.

CAUTION

When you specify 0, 1, 2, or -1 the System Directory may be locked for a brief time, so no other users will be able to access the System Directory.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is Breakable (aborts execution).

OPERATION

This command lists descriptions of one or more disc files at the level of detail you select. You need not have access to a file to list a description of it. However, a file description will not be listed unless the file's home volume set (PV) is mounted. A standard user may list level 0, 1, and 2 information for any file in the system. A user with Account Manager capability may list level -1 data for files in his own account. A user with System Manager capability may list -1 data for any file in the system. Doing a :LISTF to the line printer or magnetic tape will print the date and time at the top.

Note that this command applies only to permanent disc files.

You may request that the file information be displayed on devices other than the standard listing device. Name the desired device in the FILE command as follows:

```
:FILE PRTR;DEV=LP          ** Equates name PRTR with device class name LP. **
:LISTF @.@,2;*PRTR         ** Direct level 2 description of all files in all groups of
                              logon account to PRTR. **
```

If MPE fails to locate a requested file, the following error will be displayed:

```
NON-EXISTENT FILE (CIERR 907)
```

If you have a wildcard character in your request, and MPE cannot locate any files satisfying those conditions, the following warning is displayed:

```
NO FILES FOUND IN FILE-SET (CIWARN 431)
```

EXAMPLES

The following examples illustrate several uses of the :LISTF command.

Level 0 (Default) Output Format

```
:LISTF

FILENAME

APPB          APPC          APPD          APPE          APPF          CONMSG
CONOP1        CONOP4        CONOP5        CONOPEX       CONSG         EXAMPLES
GIMAGE        GIMCOM        GIMDOC        GIMDS         GIMPO         INDEX
K10130955     LINDA         LIST          LOGON         MEMO          MTS
MTS3000       OPFRONT       PS            REPLY         SCHED         VIP
```

Level 1 Output Format

```
:LISTF,1
ACCOUNT=  LEWIS         GROUP=  PUB

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

CONOP1               80B  FA           10          10


K0880909  TDPQ      104B  FA          636        2269
BOXES     DRAW      128W  FB           84          84
EXAMPLES            80B  FA           20          20
ERRORLOG             80B  FAO          12        1024
```

Level 2 Output Format

```
:LISTF,2
ACCOUNT=  USERS         GROUP=   BARBARA

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

APPB              88B   FA          414         414  16       162 14 14
APPM              80B   VAM         119         240  16        12  8  8
APPC              80B   FAO         127         208  16        70  8  8
APPK      KSAM    80B   FA           16         200  16        20  2  7
```

Level -1 Output Format

```
:LISTF UDCS, -1
F  =  UDCS
052504 041523 020040 020040 001001 120353                UDCS........
052504 041523 020040 020040 041517 046504 020040 020040 UDCS....COMD....
045440 020040 020040 020040 046501 051111 051523 040440 K.........TECH.
020040 020040 020040 020040 020202 004040 000001 123525 ..............U
123525 123525 000000 000000 000014 000000 000000 002163 .U.U..........s
000000 000000 120173 021427 000005 177650 002374 005015 ................
000024 000036 000000 002163 001001 120353 000400 104644 .......s........
001401 032743 001001 162503 001401 036363 001401 041324 ..5....C......B.
001001 164214 001401 042166 001002 001740 001401 044134 ......Dv......H.
001002 002656 001401 045323 001002 006740 000400 150014 ......J.........
000000 000000 000000 000000 000000 000000 000000 000000 ................
000000 000000 000000 000000 000000 000000 000000 000000 ................
000000 000000 000000 000000 000000 000000 000000 000000 ................
000000 000000 000000 000000 000000 000000 000000 000000 ................
000000 000000 000000 000000 004072 034411 123525 000000 ..........9..U..
000000 000000 000000 000000 000000 000000 004073 002010 ................
000000 000000 000000 000000 042111 051503 020001 000001 ........DISC....
```

The column headings have the following meanings:

HEADING	MEANING
FILENAME	Gives the filename. An asterisk (*) following the filename in the 1 and 2 option indicates the file is open for READ or WRITE access.
CODE	Refer to the :BUILD or :FILE command for an explanation of the file code
SIZE	The size of the records under the heading SIZE, indicating the number of words (W) or bytes (B).
TYP	The first column under this heading contains information concerning record format. This will be listed in one of the following ways: F (fixed length), V (variable length), or U (undefined length). The second column indicates whether the file is A (ASCII) or B (binary).

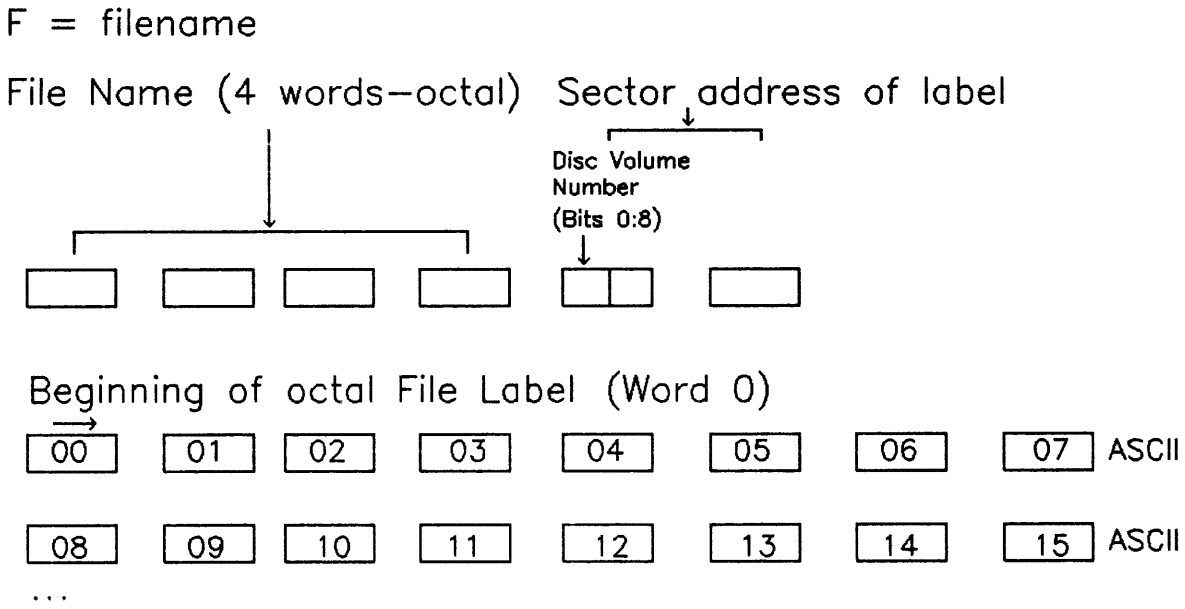
The third column indicates the kind of file. This information will appear in one of the following ways: Blank (Standard file unless KSAM appears in the code field.), O, (circular file), M (message file), or R (relative I/O file).

In the fourth column a "C" will appear only if the ;CCTL option is taken.

EOF	End of file location.
LIMIT	Maximum number of records allowed.
R/B	Blocking factor.
SECTORS	Sectors in use.
#X	Extents allocated.
MX	Maximum number of extents allowed.

A pictorial explanation of the level -1 output, follows:

:LISTF *filename*, -1



The disc file label contains the following:

WORDS		CONTENTS
0-3		Local filename.
4-7		Group name.
8-11		Account name.
12-15		Identity of file creator.
16-19		File lockword.
20-21		File security matrix.
22	(Bits 0:7)	Reserved for NLS.
	(Bits 8:13)	Not used.
	(Bits 14:1)	Store/Restore release bit.
	(Bits 15:1)	File secure bit: If 1, file secured. If 0, file released.
		File creation date.*
23		Last access date.*
24		Last modification date.*
25		File code.
26		Private Volume Information.
27	(Bits 0:1)	Class flag bit.
	(Bits 1:3)	Not used.
	(Bits 4:4)	Mounted volume table index.
	(Bits 8:8)	Volume mask.
28	(Bit 0:1)	Store BIT. (If on, :STORE or :RESTORE, in progress.)
	(Bit 1:1)	Restore Bit. (If on, :RESTORE in progress.)
	(Bit 2:1)	Load Bit. (If on, program file is loaded.)
	(Bit 3:1)	Exclusive Bit. (If on, file is opened with exclusive access.)
	(Bits 4:4)	Device subtype
	(Bits 8:6)	Device type.
	(Bits 14:1)	File is open for write.
	(Bits 15:1)	File is open for read
29	(Bits 0:8)	Number of user labels written.
	(Bits 8:8)	Number of user labels.
30-31		Maximum number of logical records.
32-33		File control block vector.
34		Checksum.
35		Coldload identity.
36		Foptions specifications.
37		Logical record size (in negative bytes).
38		Block size (in words).
39	(Bits 0:8)	Sector offset to data.
	(Bits 8:3)	Not used.
	(Bits 11:5)	Number of extents minus 1.
40		Logical size of last extent size in sectors.
41		Extent size.
42-43		Number of logical records in file.
44-107		Two-word addresses of up to 32 disc extents, beginning with address of first extents (words 44-45).
108-109		File allocation time.
110		File allocation date.
111		Not used.

112-113	Start of file block number.
114-115	Block number of last block.
116-117	Number of open and closed records.
124-127	Device class.

* Dates are in the same format as the value returned from the CALENDAR intrinsic:

Bits 0 6 7 15

Year of Century Day of Year

GENERAL EXAMPLES

To list all of the files in your logon group that have X as the second alphabetic character in their names, enter:

```
:LISTF ?X@

FILES FOR JASON.MANU,PEN

EXAMP
:
```

To list all of the three-character files in your logon group and account that end in the letter N, enter:

```
:LISTF ??N

FILES FOR JASON.MANU,PEN

SYN
:
```

:LISTFTEMP

Lists the descriptions of one or more temporary disc files for a job or session.

SYNTAX

```
:LISTFTEMP [fileset][;listlevel][;listfile]
```

PARAMETERS

fileset Specifies the set of files to be listed. This positional parameter takes the following form:

filename[.*groupname*[.*acctname*]]

The characters @, #, and ? may be used as "wildcard" characters in any position of the *fileset* parameter. The wildcard characters count toward the eight character limit for group, account, and filenames. These wildcard characters have the following meanings:

@	Specifies zero or more alphanumeric characters. When used by itself, @ denotes all members of the set.
#	Specifies one numeric character.
?	Specifies one alphanumeric character.

These characters can be used with files as follows:

n@	Lists all files starting with the character "n".
@n	Lists all files ending with the character "n".
n@x	Lists all files starting with the character "n" and ending with the character "x".
n#######	Lists all files starting with the character "n" followed by seven digits.
?n@	Lists all files whose second character is "n".
n?	Represents all two-character files that start with the character "n".
?n	Represents all two-character files that end with the character "n".

Depending upon their placement in the *fileset* parameter, these characters may also be used to indicate groups and accounts. If you specify nothing for the group or account, MPE assumes the logon group and account. Default is @ (all files in logon group).

listlevel Displays information about the file. The options are level 0, 1, 2, or -1:

0	Displays only the filename. Default is 0 if not listlevel is specified.
1	Displays the filename, file code, record size, format, the current end-of-file location, and the maximum number of records allowed in the file. It also shows whether the file is ASCII or binary; CCTL or NOCCTL; CIR, MSG, or STD).
2	Displays all of the information found with the 1 option, plus the blocking factor, the number of disc sectors in use (including those in use for file labels and user headers), the number of extents currently allocated, and the maximum number of extents allowed.
-1	Displays the octal listing of the file label. The first line of this listing is the directory entry for the file being listed. The -1 option is available only to users who have System Manager (SM) or Account Manager (AM) capability.

listfile The name of the output file to which the descriptions are written. It is automatically specified as an ASCII file with variable length records and these characteristics: closed in the temporary domain, CCTL (user supplied carriage-control characters), OUT access mode, and EXC (exclusive access) option. The remainder of its characteristics are those obtained with the :FILE command default specifications. This file is temporary and cannot be overwritten by a :BUILD command. The default is $STDLIST.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable (aborts execution).

OPERATION

This command lists descriptions of one or more temporary disc files at the level of detail you select. A file description will not be listed unless the file's home volume set (PV) is mounted. A standard user may specify level 0, 1, and 2 information for any file in the system. A user with Account Manager or System Manager capability may list level -1 data for files in his or her own account. When you send :LISTFTEMP data to the line printer or magnetic tape, the date and time will be appear at the beginning of the listing.

:LISTFTEMP displays the system file $OLDPASS only if wildcard characters are used in the file specification and if $OLDPASS fits the specification.

You may have the file information displayed on a device other than your standard list device. To do so, name the desired device in the :FILE command, then backreference the formal filename on the :LISTFTEMP command line. In the following example, the filename PRTR is equated with the device class LP. Then, a level 2 description of all files in all groups of the logon account is sent to PRTR.

```
:FILE PRTR;DEV=LP
:LISTFTEMP @.@,2;*PRTR
```

If MPE fails to locate the file you request, one of the following error messages will be displayed:

```
TEMPORARY FILE NOT FOUND (CIWARN 3289)

NO TEMP FILES EXIST (CIWARN 3284)
```

If you use a wildcard character in your request, and if MPE cannot locate any files satisfying those conditions, the following warning is displayed:

```
TEMPORARY FILE NOT FOUND (CIWARN 3289)
```

:LISTFTEMP requires 8K of stack space to execute. If you use the command programmatically, make the appropriate *maxdata* allowances for your application.

EXAMPLES

The following examples illustrate several uses of the :LISTFTEMP command.

Level 0 (Default) Output Format

```
:LISTFTEMP @.@.@

TEMPORARY FILES FOR JASON.MANU,PEN

$OLDPASS.PEN.MANU
AJ.PEN.MANU
AJJ.PEN.MANU
Z10Z414A.JON.MANU
TEMPFILE.TEMPGROUP.SYS.
```

Level 1 Output Format

```
:LISTFTEMP B@,1

TEMPORARY FILES FOR JASON.MANU,PEN
ACCOUNT=  MANU        GROUP=  PEN

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

BONOP1             80B  FA          10         10
BOTE      TDPQ    104B  FA         636       2269  (TEMP)
BOXES     DRAW    128W  FB          84         84  (TEMP)
BRAMPLES*          80B  FA          20         20  (TEMP)
BRRORLOG           80B  FAO         12       1024  (TEMP)
```

Level 2 Output Format

```
:LISTFTEMP A@,2

TEMPORARY FILES FOR JASON.MANU,PEN
ACCOUNT=  MANU         GROUP=  PEN

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

APPB               88B  FA         414        414  16      162 14 14 (TEMP)
APPM               80B  VAM        119        240  16       12  8  8 (TEMP)
APPC               80B  FAO        127        208  16       70  8  8 (TEMP)
APPK      KSAM     80B  FA          16        200  16       20  2  7 (TEMP)
```

The column headings have the following meanings:

HEADING	MEANING
FILENAME	Displays the filename. An asterisk (*) following the filename in the level 1 and 2 option indicates the file is open for READ or WRITE access.
CODE	Indicates a special file format. Refer to the :BUILD or :FILE command for a list of file codes and their meaning.
SIZE	Displays the size of the records under the heading SIZE, indicating the number of words (W) or bytes (B).
TYP	The first column under this heading contains information concerning record format. This will be listed in one of the following ways: F (fixed length), V (variable length), or U (undefined length). The second column indicates whether the file is A (ASCII) or B (binary). The third column indicates the kind of file. This information will appear in one of the following ways: Blank (Standard file unless KSAM appears in the code field.), O, (circular file), D (message file), or R (relative I/O file).

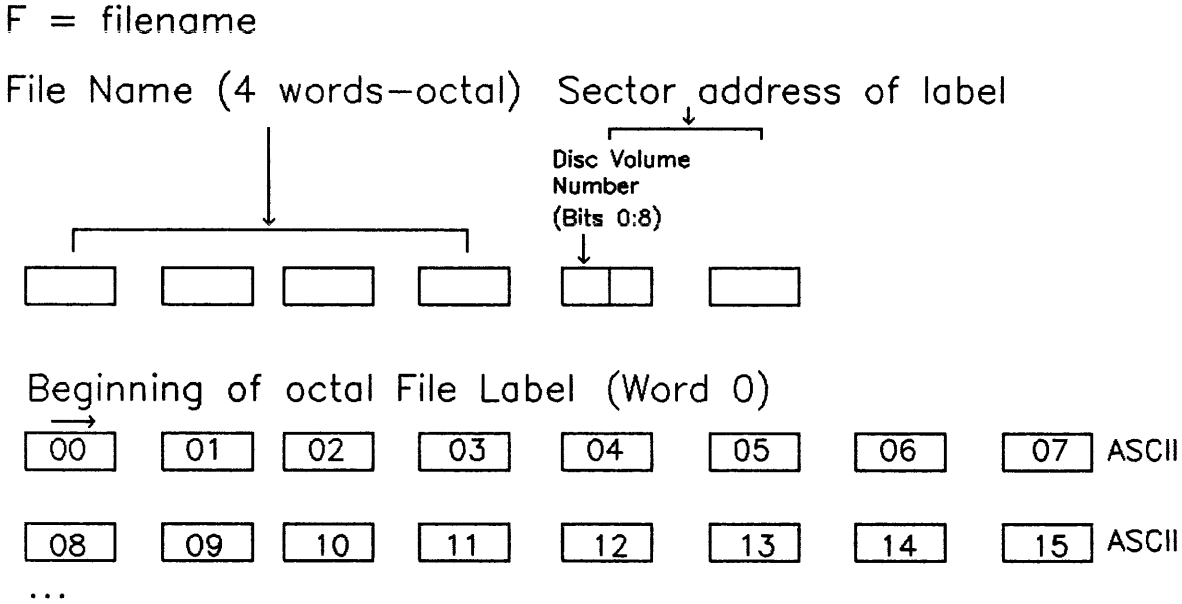
	In the fourth column a "C" appears only if the ;CCTL option is taken.
EOF	End of file location.
LIMIT	Maximum number of records allowed.
R/B	Blocking factor.
SECTORS	Sectors in use.
#X	Extents allocated.
MX	Maximum number of extents allowed.
(TEMP)	Temporary files.

Level -1 Output Format

```
:LISTFTEMP UDCS, -1
F  =  UDCS                                (TEMPORARY FILES)
052504 041523 020040 020040 001001 120353                UDCS........
052504 041523 020040 020040 041517 046504 020040 020040 UDCS....COMD....
045440 020040 020040 020040 046501 051111 051523 040440 K.........TECH.
020040 020040 020040 020040 020202 004040 000001 123525 ..............U
123525 123525 000000 000000 000014 000000 000000 002163 .U.U..........s
000000 000000 120173 021427 000005 177650 002374 005015 ................
000024 000036 000000 002163 001001 120353 000400 104644 .......s........
001401 032743 001001 162503 001401 036363 001401 041324 ..5....C......B.
001001 164214 001401 042166 001002 001740 001401 044134 ......Dv......H.
001002 002656 001401 045323 001002 006740 000400 150014 ......J.........
000000 000000 000000 000000 000000 000000 000000 000000 ................
000000 000000 000000 000000 000000 000000 000000 000000 ................
000000 000000 000000 000000 000000 000000 000000 000000 ................
000000 000000 000000 000000 000000 000000 000000 000000 ................
000000 000000 000000 000000 004072 034411 123525 000000 .........9..U..
000000 000000 000000 000000 000000 000000 004073 002010 ................
000000 000000 000000 000000 042111 051503 020001 000001 ........DISC....
```

A pictorial explanation of the level -1 output, follows:

:LISTFTEMP *filename*, -1



The disc file label contains the following:

WORDS		CONTENTS
0-3		Local filename.
4-7		Group name.
8-11		Account name.
12-15		Identity of file creator.
16-19		File lockword.
20-21		File security matrix.
22	(Bits 0:7)	Reserved for NLS.
	(Bits 8:13)	Not used.
	(Bits 14:1)	Store/Restore release bit.
	(Bits 15:1)	File secure bit: If 1, file secured. If 0, file released.
		File creation date.*
23		Last access date.*
24		Last modification date.*
25		File code.
26		Private Volume Information.
27	(Bits 0:1)	Class flag bit.
	(Bits 1:3)	Not used.
	(Bits 4:4)	Mounted volume table index.
	(Bits 8:8)	Volume mask.
28	(Bit 0:1)	Store BIT. (If on, :STORE or :RESTORE, in progress.)
	(Bit 1:1)	Restore Bit. (If on, :RESTORE in progress.)
	(Bit 2:1)	Load Bit. (If on, program file is loaded.)
	(Bit 3:1)	Exclusive Bit. (If on, file is opened with exclusive access.)
	(Bits 4:4)	Device subtype

	(Bits 8:6)	Device type.
	(Bits 14:1)	File is open for write.
	(Bits 15:1)	File is open for read
29	(Bits 0:8)	Number of user labels written.
	(Bits 8:8)	Number of user labels.
30-31		Maximum number of logical records.
32-33		File control block vector.
34		Checksum.
35		Coldload identity.
36		Foptions specifications.
37		Logical record size (in negative bytes).
38		Block size (in words).
39	(Bits 0:8)	Sector offset to data.
	(Bits 8:3)	Not used.
	(Bits 11:5)	Number of extents minus 1.
40		Logical size of last extent size in sectors.
41		Extent size.
42-43		Number of logical records in file.
44-107		Two-word addresses of up to 32 disc extents, beginning with address of first extents (words 44-45).
108-109		File allocation time.
110		File allocation date.
111		Not used.
112-113		Start of file block number.
114-115		Block number of last block.
116-117		Number of open and closed records.
118-119		The time of the last file modification.
124-127		Device class.

* Dates are in the same format as the value returned from the CALENDAR intrinsic:

Bits 0 6 7 15

Year of Century Day of Year

GENERAL EXAMPLES

To list all of the files in your logon group that have X as the second alphabetic character in their names, enter:

```
:LISTFTEMP ?X@

TEMPORARY FILES FOR JASON.MANU,PEN

EXAMP
:
```

To list all of the three-character files in your logon group and account that end in the letter N, enter:

```
:LISTFTEMP ??N

TEMPORARY FILES FOR JASON.MANU,PEN

SYN
:
```

:LISTGROUP

Prints a list of attributes for groups.

SYNTAX

```
            {group.acct}
            {group     }
:LISTGROUP  {@.acct    } [listfile]
            {@         }
            {@.@       }
```

PARAMETERS

@	Specifies that all groups in the logon account are to be listed. Default.
@.@	Specifies that all groups in all accounts are to be listed. Only users with SM capability may specify this option. Default is @.
group.acct	Specifies the group and account to be listed. Account Managers cannot specify an account other than their own.
group	Specifies the group (in the logon account) to be listed. Default is @.
@.acct	Specifies that all groups in the designated account are to be listed. If an Account Manager specifies an account, it must be his or her own account. Default is @.

Note that the characters @, #, and ? can be used as "wildcard" characters, but will count toward the eight character limit. These wild card characters have the following meanings:

@	Specifies one or more alphanumeric characters or denotes all members of the set.
#	Specifies one numeric character.
?	Specifies one alphanumeric character.

The characters can be used as follows:

n@	List all groups starting with the character "n".
@n	List all groups ending with the character "n".
n@x	List all groups starting with the character "n" and ending with the character "x".
n#...#	List all groups starting with the character "n" followed by up to seven digits.
?n@	List all groups whose second character is "n".

	n?	List all two-character groups starting with the character "n".
	?n	List all two-character groups ending with the character "n".
listfile		Destination of the attribute listing. The default is $STDLIST, but the output may be redirected with the :FILE command.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is Breakable (aborts execution). A user must have Account Manager (AM) or System Manager (SM) capability to execute this command.

OPERATION

This command lists the attributes for all groups in one account (AM capability) or all groups in all accounts (SM capability). Also available is information on groups spanned to Private Volumes (PV), whether or not the volumes are mounted when the information is requested.

A sample octal dump appears below. The listing includes both group and system information, headed by "G = " and the group's name. A translation of those bytes that contain alphanumeric ASCII characters appears to the right of each line of the octal dump.

```
G  =  PUB
050125 041040 020040 020040 000514 020040 020040 020040 PUB......L......
020040 000000 001026 077777 177777 000000 046536 077777 ............M...
177777 000002 006457 077777 177777 020410 041010 000601 ............B...
000000 000516 020040 020040 020040 020040 020040 020040 ...N............
020040 020040 020040 020040 020040 020040 000000 000000 ................
000000
```

Below is the octal listing translated into its decimal representation:

```
--00-- --01-- --02-- --03-- --04-- --05-- --06-- --07--
--08-- --09-- --10-- --11-- --12-- --13-- --14-- --15--
--16-- --17-- --18-- --19-- --20-- --21-- --22-- --23--
--24-- --25-- --26-- --27-- --28-- --29-- --30-- --31--
--32-- --33-- --34-- --35-- --36-- --37-- --38-- --39--
--40--
```

The information in the octal dump is decoded in the following table:

WORDS	CONTENT
0-3	Group name, in ASCII.
4	File index pointer.
5-8	Password.
9-10	Permanent file space usage count (in sectors).
11-12	Permanent file space limit (in sectors).
13-14	Central-processor time usage count (in seconds).
15-16	Central-processor time limit (in seconds).
17-18	Connect-time count (in minutes).
19-20	Connect-time limit (in minutes).
21-22	Purge and group-security flags.
23	Capability-class attributes.
24	Group directory base linkage.
25	Group volume set definition index.
26-29	Definition's account name.
30-33	Definition's group name.
34-37	Definition's volume set name.
38	GSAVEFIPNTR
39	GMOUNTRECNTR
40	GSPARE

Where:

- GSAVEFIPNTR contains the original Group File Index Pointer or an address relative to the base of the SYSVS (System Volume Set) directory.
- GMOUNTRECNTR is a counter which keeps track of the number of times a group's Home Volume Set has been bound.
- GSPARE is an MPE reserved word.
- The group name and password are eight-character names, right-padded with blanks.
- The double word numeric quantities are double word integers, with %17777777777 representing "unlimited".
- Group attributes duplicate the second word of the double word capability returned by the -WHO intrinsic. Refer to the MPE V Intrinsics Reference Manual (32033-90007).

EXAMPLE

To list the attributes of the PUB group in his or her own account, the Account Manager enters:

```
:LISTGROUP PUB

G  =  PUB
050125 041040 020040 020040 000514 020040 020040 020040 PUB.....L.....
020040 000000 001026 077777 177777 000000 046536 077777 .............M...
177777 000002 006457 077777 177777 020410 041010 000601 .............B...
000000 000516 020040 020040 020040 020040 020040 020040 ...N............
020040 020040 020040 020040 020040 020040 000000 000000 ................
000000
```

:LISTLOG

Lists currently active logging identifiers on the system and whether log file changing has been enabled.

SYNTAX

```
:LISTLOG [logid[;PASS]]
```

PARAMETERS

logid	The specific logging identifier to be verified. Default is to list all currently active logging identifiers on the system.
PASS	Causes the password associated with the logging identifier to be displayed. The password can be used only by the creator of the logging identifier.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have User Logging (LG) capability to execute this command.

OPERATION

This command lists the logging identifier specified with its associated creator and log file. The column labeled :CHANGE indicates whether the :CHANGELOG command is permitted: that is, whether the name of the first logging file ends in *"001"* and thus follows the naming convention required by the :CHANGELOG command. The column labeled AUTO indicates whether an automatic :CHANGELOG is permitted: that is whether the AUTO parameter has been specified with a :GETLOG or :ALTLOG command.

If the *logid* parameter is not entered, all logging identifiers on the system are displayed with their creators and log files. The PASS parameter, which can be used only by the creator of the logging identifier specified, causes the password associated with the logging identifier to be listed. You must have User Logging (LG) capability in order to use this command.

EXAMPLE

To list all logging identifiers on the system, enter:

```
:LISTLOG

LOGID     CREATOR            CHANGE   AUTO   CURRENT LOGFILE

TESTLOG   LALITHA.MPEM       YES      YES    LAL001.PEJAVAR.MPEM
TEST1     MARK.MPEM          YES      NO     M001.KSAM3000.MPEM
TEST2     PAT.MPEM           NO       NO     TEST.ALVAREZ.MPEM
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)
MPE V Intrinsics Manual (32033-90007).
Refer also to the :ALTLOG, :CHANGELOG, and :GETLOG commands in this section.

:LISTUSER

Lists attributes currently assigned to users.

SYNTAX

```
          {user.acct}
          {user     }
:LISTUSER {@.acct   } [,listfile]
          {@        }
          {@.@      }
```

PARAMETERS

user.acct — The specified user in the specified account. If Account Managers specify *acct*, it must be their own account.

user — The specified user in the logon account.

@.acct — All users in the specified account. Account Managers may specify their own account only.

@ — All users in the logon account. Default.

@.@ — All users in all accounts. (The attributes of the account are also listed.) This option can be specified only by users with System Manager (SM) capability.

Note that the characters @, #, and ? can be used as "wildcard" characters, but will count toward the the eight character limit. These wild card characters have the following meanings:

@	Specifies zero or more alphanumeric characters and denotes all members of the set.
#	Specifies one numeric character.
?	Specifies one alphanumeric character.

The characters can be used as follows:

n@	List all users starting with the character "n".
@n	List all users ending with the character "n".
n@x	List all users starting with the character "n" and ending with the character "x ".
n##...#	List all users starting with the character "n" followed by up to seven digits.
?n@	List all users whose second character is "n".

	n?	List all two-character users starting with the character "n".
	?n	List all two-character users ending with the character "n".
listfile		Destination of the attribute listing. The default is $STDLIST, but the output may be redirected with the :FILE command.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is Breakable (aborts execution). A user must have AM (Account Manager) or SM (System Manager) capability to execute this command.

OPERATION

This command generates an octal dump of the attributes currently assigned to users. Each entry is headed by a "U = " and the user's name; it contains both user and system information. A translation of those bytes containing ASCII data appears to the right of each line, as shown in the sample below. Account Managers may list only users in their own account, while System Managers may specify all users in all accounts on the system.

```
U  =  COMGRAMS
041517 046507 051101 046523 060003 000613 000000 000000 COMGRAMS........
020040 020040 020040 020040 050125 041040 020040 020040 ........PUB.....
000000 040226 000121                                     .....Q
```

The octal information is translated into a decimal representation below.

```
--00-- --01-- --02-- --03-- --04-- --05-- --06-- --07--
--08-- --09-- --10-- --11-- --12-- --13-- --14-- --15--
--16-- --17-- --18--
```

Each word or group of words is decoded in the following table:

WORDS	CONTENT
0-3	User name.
4-5	Capability
6-7	Local attributes.
8-11	Password.
12-15	Home group.
16	Number of users logged-on.
17	Maximum job priority
18	User entry in COMMAND.PUB.SYS

- The username, password, and home group are eight-character names, right padded with blanks.
- Capability can be decoded by reference to the WHO intrinsic. Refer to the MPE V Intrinsics Reference Manual (32033-90007).
- The maximum job priority is a numerical quantity; 150 (%226) for example, represents the CS subqueue.

EXAMPLE

To list the attributes of the user MGR in the account SOPRM, enter:

```
:LISTUSER MGR.SOPRM
U  =  MGR
046507 051040 020040 020040 077607 000713 000000 000000 MGR.............
020040 020040 020040 020040 050125 041040 020040 020040 ........PUB.....
000002 040226 000466                                    .....6
:
```

Produces a formatted listing of volume set definition information.

SYNTAX

```
                    {0}
:LISTVS [vslist][{1}][,listfile]
                    {2}
```

PARAMETERS

vslist

Specifies the volume set definitions to be listed. This positional parameter has the form:

vsname[.*groupname*[.*acctname*]]

The characters @, #, and ? can be used as wildcard characters in any position of the *vslist* parameter. The wildcard characters count toward the eight character limit for user, group, account, and filenames. These wildcard characters have the following meanings:

@	Specifies one or more alphanumeric characters. When used by itself, @ denotes "all members of the set".
#	Specifies one numeric character.
?	Specifies one alphanumeric character.

These characters can be used as follows:

n@	Lists all volume set definitions starting with the character "n".
@n	Lists all volume set definitions ending with the character "n".
n@x	Lists all volume set definitions starting with the character "n" and ending with the character "x".
n#######	Lists all volume set definitions starting with the character "n" followed by seven digits.
?n@	Lists all volume set definitions whose second character is "n".
n?	Lists all two-character volume set definitions starting with the character "n".
?n	Lists all two-character volume set definitions ending with the character "n".

Note that depending upon the placement of these characters in the *vslist* parameter, they can also be used to replace group and account names. If you do not specify names for the group or account, MPE assumes the logon group and account. Default is @ (lists all home volume sets in logon group).

0, 1, or 2

Displays information about volume sets. The 0 option displays volume set names and volume class names The 1 displays the information shown for 0 as well as volume names of volume set members and volume class members, and types of devices on which the volumes reside. The 2 option displays the information shown for 0 and 1. With the 2 option, if the device is mounted, then the *ldev* and the status "MOUNT" also appear. An asterisk (*) following a volume set indicates that the volume set is mounted . Default is 0.

listfile

Actual file designator of output file to receive listings. Default is $STDLIST. If you do not use the default, this file must be named in a :FILE command and backreferenced.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is Breakable (aborts execution). A user must have Use Volumes (UV) or Create Volumes (CV) capability to execute this command.

OPERATION

The :LISTVS command allows you to determine the configuration of a volume set and produce a formatted listing of volume set definition information.

The numeric parameter of the :LISTVS command determines the amount of information displayed. If omitted or if a 0 is specified, item 1 in the following list is displayed. If detail is 1, items 1 through 3 are displayed. If detail 2 is specified, items 1 through 4 are displayed:

1. Volume set names and volume class names.
2. Volume set member volume names and types.
3. Volume class member volume names and types.
4. Logical device numbers (*ldev*) and status of the devices on which the volume set is mounted. This information is omitted for any volume set not currently mounted.

EXAMPLES

To display the all members, numbers, class names, and types of the volume set, along with the logical device number and status of the volume set, enter:

```
:LISTVS,2
 ACCOUNT= TECHPUBS          GROUP= MAC

 VOLSET          MEMBERS       TYPE      LDEV     STATUS
 --------        --------    --------    ----    --------

   MAC*            MAC       HP7920        4     MOUNTED
:
```

To display the members and type of the volume set for all users and accounts, enter:

```
:LISTVS @.@,1
 ACCOUNT= TECHPUBS          GROUP= MAC

 VOLSET          MEMBERS       TYPE
 --------        --------    --------

 MAC*          MAC         HP7920
:
```

To display the volume set numbers and volume class names, enter:

```
:LISTVS @.@,0
 ACCOUNT= TECHPUBS          GROUP= MAC

 VOLSET
 --------

 MAC*
:
```

To display volume set definition information for volume sets in your logon group and account, with the listing sent to a line printer, enter:

```
:FILE LIST;DEV=LP
:LISTVS @,1;*LIST
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)

:LMOUNT

Performs a logical mount of a private volume set.

SYNTAX

```
:LMOUNT vcsname.group.account [;GEN=[genindex]]
```

PARAMETERS

vcsname	Volume class/set name which specifies the mounting of a previously defined volume class or volume set.
group.account	Specifies the group and account under which the volume set was created.
genindex	A value from -1 to 32767 specifying which generation of the volume set is to be mounted (-1 allows any generation to be mounted). If omitted, the system does not check the generation version of the specified volume set.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It is executable only at the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

When the :LMOUNT command executes, all disc drives containing members of the specified volume set become ASSIGNED. Each volume set is logically attached to the drive until a :LDISMOUNT command is executed, at which time the disc drive becomes AVAILABLE.

System users issue mount requests implicitly through their programs, or explicitly with a :MOUNT command. User initiated mount requests cause the specified volume set to be linked into the system's accounting structure at the group level. This is called "binding". When the Console :LMOUNT command is executed, no such binding occurs.

If the Private Volumes Facility was enabled with :VMOUNT ON,AUTO, MPE will automatically attempt to satisfy the mount request; the :LMOUNT will succeed if the specified volume class/set is physically connected to the system.

If the Private Volumes Facility was enabled with :VMOUNT ON (omitting the AUTO parameter), you must reply to your own mount request, even though the volume class/set may already be mounted and in use. If the specified volume class/set is not physically present, the system looks for available drives of the types required to satisfy the request. If sufficient drives are available, the system reserves them and prints a message telling you to mount the volume class/set on one or more logical devices, and to reply "YES " or "NO ". If sufficient drives are not available, the system reserves those which will satisfy part of your request, and prints a message listing the type and number of drives it requires.

If you can free the required drive(s), reply "YES" after the drive is AVAILABLE. If you reply "NO", the original :LMOUNT request will be canceled.

EXAMPLE

To mount a private volume set named DATABASE, in the PAYROLL group of the ACCTNG account, enter:

```
:LMOUNT DATABASE.PAYROLL.ACCTNG
```

ADDITIONAL DISCUSSION

Refer to the discussion of "PRIVATE VOLUMES" in Section of the MPE V System Operation and Resource Management Reference Manual (32033-90005).

Starts, restarts, or stops User Logging.

SYNTAX

```
                {,RESTART}
:LOG logid      {,START  }
                {,STOP   }
```

PARAMETERS

logid	Logging identifier previously established with a user :GETLOG command.
START	Initiates a logging process.
RESTART	Restarts a logging process.
STOP	Terminates a logging process.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have System Supervisor (OP) capability to execute this command.

OPERATION

This command allows you to start, restart, or stop User Logging. For further discussion of User Logging, refer to the MPE V System Operation and Resource Management Manual (32033-90005) and to the MPE V Intrinsics Reference Manual (32033-90007).

The release of the G.02.00 version of MPE, make it unnecessary to use the :LOG command (with STOP, START, or RESTART) to change log files. You may change log files "on the fly", without the delay normally caused by executing a :LOG command. You may instead invoke the :CHANGELOG command to enable interactive log file changing; you may use the AUTO parameter of the :ALTLOG and :GETLOG commands to enable automatic log file changing. Refer to the commands :ALTLOG, :CHANGELOG, :GETLOG, and :SHOWLOGSTATUS in this section of the manual.

EXAMPLE

To start the logging process identified by *logid* LOGPROCX enter:

```
:LOG LOGPROCX,START
```

=LOGOFF

Aborts all executing jobs/sessions and prevents any further logon.

SYNTAX

```
=LOGOFF
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be issued only from the Console.

OPERATION

This command sets the job and session execution limits to 0 and aborts all jobs and sessions, including the session logged onto the System Console. To enter subsequent console commands, you must logon to the console using =LOGON.

Execution of this command leaves the system in a job/session inactive state. However, since devices are still job/session accepting, job introduction is not disabled: System Managers and System Supervisors can still introduce and run HIPRI jobs. Spoolers will remain operating, but unless a user is able to initiate a HIPRI session at the System Console, it is impossible to alter their operation.

Any pending requests that require a =REPLY from the System Console must be satisfied before issuing =LOGOFF, or MPE will abort the requests.

EXAMPLE

To abort all executing jobs/sessions, enter:

```
A^C
=LOGOFF
16.53/25/ALL JOBS LOGGED-OFF
```

=LOGON

Enables job/session processing following =LOGOFF .

SYNTAX

```
=LOGON
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be issued only from the Console.

OPERATION

This command enables the processing of jobs/sessions following the execution of the =LOGOFF command. =LOGON re-establishes the job/session limits that were in effect before the execution of a =LOGOFF command, and allows non-HIPRI jobs/sessions to logon again.

EXAMPLE

To enable job/session processing, enter:

```
=LOGON
```

Requests the System Operator to mount a volume set.

SYNTAX

```
:MOUNT{   *    }[ groupname[ acctname]]
      { vcsname }
  [ GEN=[genindex]]
```

PARAMETERS

* or *vcsname*	Specifies the volume set or volume class name. The asterisk (*) specifies the home volume set for the group and account specified, or the logon group and account if *groupname.acctname* is not specified. The *vcsname* parameter is the volume class/set name, specifying a previously defined (with a :NEWVSET command) volume class name or volume set name. Default is (*).
groupname.acctname	Specifies the group and account under which the volume set was created. Default is logon group and account.
genindex	A value from -1 to 32767 specifying which generation of the home volume set is to be mounted. A value of -1 indicates that any generation is permitted. If omitted, the system will ignore the generation when attempting to satisfy the :MOUNT request.

USE

This command may be issued from a Session, Job or in BREAK. It may not be used from a Program. It is not Breakable.

OPERATION

The :MOUNT command connects you to a specific volume set. In addition, this command allows you to specify that the volume set is to be treated as a home volume set; therefore, it is to be bound to the system's accounting structure with respect to a particular group. The :MOUNT command is rejected if there is an operator initiated :LDISMOUNT command pending for the specified volume set. The requesting job is suspended until the mount is completed or rejected.

In addition, various user commands which give you access to your logon group's home volume set will implicitly initiate mount requests if the volume set is not mounted already. An example of one of these commands (:BUILD) is:

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

To issue a mount request programmatically you may issue an FOPEN call referencing a file residing on an unmounted volume set; this causes an implicit user initiated mount request. An FOPEN mount remains in effect until a corresponding FCLOSE intrinsic call is issued. The programmatic request is used when a single job/session step requires a certain volume set. Refer to the MPE V Intrinsics Reference Manual (32033-90007) for a description of programmatic mounting.

EXAMPLE

To request the System Operator to mount volume set MYSET, generation index 43, enter:

```
:MOUNT MYSET;GEN=43
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)

:NEWACCT

Creates a new account and an associated Account Manager and PUB Group.

SYNTAX

```
:NEWACCT acctname,mgrname
    [;PASS=[password]]
    [;FILES=[filespace]]
    [;CPU=[cpu]]
    [;CONNECT=[connect]]
    [;CAP=[capabilitylist]]
    [;ACCESS=[fileaccess]]
    [;MAXPRI=[subqueuename]]
    [;LOCATTR=[localattribute]]
    [;VS=[volset:SPAN]]
```

PARAMETERS

acctname — Name to be assigned to the new account. This name must contain from one to eight alphanumeric characters, beginning with an alphabetic character.

mgrname — Name of the Account Manager. This is always the first user created under the account. The manager receives the following attributes:

- User Password — None.
- Capability List — Same as the account capability list.
- Scheduling Priority — Same as the account maximum priority.
- Local Attribute — Same as account local attributes.
- Home Group — PUB

The attributes of an Account Manager may be changed with the :ALTUSER command after *mgrname* is defined. However, in no case, is this user granted attributes greater than those assigned the account.

password — Account password, used for verifying logon access only. This password must contain from one to eight alphanumeric characters beginning with an alphabetic character. Default is that no password is assigned.

filespace Disc storage limit, in sectors, for the permanent files of the account. The maximum value you may define is 2,147,483,647 sectors. Default is unlimited file space.

cpu Limit on total CPU time, in seconds, for this account. This limit is checked only when a job or session is initiated, thus, the limit never causes the job or session to abort. The maximum value you may define with :NEWACCT is 2,147,483,647 seconds. Default is that no limit is assigned.

connect Limit on total session connect time, in minutes, allowed the account. This limit is checked at logon, and when the job or session initiates a new process. The maximum value you may define is 2,147,483,647 minutes. Default is that no limit is assigned.

capabilitylist List of capabilities, separated by commas, permitted this account. Each capability is denoted by a two-letter mnemonic, as follows.

System Manager	=	SM
Account Manager	=	AM
Account Librarian	=	AL
Group Librarian	=	GL
Diagnostician	=	DI
System Supervisor	=	OP
Network Administrator	=	NA
Node Manager	=	NM
Permanent Files	=	SF
Access to non-sharable I/O devices	=	ND
Use Volumes	=	UV
Create Volumes	=	CV
Use Communications Subsystem	=	CS
Programmatic Sessions	=	PS
User Logging	=	LG
Process Handling	=	PH
Extra Data Segments	=	DS
Multiple RINS	=	MR
Privileged Mode	=	PM
Interactive Access	=	IA
Local Batch Access	=	BA

Default is AM,AL,GL,SF,ND,IA,BA.

fileaccess

File access restrictions assigned the account. Default is R,L,A,W,X:AC.

```
   {R}
   {L}
([{A}[,...]:{ANY} ][;...])
   {W}     {AC }
   {X}
```

where R, L, A, W, and/or X specify modes of access by types of users (ANY or AC) as follows:

R	=	Read
L	=	Lock (allows exclusive access to file)
A	=	Append (implicitly specifies L also)
W	=	Write (implicitly specifies A and L also)
X	=	Execute

The user types are specified as follows.

ANY	=	Any user
AC	=	Member of this account only

The default is no security restrictions at the account level. Two or more modes may be specified if they are separated by commas. Both user types may be specified if they are separated by commas.

subqueuename

The name of the subqueue of highest priority that can be requested by any process of any job/session in the account. This parameter is specified as AS, BS, CS, DS, or ES.

CAUTION

Processes capable of executing in the AS or BS subqueues can deadlock the system. Assigning non-priority system and user processes to these subqueues can prevent critical processes from executing. Exercise extreme caution when assigning processes to these subqueues.

localattribute

The local attribute of the account, as defined at the installation site. This is a double-word bit map used to further classify accounts. When it is not part of standard MPE security provisions, programmers may define local attributes (which will be checked by the -WHO intrinsic) to enhance their software's security. Default is double-word 0.

volset

The volume set or class reference which, when fully qualified, is in the form *vcsid.groupname.acctname*, where *vcsid* refers to a previously defined volume set or class.

SPAN

Specifies that the *accountname* is to be inserted in the accounting directory of the specified volume set (*volset*). The specified volume set must have been physically mounted previously for the SPAN operation to succeed.

The associated Account Manager and PUB group are not created in the accounting directory of the specified volume set.

Once the account has been spanned to the volume set, it does not need to be spanned again (i.e. the *volset* parameter is not necessary) on another system which shares the volume set under the same account name.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have System Supervisor (SM) capability to execute this command.

OPERATION

The :NEWACCT command may only be executed by the System Manager. The System Manager is responsible for establishing the accounting structure best suited to the computer installation.

When a keyword is specified, but its corresponding parameter is omitted (as in ;ACCESS=), the default value for that keyword is assigned (in this case, R, L, A, W, X:AC). The default is also assigned when an entire keyword parameter group (such as ;ACCESS=*fileaccess*) is omitted.

After the System Manager creates accounts and their PUB groups, and has designated the Account Managers for those accounts, the new Account Managers may logon and redefine their own attributes and those of their PUB groups. Account Managers also define new users and groups. However, the capabilities and attributes the Account Managers assign to groups and users cannot exceed those assigned to the account itself by the System Manager. For example, if the System Manager does not assign the account DS capability, no users in the account are permitted DS capability (which prohibits them from using extra data segments).

The PUB Group is initially assigned no password and the same capability class attributes, permanent file space limit, CPU limit, and connect time limit as the account. Its initial security allows READ and EXECUTE access to all users who successfully logon to the account, and APPEND, WRITE, LOCK, and SAVE access to Account Librarian (AL) and Group Users (GU) only. These access provisions are (R,X:ANY;A,W,L,S:AL,GU).

EXAMPLE

To create an account with the account name ACI , and the Account Manager name MNGR , with all other parameters assigned by default, enter:

```
:NEWACCT ACI,MNGR
```

:NEWGROUP

Creates a new group within an account.

SYNTAX

```
:NEWGROUP groupname
    [;PASS=[password]]
    [;FILES=[filespace]]
    [;CPU=[cpu]]
    [;CONNECT=[connect]]
    [;CAP=[capabilitylist]]
    [;ACCESS=[fileaccess]]
    [;VS=[volset[:SPAN]]
```

PARAMETERS

groupname — The name of the new group, which must consist of one to eight alphanumeric characters, beginning with an alphabetic character.

password — Group password, used for verifying logon access only. Default is that no password is assigned.

capabilitylist — A list of capability-class attributes, consisting of any or all of the following: IA, BA, PM, MR, DS, or PH, where:

Process Handling	=	PH
Extra Data Segments	=	DS
Multiple RINS	=	MR
Privileged Mode	=	PM
Interactive Access	=	IA
Local Batch Access	=	BA

This list imposes a limit on program files belonging to the group. A capability cannot be assigned to the group if it has not been defined for the account in which the group resides. Default is IA, BA.

filespace — Disc storage limit, in sectors, for the permanent files of the group. You cannot specify a *filespace* for a group that is greater than the limits currently defined for the group's account. If an account's *filespace* is changed later, group limits are automatically changed. Default is a storage limit equivalent to the account's *filespace*.

cpu

Limit on total CPU time, in seconds, for the group. This limit is checked when a job or session is initiated, and, if exceeded during an active job or session never causes the job or session to abort. If the limit is exceeded, new jobs or sessions will be prevented from initiating until the counter is reset with the :RESETACCT command. The maximum value you may define with this command is 2,147,483,647 seconds. However, you cannot specify a group's CPU limit that exceeds the limit defined for the account. If an account's CPU limit is changed later, the groups' limits will be changed automatically. Default is the account CPU limit.

connect

Limit on total session connect time, in minutes, allowed the group. This limit is checked at logon, and when the job or session initiates a new process. The maximum value you may define is 2,147,483,647 minutes, and it cannot exceed the limit defined for the account. If the account connect is later changed, the group connect time is automatically changed. Default is the account connect limit.

fileaccess

The restriction on file access pertinent to this group. Default is R,X:ANY;A,W,L,S:AL,GU for the Public Group (PUB); and R,A,W,L,X,S:GU for all other groups.

```
{R}                {ANY}
{L}                {AC }
{A} [,...]:        {GU } [,...]
{W}                {AL }
{X}                {GL }
                   {CR }
```

GL, CR) as follows:

R = Read
L = Lock (exclusive file access)
A = Append (implies L)
W = Write (implies A and L)
X = Execute
S = Save

The user types are specified as follows:

ANY = Any user
AC = Member of this account only
GU = Member of this group only
AL = Account librarian user only
GL = Group librarian user only
CR = Creating user only

Two or more user or access types may be specified if they are separated by commas.

volset

The volume set or class reference which, when fully qualified, is in the form *vcsid.groupname.accountname*, where *vcsid* refers to a previously defined volume set or class.

SPAN

The SPAN option specifies that *groupname* is to be inserted in the accounting directory of the specified volume set (*volset*). The specified volume set must have been previously mounted physically for the SPAN operation to succeed.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have Account Manager (AM) capability to execute this command.

OPERATION

Account Managers use the :NEWGROUP command to create groups within their accounts and assign attributes to each. The attributes assigned to the group may not exceed those permitted the accounts themselves (defined when the System Manager created the accounts). However, within account limits, the Account Manager may redefine the group and user attributes and capabilities, as well as those of the PUB group.

The PUB group is initially assigned no password and the same capability-class attributes, permanent file space limit, and CPU time limit as the account. Its initial security grants READ (R) and EXECUTE (X) access to all users (ANY) who successfully logon to the account. APPEND (A), WRITE (W), LOCK (L), and SAVE (S) access is assigned to the Account Librarian (AL) and Group Users (GU) only.

When a keyword parameter (such as ;PASS=) or keyword parameter group (such as ;PASS=*password*) is omitted from the :NEWGROUP command, the default value corresponding to that parameter is assigned.

EXAMPLE

To create a new group named GROUP1 which will be assigned all default capabilities, enter:

```
:NEWGROUP GROUP1
```

:NEWUSER

Defines a new user.

SYNTAX

```
:NEWUSER username
    [;PASS=[password]]
    [;CAP=[capabilitylist]]
    [;MAXPRI=[subqueuename]]
    [;LOCATTR=[localattribute]]
    [;HOME=[homegroupname]]
```

PARAMETERS

username — The name of the user. The name must consist of one to eight alphanumeric characters, beginning with an alphabetic character.

password — User password, used for verifying logon access only. The password must consist of one to eight alphanumeric characters, beginning with an alphabetic character. Default is that no password is assigned.

capabilitylist — List of capabilities, separated by commas, permits this account. Each capability is denoted by a two-letter mnemonic, as follows:

System Manager	=	SM
Account Manager	=	AM
Account Librarian	=	AL
Group Librarian	=	GL
Diagnostician	=	DI
System Supervisor	=	OP
Network Administrator	=	PS
Node Manager	=	NA
Permanent Files	=	NM
Access to nonsharable I/O devices	=	SF
Use Volumes	=	ND
Create Volumes	=	UV
Use Communications Subsystems	=	CV
Programmatic Sessions	=	CS
User Logging	=	LG
Process Handling	=	PH
Extra Data Segments	=	DS
Multiple RINs	=	MR
Privileged Mode	=	PM
Interactive Access	=	IA
Local Batch Access	=	BA

Capabilities assigned to the user with the ;CAP= parameter cannot exceed those assigned the account. If the account's capabilities are altered, any capabilities removed from the account are also removed from the user. The user's capabilities are always verified to be a subset of the account's at logon. This prevents a user from being granted a capability not assigned the account. Note that CV capability, which allows users to define Private Volumes, also gives the user UV capability, so that they may use private volumes. Default is IA, BA, ND, SF.

subqueuename The name of the highest priority subqueue that any job or session in the account can request for executing processes. The *subqueuename* may be either AS, BS, CS, DS, or ES. The priority specified for the user in :NEWUSER cannot be greater than that specified for the account.

The *subqueuename* defined for the user is checked against the *subqueuename* defined for the user's account at logon. The lower priority of the two is used as the maximum priority and restricts all processes of the job/session. Also, the priority requested by the user at logon is checked against the *subqueuename* defined for that user, and the lower of these two values is granted. Default is CS.

CAUTION

Processes capable of executing in the AS or BS subqueues can deadlock the system. Assigning non-priority system and user processes to these subqueues can prevent the execution of critical system processes. Exercise extreme caution in assigning processes to these subqueues.

localattribute The local attribute of the user, as defined at the installation site. This is a double word bit map of arbitrary meaning which might be used to further classify users. While it is not involved in standard MPE security provisions, it is available to processes through the -WHO intrinsic for use in the programmer's own security provisions. The :NEWUSER command checks the local attributes of the user with those of the account. Default is double word 0 (null).

homegroupname The name of an existing group to be assigned as the user's home group. If none is assigned, the user must always specify a group when logging on. Default is that no home group is assigned.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have AM capability to execute this command.

OPERATION

The Account Manager uses the :NEWUSER command to define an account member. When the user is defined, the Account Manager may also assign the user a password, capabilities, and limit the user's use of system resources. Parameters defining these values may also be omitted from the command line; in this case, the defaults will be assigned the user.

EXAMPLE

To define a new user named LHSMITH, assign a password of SMITTY, and a homegroup of HOMEGPX, enter:

```
:NEWUSER LHSMITH;PASS=SMITTY;HOME=HOMEGPX
```

:NEWVSET

Defines private volume sets and classes.

SYNTAX

```
:NEWVSET vsname MEMBERS=vname:type[,vname:type...]
  [,CLASS=vcname:vname[,vname...]]
```

PARAMETERS

vsname	Volume set name, consisting of one to eight alphanumeric characters, beginning with an alphabetic character. This name is also given to the master volume of the set. MPE references *vsname* as *vsname.groupname.acctname*, where *groupname* and *acctname* are the logon group and account.
vname:type	Names of members of the volume set and the types of devices on which the members will reside. A maximum of eight volume names can be specified. The list of volume names must include the master volume set name. The type of disc to accommodate the volume must be specified with each *vsname*. The *type* can be HP 7902A, HP 7905, HP 7906, HP 7911, HP 7912, HP 7914, HP 7920, HP 7925, HP 7933, HP 7935, HP 7945 or HP 9895.
vcname:vname	Volume class name, consisting of one to eight alphanumeric characters, beginning with an alphabetic character. Each volume class consists of a volume class name and an associated list of volume names (*vname*). The list of volume names must be a subset of the volumes composing the volume set, and one of the volume names must be that of the set's master volume.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have Create Volumes (CV) capability to execute this command.

OPERATION

The :NEWVSET command defines private volume sets and classes, making known to MPE the members of the volume set and their corresponding storage types. Once a volume set or volume class is defined, its definition is kept on disc in a volume set definition entry. A physical volume set is not created at the time the volume set is defined.

EXAMPLE

To define a volume set with *vsname* USERSET; with members USERSET (master volume, and MEM2, of *type* HP7920; and class (*vcname:vname*) CLASS1:USERSET,MEM2, enter:

```
:NEWVSET USERSET;MEMBERS=USERSET:HP7920,MEM2:HP7920;CLASS=CLASS1:USERSET,MEM2
```

:OPENQ

Opens the spool queue for a specified logical device or device class. This command is supported by the G.00.00 and later releases of MPE.

SYNTAX

```
:OPENQ {ldev    }
       {devclass}
```

PARAMETERS

ldev	The logical device number of the spooled device.
devclass	The device class name of the spooled device; *devclass* must begin with a letter and consist of eight or fewer alphanumeric characters.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It may be executed only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE mmands.

co

OPERATION

:OPENQ enables the Operator to control the spool queue of a specified device or device class without affecting the operation of the spooler process. It also gives the Operator access to spool queues for which no spooler or physical device exists.

Since spoolfiles can be created faster than they are processed, you may want to issue a :SHUTQ command, to clear the backlog of files in the queue, and then re-open it with an :OPENQ command when the queue is clear.

:OPENQ also serves as an option to the :STOPSPOOL command. The :STOPSPOOL and the :SHUTQ commands, are documented in this section.

EXAMPLE

To open the spool queue for logical device 6, enter:

```
:OPENQ 6
```

:OUTFENCE

Defines the minimum priority an output spoolfile needs in order to be printed.

SYNTAX

```
:OUTFENCE outputpriority [;LDEV=ldev]
```

PARAMETERS

outputpriority	A number between 1 and 14, inclusive. A larger number is more limiting.
ldev	The logical device number of an output device.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It is executable only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

:OUTFENCE controls the processing of all output spoolfiles by establishing a numerical limit, or "fence", that, along with each spoolfile's *outputpriority*, determines whether a file will be printed or deferred. Individual spoolfiles which are READY will be printed only if their *outputpriority* is higher than the current outfence. To prevent any spoolfiles from being printed, set the outfence to 14. To defer a subset of spoolfiles, set the outfence higher than the *outputpriority* of any spoolfile in the group.

You may wish to alter the printing priority of a single file without affecting the entire system. If that is the case, change the *outputpriority* of the specific spoolfile with the :ALTSPOOLFILE command. (Refer to discussion of :ALTSPOOLFILE in this section.)

EXAMPLES

To defer all output spoolfiles except those waiting to be printed by *ldev* 6, usually configured (the system line printer), set the global outfence to 14 and the outfence of *ldev* 6 to 7, as shown below:

```
:OUTFENCE 14
:OUTFENCE 7;LDEV=6
```

To display the new global *outputpriority* and the *outputpriority* of logical device 6, execute the :SHOWOUT command, as in the example below. Note that the summary statistics at the bottom of the listing will immediately reflect the new OUTFENCE, but individual entries for each output spoolfile may not. Once the currently ACTIVE spoolfile is finished, however, no files directed towards a device other than *ldev* 6 will become ACTIVE.

```
:SHOWOUT

DEV/CL   DFID    JOBNUM  FNAME    STATE FRM SPACE RANK PRI #C
6        #0999   #J19    $STDLIST OPENED       512       8  1
6        #01030  #S77    EDLIST   OPENED       512       8  1
SLOWLP   #01029  #S71    OUT      READY        232   1   7  1
20       #01001  #S60    $STDLIST OPENED
11       #01022  #S33    GALLIST  READY        768    1  7  1

11 FILES:
    1 ACTIVE
    1 READY; INCLUDING 1 SPOOFLES, 0 DEFERRED
    9 OPENED; INCLUDING 2 SPOOFLES
    0 LOCKED; INCLUDING 0 SPOOFLES
    4 SPOOFLES:  2024 SECTORS
OUTFENCE = 14
OUTFENCE = 7    FOR LDEV 6
```

To reset the outfence for all output spoolfiles, enter:

```
:OUTFENCE 6
```

:PARTBACKUP

Performs a backup of MPE and all of the files that have been modified since the last full backup for G.01.00 and later releases of the MPE operating system.

SYNTAX

```
:PARTBACKUP [dumptape][,auxlistfile]
```

PARAMETERS

dumptape	Specifies the magnetic tape destination for the backup. If *dumptape* is specified, it must be backreferenced to a previous :FILE command or file equates to the formal file designator DUMPTAPE. The default is TAPE. If there is no device class TAPE, the device class CTAPE will be used.
auxlistfile	Actual file designator of the output file (device) to which all listings requested during the execution of the backup process are written. The default is $STDLIST.

USE

This command may be issued from a Session, Job or Program, but not in BREAK. It is Breakable (suspends execution). It requires System Supervisor (OP) capability.

OPERATION

This command starts the backup process and copies MPE and all of the files that have been modified since the last full backup. The two commands :PARTBACKUP and :FULLBACKUP provide an alternative to going through the :SYSDUMP dialog. The only user interaction required with these commands is mounting tapes or serial discs and replying to any mount requests. These replies are unnecessary, however, if the devices are configured as AUTO-REPLY.

During the backup, PARTBACKUP will display messages at one minute intervals to report it progress:

```
STORE OPERATION IS 27% COMPLETE
```

EXAMPLES

In the following example the first :FILE command defines DUMP as a magnetic tape file. The second :FILE command defines LIST as a line printer. The TAPE and LP parameters are device class names arbitrarily defined during the previous system configuration.

```
:FILE DUMP;DEV=TAPE
:FILE LIST;DEV=LP
:PARTBACKUP *DUMP,*LIST
```

The following will be displayed at the Console:

```
?11:08/#s278/89/LDEV# FOR "DUMP" ON TAPE (NUM)?
```

In the following example, a file equation with the formal file designator of DUMPTAPE was previously entered (e.g. :FILE DUMPTAPE;DEV=SYSTAPE). As a result, the user may omit the first parameter, and the backup will go to the device class SYSTAPE.

```
:PARTBACKUP,*LIST
```

The following will be displayed at the Console:

```
?11:08/#S278/89/LDEV# FOR "DUMPTAPE" ON SYSTAPE (NUM)?
```

In the following example, the *dumptape* is not specified, and there is no file equation for DUMPTAPE. A temporary file equation (FILE DUMPTAPE;DEV=TAPE) will be used.

```
:PARTBACKUP
```

The following will be displayed at the Console:

```
?11:08/#S278/89/LDEV# FOR "DUMPTAPE" ON TAPE (LDEV)?
```

:PASCAL

Compiles a PASCAL program. PASCAL/3000 is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:PASCAL [textfile][,[uslfile][,listfile]]
 [;INFO=quotedstring]
```

PARAMETERS

textfile Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is PASTEXT. Default is $STDIN.

uslfile Actual file designator of the USL file to which the object code is stored. This can be any binary output file with a file code of USL or 1024. Its formal file designator is PASUSL. If the *uslfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS. If entered, this parameter indicates that the USL file was created in one of four ways:

- By using the MPE :SAVE command to save the default USL file $OLDPASS, created during a previous compilation.
- By building the USL with the MPE Segmenter -BUILDUSL command. Refer to the MPE Segmenter Reference Manual (30000-90011).
- By creating a new USL file and specifying the MPE :BUILD command with a file code of USL or 1024.
- By specifying a nonexistent *uslfile* parameter, thereby creating a permanent file of the correct size and type.

listfile Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is PASLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (PASTEXT, PASUSL, and PASLIST) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the :FILE command.

quotedstring A sequence of characters between two single quotation marks (apostrophes) or between two double quotation marks. You may use the delimiter as part of the string so long as the delimiter appears twice. Any occurrence of two single quotation marks in a row, or two double quotes in a row, is considered part of the string, and, therefore, not the terminating delimiter.

INFO=*quotedstring* is used in the PASCAL programming language to pass initial compiler options to a program. PASCAL/3000 brackets the *quotedstring* with dollar signs and places it before the first line of source code in the text file. For more information, refer to the :RUN command in this section.

USE

This command may be issued from a Session or a Job. It may not be used from a Program or in BREAK. It is Breakable (suspends execution).

OPERATION

The :PASCAL command compiles a PASCAL program and stores the object code in a User Subprogram Library (USL) file on disc. If *textfile* is not specified, MPE expects the source program to be entered from your standard input device. If you do not specify a *listfile*, MPE sends the program listing to your standard list device and identifies it by the formal file designator, PASLIST.

If you create the USL prior to compilation, you must specify a file code of USL or 1024. If you omit the *uslfile* parameter, the object code is saved in the temporary file domain as $OLDPASS. To keep it as a permanent file, you must save $OLDPASS under another name.

EXAMPLES

The following example compiles a PASCAL program entered from the standard input device and stores the object code in the USL file $OLDPASS. The listing is then sent to the standard list device.

```
:PASCAL
```

The next example compiles a PASCAL program contained in the disc file PASCSRC, and stores the object code in the USL file PASCOBJ. The program listing is stored in the disc file LISTFILE.

```
:PASCAL PASCSRC,PASCOBJ,LISTFILE
```

ADDITIONAL DISCUSSION

PASCAL/3000 Reference Manual (32106-90001)
MPE Segmenter Reference Manual (30000-90011)

:PASCALGO

Compiles, prepares, and executes a PASCAL/3000 program. PASCAL is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:PASCALGO [textfile][;listfile]
   [;INFO=quotedstring]
```

PARAMETERS

textfile Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is PASTEXT. Default is $STDLIST.

listfile Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is PASLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (PASTEXT , and PASLIST) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the :FILE command.

quotedstring A sequence of characters between two single quotation marks (apostrophes) or between two double quotation marks. You may use the delimiter as part of the string so long as the delimiter appears twice. Any occurrence of two single quotation marks in a row, or two double quotes in a row, is considered part of the string, and, therefore, not the terminating delimiter.

INFO=*quotedstring* is used in the PASCAL programming language to pass initial compiler options to a program. PASCAL/3000 brackets the *quotedstring* with dollar signs and places it before the first line of source code in the text file. For more information, refer to the :RUN command in this section.

USE

This command may be issued from a Session or Job. It may not be used from a Program, or in BREAK. It is Breakable (suspends execution).

OPERATION

The :PASCALGO command compiles, prepares, and executes a PASCAL program. If *textfile* is omitted, MPE expects input from your standard input device. If you do not specify *listfile*, MPE sends the program listing to the formal file designator PASLIST (default is $STDLIST).

The USL file created during the compilation is the system defined temporary file $OLDPASS, which is passed directly to the Segmenter. It can only be accessed if you do not use the default for *progfile*. This is because the Segmenter also uses $OLDPASS to store the prepared program segments, overwriting any existing temporary file of the same name.

EXAMPLES

To compile, prepare, and execute a PASCAL program entered from your standard input device, with the program listing sent to your standard list device, enter:

```
:PASCALGO
```

To compile, prepare, and execute a PASCAL program from the disc file PASCSRC and send the program listing to the file LISTFILE, enter:

```
:PASCALGO PASCSRC,LISTFILE
```

ADDITIONAL DISCUSSION

PASCAL/3000 Reference Manual (32106-90001)
MPE Segmenter Reference Manual (30000-90011)

:PASCALPREP

Compiles and prepares a PASCAL/3000 program. PASCAL is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:PASCALPREP [textfile][,progfile][,listfile]
   [;INFO=quotedstring]
```

PARAMETERS

textfile Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is PASTEXT. Default is $STDIN.

progfile Actual file designator of the program file to which the prepared program segments are written. You will get the best results by omitting the *progfile* parameter. When *progfile* is omitted, the MPE Segmenter creates the program file, which will be stored in the temporary file domain as $OLDPASS. If you do create your own program file, you must do so in one of two ways:

- By using the MPE :BUILD command, and specifying a file code of 1029 or PROG, and a *numextents* value of 1. This file is then used by the :PREP command.
- By specifying a nonexistent file in the *progfile* parameter, in which case a job/session temporary file of the correct size and type is created.

listfile Actual file designator of the file to which the program listing is written. This can be any ASCII output file. Formal file designator is PASLIST. Default is $STDLIST.

NOTE

The formal file designators used in this command (PASTEXT and PASLIST) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the :FILE command.

quotedstring A sequence of characters between two single quotation marks (apostrophes) or between two double quotation marks. You may use the delimiter as part of the string so long as the delimiter appears twice. Any occurrence of two single quotation marks in a row, or two double quotes in a row, is considered part of the string, and, therefore, not the terminating delimiter. INFO=*quotedstring* is used in the PASCAL programming language to pass initial compiler options to a program. PASCAL/3000 brackets the *quotedstring* with dollar signs and places it before the first line of source code in the text file. For more information, refer to the :RUN command in this section.

USE

This command may be issued from a Session or Job. It may not be used from a Program, or in BREAK. It is Breakable (suspends execution).

OPERATION

The :PASCALPREP command compiles and prepares a PASCAL program into a program file on disc. If you do not specify *textfile*, MPE expects input from the current input device. If you do not specify *listfile*, MPE sends the listing output to the formal file designator PASLIST (default $STDLIST). The USL file $OLDPASS, created during compilation, is a temporary file passed directly to the Segmenter. You may access it only if you do not use the default for *progfile*. This is because the Segmenter also uses $OLDPASS to store the prepared program segments, overwriting any existing temporary file of the same name.

EXAMPLES

The following example compiles and prepares a PASCAL program entered through your standard input device and stores the prepared program segments in the file $OLDPASS. The listing will be printed on your standard list device.

```
:PASCALPREP
```

To compile and prepare a PASCAL source program from the source file PASCSRC, storing it in PASCPROG, and sending the listing to your standard list device, enter:

```
:PASCALPREP PASCSRC,PASCPROG
```

ADDITIONAL DISCUSSION

PASCAL/3000 Reference Manual (32106-90001)
MPE Segmenter Reference Manual (30000-90011)

:PREP

Prepares program from a User Subprogram Library (USL) file onto a program file.

SYNTAX

```
:PREP uslfile,progfile
   [;ZERODB]                          [;CAP=capabilitylist]
   [;PMAP]                            [;RL=filename]
   [;MAXDATA=segsize]                 [;PATCH=patchsize]
   [;STACK=stacksize]                 [;NOSYM]
   [;DL=dlsize]                       [{;FPMAP   }]
                                      [{;NOFPMAP}]
```

PARAMETERS

uslfile — Actual file designator of User Subprogram Library (USL) file into which program has been compiled.

progfile — Actual file designator of program file onto which prepared program segments are written. This can be any binary output file created in one of two ways:

- By using the MPE :BUILD command to create a new file and specifying a file code of PROG or 1029, and one extent.
- By specifying a nonexistent file in the *progfile* parameter, in which case a file of the correct size and type is created. This file is a temporary job file.

ZERODB — Request to initialize to zero the initially defined, user managed (DL-DB) area of the stack, as well as the uninitialized portions of the DB-Q (initial). Default is that these areas are not affected.

PMAP — Request to produce a descriptive listing of the prepared program to a file whose formal file designator is $SEGLIST. If no :FILE command is found referencing $SEGLIST, the listing is produced on $STDLIST. Default is no listing.

segsize — Maximum permitted stack area (Z-DL) in words. This parameter should be included when it is expected that the size of DL-DB or Z-DB areas will be changed during program preparation or execution. Regardless what you specify, MPE may change the *segsize* to accommodate table overflow conditions.

If you prepare your program with *segsize* less than the configured minimum, the value is rounded up to the minimum or the amount needed by the program (as calculated by the Segmenter). The maximum actual *segsize* permitted a program is 31,232 words. You may prepare your program with a *segsize* larger than necessary so long as this maximum is not exceeded. If the specified *segsize* does exceed the maximum, it will be rounded down to 31,232 words.

stacksize — Size of initial local data area (Z-Q initial) stack, in words. This value, if specified, must be between 511 and 32767 words. This parameter overrides the default *stacksize* estimated by the MPE Segmenter.

dlsize — DL-DB area to be initially assigned to stack. This area is of interest mainly in programmatic applications. Due to system logging considerations, the DL-DB area is always rounded upward so that the distance from the beginning of the stack data segment to the DB-address is a multiple of 128 words. The value you specify must be within -1 and 32767 words. The default is estimated by the MPE Segmenter.

capabilitylist — Capability class attributes associated with a program, specified as two-character mnemonics. If more than one mnemonic is specified, each must be separated from its neighbor by a comma. The mnemonics are:

IA	Interactive Access
BA	Local Batch Access
PH	Process Handling
DS	Data Segment Management
MR	Multiple Resource Management
PM	Privileged Mode

Note that you can specify only those capabilities as assigned by the Account Manager or System Manager. Default is IA and BA.

filename — Actual file designator of the relocatable library (RL) file to be searched to satisfy external references during preparation. This can be any permanent binary file of type RL. It need not belong to your logon group, nor have a reserved, local name. This file, to which you must have READ and LOCK access, yields a single segment that is incorporated into the segments of the program file. For more information, refer to the MPE Segmenter Reference Manual (30000-90011). Default is that no library is searched.

patchsize — Specifies the size of the patch area. This size will apply to all segments within the program file. The value you specify must be within -1 and 16380 words.

NOSYM — Suppresses the symbolic debug option. Refer to the HPToolset Reference Manual (32350-90001).

FPMAP or NOFPMAP — Includes or excludes the internal PMAP information. FPMAP is a request to have internal PMAP information included in the program. NOFPMAP excludes PMAP information from the program when the system FPMAP or job/session FPMAP is on. If the symbolic debug option is invoked, default is FPMAP. Otherwise the default is NOFPMAP.

USE

This command may be issued from a Session or Job. It may not be used from a Program, or in BREAK. It is Breakable (suspends execution).

OPERATION

The :PREP command prepares a compiled source program for execution. Unless you prepare the program into a previously created program file, :PREP creates a temporary program file for you. It is a good idea to specify a nonexistent program file when you issue the :PREP command. This way, MPE will create a file of the optimum size and characteristics. (Refer to the "EXAMPLE" section.)

A compiled program is prepared by searching a relocatable library (RL) to satisfy references to external procedures required by the program. When the program is prepared, such procedures are linked to the program in the resulting program file. To use a relocatable library (RL), you must have Read and Lock access to it.

EXAMPLE

The following example prepares a program from the USL file USLX and stores it in the program file PROGX. Since you are creating PROGX when you issue the :PREP command, it is a temporary file, and must subsequently be saved to remain in the permanent file domain.

```
:PREP USLX,PROGX
:SAVE PROGX
```

In this case, the MPE Segmenter created your program file. Although you will get the best results this way, you can use the :BUILD command to create your own permanent program file. When you do so, be sure to specify a file code of "PROG" or "1029" and a *numextents* parameter value of 1, as shown below:

```
:BUILD PROGX;CODE=PROG;DISC=,1
:PREP USLX,PROGX
```

To prepare a program from the USL file named USLZ and store it in a program file named PROGZ, list the prepared program, assign a *stacksize* of 511 words, and limit access to PROGZ to those users having IA, BA, PH and DS capability enter:

```
:PREP USLZ,PROGZ;PMAP;STACK=511;CAP=IA,BA,PH,DS
```

ADDITIONAL DISCUSSION

For a complete tutorial refer to the MPE Segmenter Reference Manual (30000-90011)

Prepares and executes a compiled program.

SYNTAX

```
:PREPRUN uslfile[,entrypoint]
   [,NOPRIV]
   [,PMAP]
   [,DEBUG]
   [,LMAP]
   [,MAXDATA=segsize]
   [,PARM=parameternum]
   [,STACK=stacksize]
   [,DL=dlsize]
        {G}
   [,LIB={P}]
        {S}
   [,CAP=capabilitylist]
   [,RL=filename]
   [,NOCB]
   [,INFO=quotedstring]
             {*formaldesig}
   [,STDIN=[{fileref      }]
             {$NULL       }
              {*formaldesig}
   [,STDLIST=[{fileref      }][,NEW]]
              {$NULL        }
   [,PATCH=patchsize]
   [,NOSYM]
   [{,FPMAP   }]
   [{,NOFPMAP}]
```

PARAMETERS

uslfile — Actual file designator of the USL file to which the program has been compiled.

entrypoint — Contains a character string, terminated by a blank, specifying the entry point (label) in the program where execution is to begin when the program is executed. The *entrypoint* parameter may be the primary entry point or any secondary entry point in the program's outer block. Default is primary entry point.

NOPRIV — Declaration that the program segments will be placed in non privileged (user) mode. This parameter is for programs prepared with Privileged Mode capability and make them accessible to non privileged users. Normally, program segments containing privileged instructions are executed in Privileged Mode only if the program was prepared with Privileged Mode (PM) capability class. (A program containing legally compiled privileged code, placed in nonprivileged mode, may abort when an attempt is made to execute it.)

If NOPRIV is specified, all segments are placed in nonprivileged mode. (Library segments are not affected because their mode is determined independently.) Default is that segments of a Privileged mode program will remain in Privileged Mode.

PMAP

Request to produce a descriptive listing of the prepared program to a file whose formal file designator is $SEGLIST. If $SEGLIST is not found in a :FILE command, the listing is produced on the current list device. Default is no listing.

DEBUG

Request to issue a DEBUG call before the first executable instruction of the program. Unless the user has Read and execute access to the program file, this parameter is ignored. If Privileged Mode (PM) capability has been assigned, the user is put into privileged mode DEBUG. If not, the user is put into user mode DEBUG. Default is that the DEBUG call is not issued.

LMAP

Request to produce a descriptive listing of the allocated (loaded) program to a file whose formal file designator is LOADLIST. If no :FILE command referencing LOADLIST is found, the listing is produced on $STDLIST. Default is no listing.

ZERODB

Request to initialize to zero the initially defined user managed (DL-DB) area, and uninitialized portions of the DB-Q (initial) area. Default is that these areas are not affected.

segsize

Maximum permitted stack area (Z-DL) in words. This parameter should be included when you expect that the size of DL-DB or Z-DB areas will be changed during program preparation or execution. Regardless what you specify, MPE may change the *segsize* to accommodate table overflow conditions.

If you prepare your program with a *segsize* less than the configured minimum, the value is rounded up to the minimum or the amount needed by the program (as calculated by the Segmenter). The maximum actual *segsize* permitted a program is 31,232 words. You may prepare your program with a *segsize* larger than necessary so long as this maximum is not exceeded. If the specified *segsize* does exceed the maximum, it will be rounded down to 31,232 words.

parameternum

An integer containing a parameter to be passed to the new program (accessed through Q-4 of the outer block).

stacksize

Size of local data area, Z-Q (initial), in the stack, in words. If it is specified, this value must be between 511 and 32,767 words. The default is estimated by the MPE Segmenter.

dlsize

DL-DB area to be initially assigned to stack. Due to system logging considerations, the DL-DB area is always rounded upward, so that the distance from the beginning of the stack data segment to the DB-address is a multiple of 128 words. The value you specify must be between -1 and 32,767 words. The default is estimated by the MPE Segmenter.

G, P, or S

Searches the segmented procedure libraries of the program file's group and account. The G option searches the group library, account library, then the system library. The P option searches the account library then system library. The S option searches the system library for external references to segmented procedures. Default is S.

capabilitylist

Capability class attributes associated with the program, specified in two-character mnemonics. If more than one mnemonic is specified, each must be separated from its neighbor by a comma. The mnemonics are:

IA	Interactive Access
BA	Local Batch Access
PH	Process Handling
DS	Data Segment Management
MR	Multiple Resource Management
PM	Privileged Mode

You can specify only those attributes that you possess through assignment by the Account Manager or the System Manager. Default is IA and BA.

filename

Actual file designator of the relocatable library (RL) file to be searched to satisfy external references during preparation of the program. This can be any permanent file of type RL, to which you must have READ and LOCK access. It need not belong to the logon group, nor does it require a reserved, local name. This file yields a single segment that is incorporated into the segments of the program file. Refer to the MPE Segmenter Reference Manual (30000-90011) for a description of RL files. Default is that no library is searched.

NOCB

Request that the file system not use stack segment (PCBX) for its control blocks, even if sufficient space is available. This permits you to expand your stack (with the DLSIZE or ZSIZE intrinsics) to the maximum possible limit at a later time. It will, however, cause the File Management System to operate more slowly for this program.

quotedstring

A sequence of characters between two single quotation marks (apostrophes) or two double quotation marks. You may use the delimiting character as part of the string so long as the delimiter appears twice. Any occurrence of two single quotes, or two double quotes in a row, is considered part of the string, and, therefore, not the terminating delimiter.

;INFO=*quotedstring* is used in some programming languages (e.g. COBOLLII, Pascal) to pass compiler options to a program. These options will appear before the first line of source code in the text file. For more information, refer to the :RUN command in this section.

$STDIN

This parameter allows the user to specify the file to be used as $$STDIN by the program being executed. If omitted, or if nothing is specified after the equal sign, such as ";$STDIN=", then $STDIN defaults to the job or session's standard input device. You may use one of the following subparameters with $STDIN=:

**formaldesig*	The formal file designator for a file previously specified in a file equation.
fileref	The name of an existing permanent disc file.

$NULL — The actual file designator of a system defined file that is always treated as an empty file.

When referenced by a program as $STDIN, that program receives only an end-of-file indication when accessed. When referenced by a program as $STDLIST, the associated write request is accepted by MPE, but no physical output is actually performed. Thus, $NULL can be used to discard unneeded output from an executing program.

STDLIST — This parameter allows the user to specify the file to be used as $STDLIST by the program being executed. If $STDLIST is omitted, or if nothing is specified after the equal sign, such as ";$STDLIST=", then $STDLIST defaults to the job or session's standard list device. This parameter has the same subparameters as $STDIN, but you may also specify the keyword NEW.

NEW — The name to be assigned to a job/session temporary disc file created with the system defaults. The system defaults of the new file are fixed length ASCII 132-byte records with a maximum file size of 1023 records.

patchsize — Specifies the size of the patch area. This size will apply to all segments within the program file. The value you specify must be within -1 and 16,380 words.

NOSYM — Suppresses the symbolic debug option. Refer to the HPToolset Reference Manual (32350-90001) for more information.

FPMAP or NOFPMAP — Includes or excludes the internal PMAP information. FPMAP is a request to have internal PMAP information included in the program. NOFPMAP excludes PMAP information from the program when the system FPMAP or job/session FPMAP is on. If the symbolic debug option is invoked, default is FPMAP. Otherwise the default is NOFPMAP

USE

This command may be issued from a Session or Job. It may not be used from a Program, or in BREAK. It is Breakable (suspends execution).

OPERATION

The :PREPRUN command prepares and executes a program compiled in a USL file. Both relocatable (RL) and segmented (SL) libraries are searched during the preparation process to satisfy external references.

The USL file created during compilation is a system defined temporary file, $OLDPASS, which is passed directly to the Segmenter. It can be accessed only if you do not use the default for *progfile*. This is because the Segmenter also uses the file $OLDPASS to store the prepared program segments, overwriting any existing temporary file of the same name.

EXAMPLES

To prepare and execute a program from the USL file XUSL, with no special parameters declared, enter:

```
:PREPRUN XUSL
```

To obtain a descriptive listing of the prepared program, and a listing of the allocated (loaded) program, enter:

```
:PREPRUN XUSL;PMAP;LMAP
```

To prepare and execute a program form the USL file UBASE that begins execution at the entry point RESTART, that has a stacksize of 800 words and searches an RL file named LIBA, enter:

```
:PREPRUN UBASE,RESTART;STACK=800;RL=LIBA
```

The following example prepares and runs a program with $STDIN set to the existing disc file INPUT. $STDLIST is set to the line printer:

```
:FILE LPFILE;DEV=LP
:PREPRUN TESTPROG;MAXDATA=10000;$STDIN=INPUT;$STDLIST=*LPFILE
```

The next example also uses the "$STDIN=" and "$STDLIST=" parameters to prepare and run a program. This time, a file equation is used to set $$STDIN to INPT, and to set $STDLIST to the temporary disc file RESULTS (which is automatically created by the :RUN command).

```
:FILE INFILE=INPT,OLD;
:PREPRUN TESTPROG;DEBUG;$STDIN=*INFILE;$STDLIST=RESULTS,NEW
```

The following example of the :PREPRUN command uses the "INFO=" parameter to pass a string to the program:

```
:PREPRUN MYPROG;MAXDATA=2000;INFO="A test with ""and"" characters"
```

Note that the delimiting character is doubled within the string so that it will appear on the printout as follows:

```
A test with "and" characters
```

ADDITIONAL DISCUSSION

For a full tutorial, refer to the MPE Segmenter Manual (30000-90011)

:PTAPE

Reads a paper tape without X-OFF control.

SYNTAX

```
:PTAPE filename
```

PARAMETERS

filename Name of existing ASCII file on disc, to which input data is to be written from a paper tape. Normally, this is a file with variable length records; the record size specified must be large enough to contain the largest paper tape record.

USE

This command may be issued from a Session, Program, or in BREAK. It may not be used in a Job. It is not Breakable.

OPERATION

The :PTAPE command permits programs to read input from a terminal that is coupled to a paper tape reader/punch when the X-OFF control character is not punched on the tape. (The X-OFF character, on a tape, turns off the tape reading mechanism. This character normally appears following the carriage return/line feed character combination used to delimit each record.) The :PTAPE command transfers all data from the tape to an ASCII file (named in the *filename* parameter) on disc. It stores all records to the disc file in the order read from the tape, but deletes all carriage return and line feed characters. (If the tape does contain X-OFF characters, the operation also deletes these. The X-OFF characters are not ignored by the tape reader, however, and so you must restart the reader after each X-OFF is read.) The input terminates when the Y^C character is encountered on the tape or entered from the terminal, Control returns to the keyboard. The input data now can be read from the disc file by any program.

Note that the :PTAPE command is required only for input from HP tape readers associated with terminal keyboards.

You cannot use the :PTAPE command to copy tapes containing more than 32,767 bytes of information. If you attempt to copy such tapes, MPE will read only the first 32,767 bytes on the tape and will then terminate the operation.

EXAMPLE

To copy input from a paper tape to a disc file named TAPEFILE first, create an appropriate file with the MPE :BUILD command, and follow it with the :PTAPE command:

```
:BUILD TAPEFILE;REC=80,3,V,ASCII
:PTAPE TAPEFILE
```

Deletes file from system.

SYNTAX

```
:PURGE filereference[;TEMP]
```

PARAMETERS

filereference Actual file designator of file to be deleted, in this format:

filename[/*lockword*][.*groupname*[.*acctname*]]

To delete the file, you must have WRITE access to it.

TEMP Indicates that the file is a temporary file in the job/session temporary file domain. You must enter this parameter to delete a temporary file. Default is that a permanent file is assumed.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

This command deletes a disc file from the system. If the volume set on which the file resides is not mounted, this command will generate an implicit mount request. If a mount follows, the purge will be done.

If the file does not exist in the domain specified (temporary or permanent), the following message is displayed:

```
FILE filename NOT FOUND, NO PURGE DONE. (CIWARN 383)
```

To purge a KSAM file, run KSAMUTIL.PUB.SYS, using the >PURGE subcommand. Purging KSAM files with the MPE :PURGE command is not recommended.

EXAMPLES

To delete the permanent file PFILE, enter:

```
:PURGE PFILE
```

To delete the temporary file TFILE, enter:

```
:PURGE TFILE,TEMP
```

ADDITIONAL DISCUSSION

MPE File System Reference Manual (30000-90236)

:PURGEACCT

Removes an account and its groups and users from the system directory or from the specified volume set's directory.

SYNTAX

```
:PURGEACCT acctname [;VS=volset]
```

PARAMETERS

acctname Name of the account to be deleted. This name must contain from one to eight alphanumeric characters, beginning with an alphabetic character.

volset Volume set or volume class reference which, when fully qualified, is in the following form:

vcsid.groupname.acctname

The *vcsid* parameter refers to a previously defined volume set or volume class. When the *volset* is specified, the volume set or volume class must be mounted, or the :PURGEACCT command will fail. When *volset* parameter is specified, the account will be removed from the volume set directory, and not the system directory.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have System Manager (SM) capability to execute this command.

OPERATION

The System Manager uses the :PURGEACCT command to eliminate an entire account from the system. When :PURGEACCT is executed during a session, MPE displays a verification request to ensure that the wrong account is not deleted accidentally. Respond "YES" or "NO" to the message:

ACCT *acctname* TO BE PURGED?

No verification message is printed when the :PURGEACCT is entered in a job.

The :PURGEACCT command removes every user not currently logged on and every group/file not in use. The order in which entries are purged is: users, volume set definitions, files, groups, and finally the account. If the command is executed while the account is in use, the account remains on the system: active users, groups, and files are not purged from the account. To completely purge an account, you must execute :PURGEACCT when the account is inactive.

CAUTION

Do Not Attempt To Purge The SYS Account. The account SYS cannot be completely purged, but you can destroy critical files by attempting to do so. If you execute :PURGEACCT SYS, all groups except PUB will be purged, all users except the System Manager will be purged, and all inactive files and system files in the PUB group will be purged.

EXAMPLE

To remove an account named ACCT1 , enter:

```
:PURGEACCT ACCT1
ACCT ACCT1 TO BE PURGED? YES
:
```

:PURGEGROUP

Removes a group (and all files belonging to it) from the system, or from the specified volume set directory.

SYNTAX

```
:PURGEGROUP groupname [;VS=volset]
```

PARAMETERS

groupname — Name of the group in the logon account to be removed. This name must contain from one to eight alphanumeric characters, beginning with an alphabetic character.

volset — Volume set or volume class reference which, when fully qualified, is in the form, *vcsid.groupname.acctname*, where *vcsid* refers to a previously defined volume set or volume class.

If *volset* is specified, the volume set or volume class must be mounted or the :PURGEGROUP command will fail. When the *volset* parameter is specified, the group will be removed from the volume set directory, and not the system directory.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have Account Manager (AM) capability to execute this command.

OPERATION

Account Managers use the :PURGEGROUP command to delete a group from their account. When the command is executed during a session, MPE displays a verification request. Respond "YES" or "NO" to the message:

```
GROUP groupname TO BE PURGED?
```

No verification message is printed if the :PURGEGROUP command is entered in a job.

If the group resides on a private volume, the command will succeed only if the group's home volume set is mounted.

The order in which entries are purged is: volume set definitions, files, and finally the group. If no files in the group are in use, and the group itself is not in use, the :PURGEGROUP command will remove the entire group. Otherwise, only inactive files will be removed. To completely purge the group in this case, re-enter the :PURGEGROUP command when neither the group nor its files are in use.

CAUTION

Do Not Attempt To Purge The Pub Group of The SYS Account. The public group of the system account, PUB.SYS , cannot be completely purged. If you specify this group in the *groupname* parameter, all non-system and inactive files will be purged, which will seriously impair the proper functioning of the entire system.

EXAMPLE

To purge a group named GROUP1 , enter:

```
:PURGEGROUP GROUP1
GROUP GROUP1 TO BE PURGED? YES
```

:PURGEUSER

Removes a user from an account.

SYNTAX

```
:PURGEUSER user
```

PARAMETERS

user	Name of the user to be deleted.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have Account Manager (AM) capability to execute this command.

OPERATION

Account Managers use the :PURGEUSER command to delete a user from their account. You are asked to verify the command only when it is executed during a session, and not from a job. To do so, respond "YES" or "NO" to the message:

```
USER user TO BE PURGED?  (YES/NO)
```

An attempt to purge a user currently logged onto the system will fail, and an explanatory message will be displayed. This user will not be affected until the next logon. An attempt to purge MANAGER.SYS will always fail, since this user can never be purged.

If files created by a purged user remain after the user is purged from the system, the System Manager can remove them with the the :PURGEACCT command, or the Account Manager can eliminate them by executing :PURGEGROUP.

EXAMPLE

To remove a usernamed USER1 , enter:

```
:PURGEUSER USER1
USER USER1 TO BE PURGED? YES
```

:PURGEVSET

Deletes an existing volume set.

SYNTAX

```
:PURGEVSET vsname
```

PARAMETERS

vsname	Name of an existing volume set or volume class.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have Create Volumes (CV) capability to execute this command.

OPERATION

The :PURGEVSET command will not delete a volume set if it is currently in use; that is, if an explicit or implicit mount, has been issued. When you enter :PURGEVSET during a session, MPE displays the following verification message, to which you must respond "YES" or "NO".

```
VOLUME SET/CLASS DEFINITION vsname TO BE PURGED?  (YES/NO)
```

EXAMPLE

To purge the volume set, USERSET , enter:

```
:PURGEVSET USERSET
VOLUME SET/CLASS DEFINITION USERSET TO BE PURGED? (YES/NO) YES
```

:RECALL/=RECALL

Displays all pending Console REPLY messages.

SYNTAX

```
:RECALL
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be issued only from the Console.*

* Any user may execute :RECALL. A^C =RECALL may only be executed at the Console.

EXAMPLES

To display all pending System Console messages which require a :REPLY response, enter:

```
:RECALL
REPLIES PENDING:
10:05/#J19/15/LDEV # FOR "L00576" ON TAPE1600 (NUM)?
```

If there are replies pending, the text of the request(s) will be displayed on the Console as shown above. If there are no replies pending, the following message will appear on the Console:

```
:RECALL
NO REPLIES PENDING (CIWARN 3020)
```

NOTE

The =RECALL command should be used in the event that the :RECALL command is ineffective, or when a job or subsystem is being executed from the Console.

:REDO

Allows you to edit a command entry.

SYNTAX

```
:REDO
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, or in BREAK. It may not be used from a Program. It is Breakable (aborts execution).

OPERATION

The :REDO command has two purposes: to correct errors in the last command you entered; and two, to change the last command you entered to a new, correct command, thereby eliminating the need to completely re-enter the command.

When the :REDO command is entered, MPE displays the command to be modified and enters a mode similar to that of EDIT/3000. To edit the command, use the space bar on the terminal keyboard to position the cursor under the character(s) to be modified, then enter one of the following subcommands:

D — Delete. Deletes the character above the D. If D is repeated, each character above each D is deleted.

I — Insert. Inserts one or more characters immediately preceding the character above the I. The D and I subcommands can be used in conjunction to delete characters, then insert new characters.

R — Replace. Replaces the characters above the R with new characters. If one character is entered, the character above the R is replaced; if two characters are entered, two characters (the character above the R and the character to the right) are replaced. R is the default subcommand, except when the letter to be replaced is one of the other subcommands.

U — Undo. Cancels the effect of the previous D, I, or R subcommand. Entering a U, (RETURN), then another U (RETURN), cancels all previous subcommands for this :REDO command and restores the line being corrected to its original form.

Until you press (RETURN), each of these subcommands is available to you to edit the command line.

EXAMPLES

To correct an error by inserting characters, use the I subcommand:

```
:SHOW SP
 ^
UNKNOWN COMMAND NAME.  (CIERR 975)
:REDO
SHOW SP
   IO                  ** Enter the correction and press (RETURN). **
SHOWOUT SP             ** Press (RETURN) to execute the correction. **
NO SUCH FILE(S)
OUTFENCE = 3
```

To replace a character with another, use the R subcommand:

```
:FILE A;REC=,,G;DISC=10000,16,1;SAVE
              ^                          (CIERR 273)
:EXPECTED RECORD FORMAT OF F, V OR U.
:REDO
FILE A;REC=,,G;DISC=10000,16,1;SAVE      ** MPE displays command.        **
            RF                           ** Replace G with F.            **
:FILE A;REC=,,F;DISC=10000,16,1;SAVE     ** Corrected command displayed. **
                                         ** Press (RETURN) to execute.    **
```

Because R is the default subcommand, you can replace the letter G by entering only the character F in the correct position:

```
:REDO                                    ** MPE displays command.        **
FILE A;REC=,,G;DISC=10000,16,1;SAVE      ** Replace G with F.            **
            F                            ** Corrected command displayed. **
FILE A;REC=,,F;DISC=10000,16,1;SAVE      ** Press (RETURN) to execute.    **
:
```

: REFUSE

Disables jobs/sessions and/or data on a designated device.

SYNTAX

```
:REFUSE [JOBS,] [ldev]
        [DATA,]
```

PARAMETERS

JOBS	Disables the :JOB (or :HELLO) command from the designated device.
DATA	Disables the :DATA command from the designated device.
ldev	The logical device number of the device for which :JOB (or HELLO) and :DATA commands are refused.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be executed only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

:REFUSE prevents a device from automatically recognizing and accepting one or more of the three commands users execute to introduce jobs or sessions: :JOB, :HELLO, and :DATA. The JOBS parameter in the :REFUSE command refers to both jobs and sessions. If neither the JOBS or DATA parameter is supplied, both :JOB (or :HELLO) and :DATA commands are refused. To undo the effect of the :REFUSE command, use :ACCEPT.

EXAMPLES

To prevent logical device 35 from recognizing the :DATA command, enter:

```
:REFUSE DATA,35
```

To prevent both jobs and data recognition on logical device 35 enter:

```
:REFUSE 35
```

:RELEASE

Removes all security provisions for a file. The file will remain unsecured until the :SECURE command is issued.

SYNTAX

```
:RELEASE filereference
```

PARAMETERS

filereference Actual file designator of the file whose security provisions are to be suspended, written in this format:

filename[/*lockword*][.*groupname*[.*acctname*]]

If the file has a lockword, it must be specified. If *groupname.acctname* is not specified, MPE assumes the logon group and account.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

This command removes all security provisions for a file at all levels (file, group, and account) and allows any user in the system unlimited access to the file. The exception is privileged files: they cannot be released. Other users will have access to the file until its original security provisions are restored with the :SECURE command. The file remains accessible after the job or session ends. It also remains accessible after a system failure, unless the system is restarted with a SYSDUMP tape created before the :RELEASE command was issued for the file. This command can be used only for permanent files on disc whose labels identify you as the creator of the file. In addition, the :RELEASE command will fail if the group's home volume set is not mounted. When the default MPE security provisions are in effect, the file must be in your logon account and must belong to your logon or home group.

The :RELEASE command does not affect the file's lockword, if one was assigned. To check whether a file is secure or released, use the LISTDIR5 utility if you have version G.00.00 or later releases of the operating system, or the LISTDIR2 utility if you have an earlier (E/F.00.00 version of MPE):

```
:RUN LISTDIR2.PUB.SYS

LISTDIR2 G.02.00 (C) HEWLETT-PACKARD CO., 1983
TYPE 'HELP' FOR AID

>LISTF FILE1

**********
FILE: FILE1.MAC.TECHPUBS

FCODE: 0              FOPTIONS: STD,ASCII,FIXED
BLK FACTOR: 4         CREATOR: **
REC SIZE: 88(B)       LOCKWORD: **
BLK SIZE: 176(W)      SECURITY--READ:    ANY
EXT SIZE: 4(S)                WRITE:   ANY
# REC: 4                      APPEND:  ANY
# SEC: 4                      LOCK:    ANY
# EXT: 1                      EXECUTE: ANY
MAX REC: 4                  **SECURITY IS OFF     ** FILE1 is released. **
MAX EXT: 1            COLD LOAD ID: %21604
# LABELS: 0           CREATED: FRI,  9 MAR 1984
MAX LABELS: 0         MODIFIED: FRI,. 9 MAR 1984
DISC DEV #: 3         ACCESSED: WED, 21 MAR 1984
DISC TYPE: 3          LABEL ADR: **
DISC SUBTYPE: 0       SEC OFFSET: %2
CLASS: DISC           FLAGS: NO ACCESSORS
FCB VECTOR: %0       %0

>LISTF FILE2

**********
FILE: FILE2.MAC.TECHPUBS

FCODE: 0              FOPTIONS: STD,ASCII,FIXED
BLK FACTOR: 29        CREATOR: **
REC SIZE: 88(B)       LOCKWORD: **
BLK SIZE: 1276(W)     SECURITY--READ:    ANY
EXT SIZE: 20(S)               WRITE:   ANY
# REC: 849                    APPEND:  ANY
# SEC: 310                    LOCK:    ANY
# EXT: 16                     EXECUTE: ANY
MAX REC: 849                **SECURITY IS ON       ** FILE2 is secure. **
MAX EXT: 16           COLD LOAD ID: %21605
# LABELS: 0           CREATED: FRI, 23 MAR 1984
MAX LABELS: 0         MODIFIED: FRI, 23 MAR 1984
DISC DEV #: 3         ACCESSED: MON, 26 MAR 1984
DISC TYPE: 3          LABEL ADR: **
DISC SUBTYPE: 0       SEC OFFSET: %12
CLASS: DISC           FLAGS: NO ACCESSORS
FCB VECTOR: %0       %0
```

```
>E
END OF PROGRAM
```

EXAMPLES

To release all security provisions for the file named FILE1, in your logon group group and account MAC.TECHPUBS, enter:

```
:RELEASE FILE1
```

If MPE fails to locate the file, the following error message is displayed:

```
UNABLE TO ACCESS FILE1.MAC.TECHPUBS.  (CIERR 356)
```

:RELLOG

Removes a user logging identifier from the system.

SYNTAX

```
:RELLOG logid
```

PARAMETERS

logid The logging identifier to be removed from the system.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

The :RELLOG command removes a user logging identifier from the system by deleting it from the directory of logging identifiers. This command may be issued only by the user who created the logging identifier. To use :RELLOG, you must have User Logging (LG) capability.

After :RELLOG is issued, programs containing the removed logging identifier will not be allowed to access the logging system.

EXAMPLE

To remove the logging identifier DATALOG from the system, enter:

```
:RELLOG DATALOG
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)

:RENAME

Changes identity (filename, lockword, and/or group name) of disc file.

SYNTAX

```
:RENAME oldfilereference,newfilereference[,TEMP]
```

PARAMETERS

oldfilereference Current name of file, written in the following format:

filename[/*lockword*][.*groupname*[.*acctname*]]

If *acctname* is specified, it must be that of your logon account. You must be the creator of the file to rename it.

newfilereference New name of file, in the same format as *oldfilereference*. If *acctname* is specified, it must be that of your logon account. If *groupname* is specified, it must be one to which you have SAVE access. If *acctname* and/or *groupname* are omitted, the logon account and/or group are assumed.

TEMP Indicates that the old file was, and the new file will be, temporary files. The default is that a permanent file is assumed.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

The :RENAME command changes the system file identification for a permanent or temporary disc file. It can be used to change the name of a file, to move a file from one group to another (by specifying a different *groupname* in the *newfilereference* parameter), or to change the lockword. You cannot rename files that exist in other accounts. (To transfer a file from one account to another, use :FCOPY.)

To rename a file, its group's volume set must be physically mounted or the command will fail. Also, you cannot rename a file across volume sets.

To rename a file, you must be the file's creator and have exclusive (which implies LOCK) access to the file.

EXAMPLES

Since temporary files exist only for the duration of your current job or session, their fully qualified filenames correspond to your logon group and account. To change the name of a temporary file from OLDFILE to NEWFILE, and reassign it to the group NEWG, enter:

```
:RENAME OLDFILE,NEWFILE.NEWG,TEMP
```

Temporary files are valid only in your logon group, and are stored according to their fully qualified filenames.

To change the lockword of the permanent file FILE2 from LOCKA to LOCKB, enter:

```
:RENAME FILE2/LOCKA,FILE2/LOCKB
```

To transfer a file from one group to another within the same account, use the :RENAME command, simply naming the new group in the second parameter. (You must be the creator of the file to use this command.) To move the file MYFILE from GROUP1 to GROUP2, for example, you would enter:

```
:RENAME MYFILE.GROUP1,MYFILE.GROUP2
```

To use :RENAME in this way, you must have SAVE access to the group named in the second parameter (GROUP2 in the previous example). In addition, both groups must be in the system domain or must both reside on the same volume set.

ADDITIONAL DISCUSSION

MPE File System Reference Manual (30000-90236)

:REPLY/=REPLY

Replies to pending resource requests at the Console.

SYNTAX

```
:REPLY pin,reply
```

PARAMETERS

pin The Process Identification Number of the message sender. As part of the message requesting the REPLY, the *pin* always appears after the second slash mark ("/"). In the following example, the *pin* is 43 .

```
?16:15/#S25/43/LDEV# FOR "T" ON TAPE (NUM)?
```

reply The reply type specified in parentheses in the message, defined by one of the following:

(NUM)	Reply must be a logical device number.
(Y/N)	Reply must be YES (or Y), or NO (or N).
(MAX CHARS.=*nn*)	Reply must be a string expression consisting of *nn* characters or less.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be issued only from the Console.*

* :REPLY may be distributed to specific users with the :ALLOW or :ASSOCIATE commands.

OPERATION

User programs that have requested the use of a device and are waiting for your :REPLY/=REPLY remain suspended indefinitely, and cannot be aborted until the :REPLY is issued. If for any reason you cannot reply as requested, such as the particular device is nonexistent or a special form is unavailable, then :REPLY/=REPLY with "0" if type NUM is requested, and with "N" if type Y/N is requested. This returns an error code to the program and the :REPLY/=REPLY is aborted.

The reply will usually take the form "(NUM)" or "(Y/N)", since "(MAX CHARS.=nn)" is used only for labeled tapes.

If your reply is not of the type specified, if the *pin* is incorrect, or if the device you allocate is already allocated to someone else, the system displays the following error message:

```
NO REPLY OUTSTANDING FOR PROCESS
```

EXAMPLES

To reply to a message from the MPE system, enter:

```
10:05/#J19/15/LDEV# FOR "NAS" OF TAPE1600 (NUM)?
:REPLY 15,7
```

or

```
=REPLY 15,7
```

To reply to a FORMS message from the MPE system, enter:

```
15:46/#S93/22/FORMS: PLEASE MOUNT MAILING LABEL FORMS
?15:46/#S39/22/SP#12/LDEV# FOR #S93;OUTFILE ON LP (NUM)?

:REPLY 22,12
15:46/#S39/22/LDEV#12 FORMS ALIGNED OK (Y/N)?    ** Answering NO (N) to this    **
                                                 ** question causes the printing **
                                                 ** to be deferred to a much     **
                                                 ** lower priority. After the    **
                                                 ** forms have been properly     **
                                                 ** aligned, use the :ALTSPOOL-  **
                                                 ** FILE to change the           **
                                                 ** spooling priority, in order  **
                                                 ** to send the spoolfile to the **
                                                 ** printer.                     **

:REPLY 22,NO
?15:48/#S93/22/LDEV#12 FORMS ALIGNED OK (Y/N)?   ** Answering YES (Y) to this   **
                                                 ** question causes the spoolfile **
                                                 ** to go to the printer         **
                                                 ** in its assigned sequence.    **

:REPLY 22,Y
```

When the next spoolfile becomes ACTIVE, you will be requested to mount the appropriate special or standard forms.

To reply to a standard forms request, enter:

```
16:00/#S93/22/STANDARD FORMS
?16:00/#S93/22/LDEV # FOR #S95;L ON LP (NUM)?

:REPLY 22,12
```

Displays accounting information for the logon account and group.

SYNTAX

```
:REPORT [groupset][,listfile][;VS=volset]
```

PARAMETERS

groupset Specifies the accounts and groups for which information is to be listed. The permissible entries, some of which use "wildcard" characters, and their capability requirements (Account Manager, AM, and/or System Manager, SM), are listed below.

group	Reports on the specified group in the logon account. This is the default for standard users, who may specify only their logon group.
@	Reports on all groups in the logon account. This is the default for Account Managers, but may be executed by users with AM or SM capability.
group.acct	Reports on the specified group in the specified account. This requires SM capability.
@.acct	Reports on all groups in the specified account. This requires AM capability (if it is the logon account) or SM capability for any account.
@.@	Reports on all groups in all accounts. This is the default for System Managers and requires SM capability.
group.@	Reports on specified group in any account. This requires SM capability.

The characters @, #, and ? can be used as wildcard characters, but will count toward the eight-character limit. These wildcard characters have the following meanings.

@	Specifies zero or more alphanumeric characters and denotes all members of the set.
#	Specifies one numeric character.
?	Specifies one alphanumeric character.

The characters can be used as follows.

n@	Report on all groups starting with the character "n".
@n	Report on all groups ending with the character "n".
n@x	Report on all groups starting with the character "n" and ending with the character "x".
n##...#	Report on all groups starting with the character "n" followed by up to seven digits.
?n@	Report on all groups whose second character is "n".
n?	Report on all two-character groups starting with the character "n".
?n	Report on all two-character groups ending with the character "n".

These characters, when placed appropriately in the *groupset* parameter, may also be used to report on accounts.

listfile Actual file designator of the output file to which information is to be written. The default is $STDLIST, but output may be redirected with a :FILE equation as follows:

```
:FILE LIST1,DEV=LP
:REPORT, *LIST1
```

volset Informs MPE to report accounting information for the specified volume set.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is Breakable (aborts execution). The user must have Account Manager (AM) or System Manager (SM) capability to execute this command.

OPERATION

The :REPORT command displays the total resource usage logged against accounts and groups, and the limits on those resources. For the standard user, data will be displayed for his or her own group only; an Account Manager may specify all groups in his or her account; the System Manager may specify any or all groups in any or all accounts.

The information includes usage counts and limits for permanent file space (in sectors), CPU time (in seconds), and session connect time (in minutes). The file space usage count reflects the number of sectors used at the time the :REPORT command is issued. However, CPU time and connect time usage appear as they were immediately before the beginning of the current job.

EXAMPLE

To obtain accounting information for your group, enter the :REPORT command. Accounting information will be displayed similarly to the REPORT of the SOPRM account shown below:

```
:REPORT @.SOPRM

ACCOUNT          FILESPACE-SECTORS      CPU-SECONDS      CONNECT-MINUTES
   /GROUP          COUNT     LIMIT      COUNT     LIMIT      COUNT     LIMIT
SOPRM              99004        **      99057        **      88407        **
   /GLOSSARY        1068        **        542        **        656        **
   /PUB              182        **        123        **       1155        **
   /SECT1            180        **         85        **        429        **
   /SECT10         11779        **      25271        **       9716        **
   /SECT2            390        **       4123        **       5302        **
   /SECT3          10675        **       8176        **      13116        **
   /SECT4           2372        **        225        **        294        **
   /SECT5          46579        **      27218        **      25744        **
   /SECT6           6008        **       9324        **       6638        **
   /SECT7           4748        **       8303        **      13263        **
   /SECT8           1957        **       6348        **       3997        **
   /SECT9           3195        **       4570        **       4213        **
```

:RESET

Cancels file equations.

SYNTAX

```
RESET {formaldesignator}
      {@               }
```

PARAMETERS

formaldesignator — A formal file designator name in the form *file*[.*group*[.*account*]][:*nodespec*], for which a :FILE command has previously been issued. The *nodespec* portion may be an environment identifier indicating the location of the file, or it may be $BACK. Specifying $BACK means that the file resides one "hop" back toward your local system (which may be the local system itself).

@ — Signifies all formal file designators specified in all :FILE commands previously issued in this session or job.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

The :RESET command resets a formal file designator to its original meaning, cancelling any :FILE command that has been issued for this formal file designator earlier in the current session or job.

NOTE

The *nodespec* parameter is not part of the HP 3000 Fundamental Operating System. The NS/3000 AdvanceNet subsystem must be purchased separately. The *nodespec* parameter is optional: if you do not have NS/3000 AdvanceNet, omitting the *nodespec* parameter will make no difference in the performance of the :RESET command.

However, specifying *nodespec* on a system that does not have NS/3000 will produce an error.

EXAMPLE

To cancel the effects of a previous :FILE command that specified characteristics for a file programmatically referred to as ALPHA enter:

```
:RESET ALPHA
```

ADDITIONAL DISCUSSION

Refer to the :FILE command in this section.

:RESETACCT

Zeroes the running counts of CPU time or connect time accumulated by an account and by all groups within that account.

SYNTAX

```
:RESETACCT [[ @    ] [[CPU    ]]]
           [[acct  ]  [[CONNECT]]]
```

PARAMETERS

@	Specifies that the counters for all accounts, and all groups within the accounts, are to be reset. Default.
acct	Specifies the name of a particular account, and all groups within the account, for which the CPU and connect time are to be reset. Default is @.
CPU	Specifies that only the CPU usage counter is to be reset. Default is that both the CPU and connect time counters are reset.
CONNECT	Specifies that only the connect time usage counter is to be reset. Default is that both the CPU and connect time counters are reset.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. A user must have System Manager (SM) capability to execute this command.

OPERATION

This command zeroes the running counts of CPU or connect time accumulated by an account and by all groups within that account. If all parameters are omitted when you execute :RESETACCT, all counters (except file space) for all groups in all accounts are reset.

EXAMPLE

To reset the CPU counter for all accounts in the system, enter:

```
:RESETACCT @,CPU
```

:RESETDUMP

Disables the Stackdump facility.

SYNTAX

```
:RESETDUMP
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable.

OPERATION

The :RESETDUMP command disables the MPE Stackdump facility (enabled by :SETDUMP command). Refer to the MPE Debug/Stack Dump Reference Manual (30000-90012). When entered in BREAK, :RESETDUMP does not modify the state of processes already created. If the Stackdump facility is not enabled, the :RESETDUMP command has no effect.

EXAMPLE

To disable the Stackdump facility, enter:

```
:RESETDUMP
```

ADDITIONAL DISCUSSION

MPE Debug/Stack Dump Reference Manual (30000-90012)

:RESTORE

Returns files that have been stored on magnetic tape or serial disc to system. Several parameters are supported only on the G.00.00 and later releases of MPE V. These parameters are noted by "(G.00.00) and later".

SYNTAX

```
:RESTORE [restorefile][;{fileset[-fileset][,fileset[-fileset]][,...]}]
                        {!indirectfile                              }

   [;DEV=device]

   [;SHOW=[showparm[,showparm[,...]]]

   [;FILES=maxfiles]

      {LOCAL                }
   [;{{GROUP=groupname      }[;...]}]
      {ACC[oun]T=acctname   }

              {GROUP  }
   [;CREATE[={ACCT   }[,...]]
              {CREATOR}

   [;CREATOR=[username]]

   [;{KEEP  }]
   [;{NOKEEP}]

   [;{OLDDATE}]
   [;{NEWDATE}]

   [;ONERR[or]={QUIT}]
               {SKIP}
```

PARAMETERS

restorefile

The name of magnetic tape or serial disc file which contains the files you want to restore to the system. If the device has been previously referenced in a file equation (with the :FILE command), it may be back-referenced by using an asterisk (*). This parameter is required for the E/F.00.00 release of MPE V.

restorefile
(G.00.00) and later

If you are using the G.00.00 release or a version of MPE V and you chose not to specify *restorefile*, then a default tape name will be used. The default tape name is equal to the user's logon identification and MPE will designate "TAPE " as the device.

fileset

Specifies the set or sets of files to be restored. This parameter has the form:

filename[.*groupname*[.*acctname*]]

The characters @, #, and ? can be used as wildcard characters in the *fileset* parameter. The wildcard characters count toward the eight character limit for the user, group, account, and file names. These wildcard characters have the following meanings:

@	Specifies zero or more alphanumeric characters. When used by itself, @ denotes "all members of the set".
#	Specifies one numeric character.
?	Specifies one alphanumeric character.

The characters can be used as follows:

n@	Restores all items starting with the character "n".
@n	Restores all items ending with the character "n".
n@x	Restores all items starting with the character "n" and ending with the character "x".
n#######	Restores all items starting with the character "n" followed by seven digits.
?n@	Restores all items whose second character is "n".
n?	Restores all two-character items starting with the character "n".
?n	Restores all two-character items ending with the character "n".

If you specify nothing for the group or account, MPE assumes the logon group and account. Default is @ (restores all files in logon group).

More than one *fileset* (and an associated *-fileset*) may be used with a single :RESTORE command, the number that may be specified is limited as follows: all files in up to 10 different acccounts; all files belonging to 15 different groups within the same account name; 20 different files within the same group and account name.

-fileset
(G.00.00) and later

Name of a file set or subfile set to be excluded from the restore. You cannot nest the *-fileset* parameter. There can only be one *-fileset* for each fileset. A negative sign (-) must precede this parameter.

!indirectfile
(G.00.00) and later

An ASCII file which contains the desired parameters of the :RESTORE command. Each line of the *indirectfile* may contain file sets and keywords. This feature is useful in cases where it is necessary to restore the same files regularly, or when the Command Interpreter 268-character limitation is a problem. An exclamation point (!) must precede this parameter.

device

Specifies the device on which the file is to reside, entered in either the *devclass or ldev* forms. The device class (*devclass*) specifies the type of device. If *devclass* is specified, the file is allocated to any of the volume set's volumes within that device class. The logical device number (*ldev*) specifies a specific device. If *ldev* is specified, the file will be allocated to that device only if one of the volumes in the volume set currently occupies the device.

Default is that MPE attempts to restore the files on the same device class as specified in the file's label. If this fails, a second attempt is made to restore the file on a logical device compatible with the type/subtype specified in the file's label. Should this fail, a third attempt is made to restore the device class "DISC". If this fails, the file is not restored.

SHOW

Request to list names of restored files. Default is a listing of the total number of all files restored and not restored. For files not restored, the reason and the names are listed. This listing is sent to $STDLIST (formal designator $SYSLIST) unless a :FILE command is entered to send the listing to some other device.

showparm
(G.00.00) and later

May be SHORT, LONG, DATES, SECURITY, or OFFLINE. You cannot specify both LONG and SHORT, but you may use more than one *showparm* if they are separated by commas. If neither SHORT nor LONG are selected, then the default is SHORT if $STDLIST is less than 99 characters, and LONG otherwise.

SHORT

For each file restored using the :RESTORE command, the following is printed on the user's terminal (default): file name, group name, account name, logical device number, reel numbers, file size (in sectors) and file code.

LONG

Displays the same information as SHORT, plus: record size, file type, EOF , file record limit, blocking factor, extents allocated and maximum extents of all files restored.

DATES

Causes SHOW to display the creation date, last access date, and last modification date of all files restored.

SECURITY

Causes SHOW to display the creator and file access matrix of all files restored.

OFFLINE

Causes SHOW output to be sent to the system line printer in addition to being displayed on the user's terminal. The output can be redirected to another output device by using a :FILE command. (Formal designator is OFFLINE.)

maxfiles

Maximum number of files that may be restored. If you are restoring from a large tape or disc, such as a SYSDUMP tape, you must include this parameter or the file(s) will not be restored. The value of *maxfiles* should be a least as large as the maximum number of files on the device from which you are restoring. Default is 4,000.

LOCAL, *groupname*, or *acctname* (G.00.00) and later

Restores files from different systems or directory structures. LOCAL restores all files to the user's creator identification, group, and account. To restore a file with the LOCAL option, the user must have READ access to the file on the tape. This is determined by checking the user's capabilities against the account, group and file securities of the tape file. Since the account and group may or may not exist in the directory, using :RESTORE with the LOCAL option requires SM capability to RESTORE from a nonwxistent account or across an existing account. AM or SM capability is necessary to RESTORE from a nonexistent group of the logon account.

The *groupname* specifies that all file restored will have their group name changed to the *groupname* provided. The *acctname* specifies that all files restored will have their accounts changed to the *acctname* provided.

CREATE=

GROUP, ACCOUNT, or CREATOR (G.00.00) and later

Builds groups, accounts, or creators which do not currently exist in the directory. If an entry is created, a message is sent to $SYSLIST (default $STDLIST). GROUP creates the group specified in the file label and requires System Manager (SM) or Account Manager (AM) capability. ACCOUNT creates the account specified in the file label and requires System Manager (SM) capability. ACCOUNT can be shortened to ACCT. CREATOR creates the creator specified in the file label and requires Account Manager (AM) capability. Default is that :RESTORE will create directory entries based on the user's capabilities.

CREATOR= (G.00.00) and later

If the *username* parameter is not specified, the creator identification from the file label will be used. Default: If the creator of the file is not found in the system directory, the file will not be restored, unless the CREATOR or the CREATE option is specified. You will get an error message: NOT RESTORED: CREATOR NOT IN DIRECTORY. In order to resotre this "orphan" file, you must use the CREATOR option or the CREATE option.

If you attempt to restore the file FILEA.GROUPA.ACCOUNTA and the creator, USERA, does not exist on the system, the command RESTORE *TAPEFILE; FILEA.GROUPA.ACCOUNTA will fail. In this case, you will receive the following message:

```
FILEA.GROUPA.ACCOUNTA NOT RESTORED: CREATOR NOT IN
DIRECTORY.
```

Refer to the "EXAMPLE" for this command.

username (G.00.00) and later

All files will have their creator identifications changed to the *username*. If the *username* does not exist, then the file is not restored.

If "CREATOR=*username*" is not specified, the default is to determine the creator in the following order:

1. The creator identification from the file label as it appears on the tape if that creator exists.

2. The user identification of the session performing the :RESTORE if the file is in the same account that the user is logged on to.

3. The string "RESTORE ". A blank must follow RESTORE.

KEEP (E/F.00.00), (G.00.00) and later, or NOKEEP (G.00.00) and later	In cases where a tape file name duplicates an existing disc file name, you may choose whether or not the file to be restored from tape should replace the disc file. The ;KEEP option directs MPE to keep the disc file, and not restore the tape file. NOKEEP purges the file from disc and restores the file from tape unless it cannot be restored, in which case :RESTORE recovers the disc file. Default is NOKEEP on G.00.00 and later releases of MPE V, but is not available on the E.00.00 and F.00.00 releases of MPE V.
NEWDATE(G.00.00) and later, or OLDDATE (E/F.00.00), (G.00.00) and later	MPE stores four dates in each file's file label: the creation date, modification date, last access date, and the allocation date. The allocation time is also stored in the label. NEWDATE, available on the G.00.00 release of MPE V, changes all dates and times to the date and time that :RESTORE was executed. OLDDATE retains the original modification and last access dates, but changes the creation and allocation dates to the date of the :RESTORE procedure.
QUIT or SKIP (G.00.00) and later	Directs :RESTORE to quit or continue when a tape error occurs. SKIP causes :RESTORE to do a file-skip-forward past the tape error, resynchronize, and begin reading again from the tape. One or more files may be lost during resychronization. QUIT causes :RESTORE to abort upon sensing a tape error. ONERROR can be shortened to ONERR. Default is SKIP for unlabeled tapes, and cannot be specified for labeled tapes. Default is QUIT for labeled tapes.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is Breakable (aborts execution). The user must have System Manager (SM), System Supervisor (OP), or Priviledged Mode (PM) capability to execute this command for priviledged files.

* Note that if there is a REPLY pending, :RESTORE will enter a special BREAK mode in which only :REPLY, :RECALL, and :RESUME are allowed.

OPERATION

This command restores data into the system, or on disc, from a file or files previously stored offline by :STORE or :SYSDUMP commands. A message is shown on the System Console requesting the System Operator to mount the tape or serial disc device identified by the *restorefile* parameter and to allocate the device.

The output generated by :RESTORE is sent to a file whose formal designator is $SYSLIST. If a disc or directory error is encountered while updating the directory, updating the Disc Free Space tables, or writing the data to the disc file, the error is reported to $SYSLIST (defaults to $STDLIST) and :RESTORE will continue. Any file belonging to a group whose home volume set has not been mounted will not be restored.

The RESTORE (G.00.00 or later) command resides in a program file (STORE.PUB.SYS), rather than a set of procedures with no global memory. The program file is invoked by both the :STORE and RESTORE commands. It is advisable to use :ALLOCATE STORE.PUB.SYS to speed up the loading process and reduce the use of system resources. If you are restoring from a large tape or disc, such as a SYSDUMP tape, you must include the *maxfiles* parameter or the file(s) will not be restored. The value of *maxfiles* should be a least as large as the maximum number of files on the device from which you are restoring.

Before entering :RESTORE, you may identify *restorefile* as a magnetic tape or serial disc file with a :FILE command. If the user does not specify a *restorefile* (G.00.00 or later), then a default tape name will be used. The default tape name is equal to the user's logon identification and the device is "TAPE". For example, if the user is logged onto TOM.MGR, then the tape request will be for "TOM". No file equation may be used to change this default. Your capabilities determine which files you may restore. If you have System Manager or System Supervisor capability, you can restore any file from a store tape or serial disc, assuming the account and group to which the file belongs, and the user who created the file, are defined in the system. If you have Account Manager capability, you can restore any file in your account. To restore files with negative file codes, you need Privileged Mode capability. If you have standard user capability, you can only restore files in your logon account.

The G.00.00 or later releases of MPE allows the user to build groups, accounts, and creators which do not currently exist in the directory. This way, you may restore files to your system without first defining the account, group, and user with the :NEWACCT, :NEWGROUP, and :NEWUSER commands.

The System Manager and System Supervisor may restore lockword-protected files without specifying the lockword only when :RESTORE is executed during a session. Users without SM or OP capability must always supply the lockword. If the :RESTORE procedure is executed as a job, however, all users, regardless of capability, must supply file lockwords. The :RESTORE command can be invoked via the :RUN STORE.PUB.SYS command. You may use the "INFO=" parameter to specify the restore option, file sets, and keywords; in this case, the :RESTORE command will execute with no further interaction. If the "INFO=" parameter is not specified, the "<--" prompt will appear. Acceptable responses are: a complete :STORE command, a complete :RESTORE command, an MPE command preceded by a colon (:), or EXIT.

When the :RUN command is used, the system defined JCW "CIERROR" will not be changed even if the :RESTORE aborts. When invoking :RESTORE via the Command Interpreter (e.g. :RESTORE *T;@) the system defined JCW "CIERROR" will be set to the value 1091 if the :RESTORE command aborts for any reason. Check this value after :RESTORE aborts with the :SHOWJCW command.

The output generated by :RESTORE is sent to a file whose formal file designator is $SYSLIST. If a disc or directory error is encountered while updating the directory, updating the Disc Free Space Map, or while writing data to the disc file, the error is reported to $SYSLIST, which defaults to $STDLIST, and :RESTORE will continue.

NOTE

The G.01.00 and later releases of :RESTORE determine whether sufficient disc space remains to restore a file that already exists on the disc. If sufficient space reamins, :RESTORE writes a new copy of the file to the disc before purging the old copy of the file. The old copy of the file is purged only if the :RESTORE operation si successful. (This is the procedure established by the G.00.00 version of :RESTORE).

If sufficient space is not available, :RESTORE first purges the old copy of the file and then writes a new copy to the disc. If the :RESTORE operation fails in this circumstance, you will receive a message on $STDLIST informing you that there is no copy of the file on the disc:

"***WARNING: OLD FILE HAS BEEN PURGED***"

EXAMPLES

To restore all files belonging to your logon group from the *restorefile* named "T", enter:

```
:FILE T;DEV=TAPE
:RESTORE *T;@;KEEP;SHOW
```

In response, the System Operator receives a request to mount the tape identified as "T". If a disc file on T already exists in the system, it will not be restored because the KEEP parameter was specified.

To restore a file ABC without specifying a *restorefile* (G.00.00 or later), no file equation need be used. For example:

```
:RESTORE;ABC.PUB.SYS;SHOW                    **Restores ABC.PUB.SYS**
STORE/RESTORE, VERSION 2   (C) 1981  HEWLETT-PACKARD CO.
FRI, APR 12, 1985, 11:56 AM

WILL RESTORE       1 FILES;  NUMBER OF FILES ON TAPE =   87

FILENAME.GROUP     .ACCOUNT  LDN  ADDRESS REEL  SECTORS CODE

ABC      .PUB      .SYS        2%00100217    2        4

FILES RESTORED:          1
FILE NOT RESTORED:       0
:
```

To restore all files without specifying a *fileset* (G.00.00 or later), a store or restore (S/R) warning will appear alerting you that all files, based on your capabilities, will be restored:

```
:RESTORE                                     **Restores ABC.PUB.SYS**
STORE/RESTORE, VERSION 2   (C) 1981  HEWLETT-PACKARD CO.
FRI, APR 13, 1985, 11:39 AM

RESTORE
      ^
****WARNING:  AN 'EMPTY' FILESET WAS FOUND...BASED ON YOUR
CAPABILITIES, '@.@.@' WILL BE ASSUMED**** (S/R 6229)
```

To have the list of restored files printed on a line printer, enter:

```
:FILE SYSLIST;DEV=LP
:FILE T;DEV=TAPE
:RESTORE *T;@;SHOW
```

To restore the file FILEA.GROUPA.ACCOUNTA when the creator, USERA, does not exist on the system, you may use one of the methods shown here:

- RESTORE *TAPEFILE; FILEA.GROUPA.ACCOUNTA; CREATOR=USERB
  (Changes the creator of FILEA to USERB. USERB must exist on the system.)

- RESTORE *TAPEFILE; FILEA.GROUPA.ACCOUNTA; CREATE=CREATOR
  (Creates USERA on the system.)

- RESTORE *TAPEFILE; FILEA.GROUPA.ACCOUNTA; CREATE
  (Creates USERA on the system.)

To restore only a subset of the fileset (G.00.00 or later), enter:

```
:RESTORE *T;@.@.@ -@.PUB.SYS      ** Restore all files except those in the PUB group of the
                                     SYS account. **
```

To use restore with the :RUN command, enter:

```
:RUN STORE.PUB.SYS; INFO="RESTORE *T;@.@.@;SHOW"
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)

:RESUME

Resumes execution of a suspended operation.

SYNTAX

```
:RESUME
```

PARAMETERS

None.

USE

This command may be issued only while in BREAK. It may not be used from a Session (other than while in BREAK mode), Job, or Program. It is not Breakable.

OPERATION

After a program or MPE command operation is suspended by pressing BREAK or by using the -CAUSEBREAK intrinsic, the :RESUME command resumes execution of the operation at the point where the execution was suspended. Note that the :RESUME command is legitimate only during a BREAK. Many MPE commands are aborted rather than suspended by a BREAK, and thus cannot be resumed. (Refer to Table 1-1 for a list of nonbreakable commands.)

If, instead of :RESUME, you enter another program command (such as :EDITOR, :FORTRAN or :RUN) or one of the nonprogram commands (:HELLO, :BYE, :JOB, or :DATA), the Command Interpreter prints the following message on your terminal: "ABORT? (YES/NO)" If you respond "YES" to the "ABORT?" message, the Command Interpreter aborts the current program and executes the command.

If you respond "NO" to the "ABORT?" message, the Command Interpreter prints the message "COMMAND NOT ALLOWED IN BREAK" and prompts you for another command. If you now enter RESUME at a colon (:) prompt, the suspended program continues at the point where it was interrupted. If you had logged on using :( ) COMMAND LOGON with a program command inside the parentheses, then responding "YES" to the "ABORT?" message causes MPE to abort the command and to log you off immediately.

EXAMPLE

To continue a suspended program at the point of interruption, enter:

```
:RESUME
 READ PENDING
   RETURN
```

:RESUMEJOB

Resumes a suspended job.

SYNTAX

```
:RESUMEJOB #Jnnn
```

PARAMETERS

#Jnnn	A job number.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be executed only from the Console.*

* Unless distributed with the :ALLOW command, or if the :JOBSECURITY is set LOW.

OPERATION

The System Operator uses the :RESUMEJOB command to direct MPE to resume processing a job suspended with the :BREAKJOB command. The job will continue execution from the point at which it was suspended; no message is issued.

EXAMPLE

To resume the processing of job 68 , enter:

```
:RESUMEJOB #J68
```

:RESUMELOG

Resumes system logging following suspension caused by an error.

SYNTAX

```
:RESUMELOG
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It may be executed only from the Console,* or by a user with System Supervisor (OP)

* Unless distributed to users with the :ALLOW command.

OPERATION

When the Operator directs MPE to resume logging with :RESUMELOG, a special Log Record is displayed that denotes the number of log events and corresponding records that were not recorded while logging was suspended; the total number of unrecorded job initiation records; and the total number of unrecorded job/session termination records.

EXAMPLES

Assume the system is online and running with logging enabled. If a recoverable error occurs, the following error message is sent to the System Console:

```
ST/10:43/LOG FILE NUMBER 104 ERROR #46.  LOGGING SUSPENDED
```

After the error is corrected, enter the :RESUMELOG command. A confirmation message will then appear at the System Console, as follows:

```
ST/10:45/LOGFILE NUMBER 104. LOGGING RESUMED.
ST/10:45/LOG FILE NUMBER 104 ON.
```

ADDITIONAL DISCUSSION

Refer to the discussion of system logging in "MPE LOGGING FACILITIES" in the MPE V System Operation and Resource Reference Manual (32033-90005).

:RESUMESPOOL

Resumes suspended spooler output to a spooled device.

SYNTAX

```
:RESUMESPOOL ldev;BACK [nnn FILES]
                       [nnn PAGES]
```

```
:RESUMESPOOL ldev;FORWARD [nnn FILES]
                          [nnn PAGES]
```

```
:RESUMESPOOL ldev;BEGINNING
```

PARAMETERS

ldev	The logical device number of a spooled device.
BACK	Instructs the spooler to backspace *nnn* files or *nnn* pages and resume printing at that point. (Refer to "OPERATION".)
FORWARD	Instructs the spooler to space forward *nnn* files or *nnn* pages and resume printing at that point. (Refer to "OPERATION".)
BEGINNING	Instructs the spooler to resume printing at the beginning of the file which had been previously suspended.
nnn	The number of files or pages you wish the spooler to backspace or space forward when printing resumes. (Must be an integer between 1 and 256, inclusive.)
FILES or PAGES	Informs the spooler process which unit of measure to use when printing resumes. For the purposes of this command, FILE is defined as the text appearing between FOPEN intrinsic statements within the spoolfile. (Refer to "OPERATION".) Note that the use of the FILES parameter is not allowed on the HP 2680A page printer or an HP 2608S CIPER-protocol printer. PAGE is the literal page (usually 60 lines or skip to channel 1), as output by the spooler to the printer.

USE

This command may be issued from a Session, Job, Program, or in BREAK. It is not Breakable. It may be executed only at the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

If you specify only the *ldev* parameter, the printer will resume printing at the beginning of the highest priority spoolfile. Otherwise, the printer will resume printing the previously ACTIVE spoolfile.

Always overestimate the number of files or pages you need when using the BACK parameter or underestimate the number when using the FORWARD parameter. This is the only way to ensure that you get all the output you need, since partial pages and header pages affect the page count. However, if you instruct the spooler to go BACK further than the beginning of the file, MPE will display an error message on the System Console and printing will resume at the beginning of the file. Similarly, an error message is generated if you instruct the spooler to advance FORWARD beyond the point where files exist. In this case, printing will not resume until a new command is issued.

By using the SPOOK utility with mode control on, you can determine where each FOPEN intrinsic occurs within a spoolfile. This is useful, for example, when you are compiling, preparing, and running large programs, and printing the entire output is unnecessary. For information about the SPOOK utility, refer to the MPE V System Utilities Reference Manual (32033-90008).

EXAMPLES

To resume output to logical device number 6 at the beginning of the file, enter:

```
:RESUMESPOOL 6;BEGINNING
```

To resume output to logical device number 6 , and reprint the last two pages, enter:

```
:RESUMESPOOL 6;BACK 2 PAGES
```

To resume output to logical device number 6, and print the highest priority spoolfile, enter:

```
:RESUMESPOOL 6
```

Compiles an RPG program. RPG is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:RPG [textfile][,[uslfile][,[listfile][,[masterfile][,newfile]]]]
```

PARAMETERS

textfile Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is RPGTEXT. Default is $STDIN.

uslfile Actual file designator of the User Subprogram Library (USL) file to which the object program is written. This can be any binary input file with a file code of USL or 1024. Its formal file designator is RPGUSL. If the *uslfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS. If entered, this parameter refers to a file created in one of four ways:

- By saving, with the MPE :SAVE command. Default is the USL file created during a previous compilation.
- By building the USL with the MPE Segmenter-BUILDUSL command. (Refer to the MPE Segmenter Reference Manual (30000-90011).
- By creating a new USL file with the MPE :BUILD command and specifying a file code of USLor already fixed in file 1024.
- By specifying a nonexistent *uslfile* parameter, thereby creating a permanent file of the correct size and type.

listfile Actual file designator of the file on which the program listing is written. This can be any ASCII output file. Formal file designator is RPGLIST. Default is $STDLIST.

masterfile Actual file designator of the master file to be merged against *textfile* to produce a composite source. This can be any ASCII input file. Formal file designator is RPGMAST. Default is that the master file is not read, and input is read from *textfile*, or from $STDIN if *textfile* is not specified. If two files being merged have identical line numbers, the lines from *textfile* or from $STDIN will overwrite those in *masterfile*.

newfile Actual file designator of the file produced by merging *textfile* and the *masterfile*. This can be any ASCII output file. Formal file designator is RPGNEW. Default is that no file is written.

NOTE

The formal file designators used in this command (RPGTEXT, RPGUSL, RPGLIST, RPGMAST, and RPGNEW) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or Job. It may not be issued in Break or from a Program. It is Breakable (suspends execution).

OPERATION

This command compiles an RPG program onto a User Subprogram Library (USL) file on disc. If you do not specify *textfile*, MPE expects input from your standard input device. If you create the USL file before compiling the source code, you must assign it a file code of USL or 1024.

EXAMPLES

The following example compiles an RPG program entered from your standard input device, storing the object code in the default USL file $OLDPASS. The listing is sent to the standard list device.

```
:RPG
```

The next example compiles an RPG program contained in the disc file SOURCE. The object code is stored in the USL file OBJECT, which is a permanent disc file created with the :BUILD command. The program listing is sent to the disc file LISTFL.

```
:BUILD OBJECT;CODE=USL
:RPG SOURCE,OBJECT,LISTFL
```

To compile an RPG program, storing the object code in the USL file OBJECT (created during the compilation process) enter:

```
:RPG SOURCE,OBJECT,LISTFL
```

ADDITIONAL DISCUSSION

RPG/3000 Compiler Reference Manual (32104-90001)
MPE Segmenter Reference Manual (30000-90011)

:RPGGO

Compiles, prepares, and executes an RPG program. RPG is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:RPGGO [textfile][,[listfile][,[masterfile][,newfile]]]
```

PARAMETERS

textfile — Actual file designator of the input file from which source program is read. This can be any ASCII input file. Formal file designator is RPGTEXT. Default is $STDIN.

listfile — Actual file designator of the file on which the program listing is written. This can be any ASCII output file. Formal file designator is RPGLIST. Default is $STDLIST.

masterfile — Actual file designator of a file which is merged against *textfile* to produce a composite source. This can be any ASCII input file. Formal file designator is RPGMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified. If two files being merged have identical line numbers, the lines from *textfile* or from STDIN will overwrite those in *masterfile*.

newfile — Actual file designator for the file produced by merging *textfile* and the *masterfile*. This can be any ASCII output file. Formal file designator is RPGNEW. Default is that no file is written.

NOTE

The formal file designators used in this command (RPGTEXT, RPGLIST, RPGMAST, and RPGNEW) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to "Implicit :FILE Commands For Subsystems" section of the :FILE command.

USE

This command may be issued from a Session or a Job. It may not be issued in BREAK or from a Program. It is Breakable (suspends execution).

OPERATION

This command compiles, prepares, and executes an RPG program. If you do not specify *textfile*, MPE expects you to enter the source code from your standard input device.

The USL file created during compilation is a system defined temporary file $OLDPASS, which is passed directly to the Segmenter. It cannot be accessed, since the Segmenter also uses $OLDPASS to store the prepared program segments and overwrites the USL file of the same name.

EXAMPLES

To compile, prepare, and execute an RPG program entered from your standard input device, sending the program listing to your standard list device, enter:

```
:RPGGO
```

To compile, prepare, and execute an RPG program read from the disc file SOURCE, sending the program listing to the disc file LISTFL, enter:

```
:RPGGO SOURCE,LISTFL
```

ADDITIONAL DISCUSSION

RPG/3000 Compiler Reference Manual (32104-90001)
MPE Segmenter Reference Manual (30000-90011)

:RPGPREP

Compiles and prepares an RPG program. RPG is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

Syntax

```
:RPGPREP [textfile] [,[progfile] [,[listfile] [,masterfile] [,newfile]]]
```

PARAMETERS

textfile Actual file designator of the input file from which the source program is read. This can be any ASCII input file. Formal file designator is RPGTEXT. Default is $STDIN.

progfile Actual file designator of the program to which the prepared program segments are written. You will get the best results by omitting the *progfile* parameter. When you omit *progfile*, the Segmenter creates the program file, which will reside in the temporary file domain as $OLDPASS. If you do create your own program file, however, you must do so in one of two ways:

- By using the MPE BUILD command, and specifying a file code of 1029 or PROG, and a *numextents* value of 1. This file is then used by the :PREP command.
- By specifying a nonexistent file in the *progfile* parameter, in which case a job or session file of the correct size and type is created. Default is that $NEWPASS is assigned.

listfile Actual file designator of the file on which the program listing is written. This can be any ASCII output file. Formal file designator is RPGLIST. Default is $STDLIST.

masterfile Actual file designator of the master file that is mergered against *textfile* to produce a composite sourcefile. This can be any ASCII input file. Formal file designator is RPGMAST. Default is that master file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified. If two files being merged have identical line numbers, the lines from *textfile* or from $STDIN will overwrite those in *masterfile*.

newfile Actual file designator of the file produced by merging *textfile* and the *masterfile*. This can be ay ASCII output file. Formal file designator is RPGNEW. Default is that no file is written.

NOTE

The formal file designators used in the command (RPGTEXT, RPGLIST, PRGMAST, and RPGNEW) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to "Implicit :FILE Commands For Subsystems" discussion of MPE :FILE command.

USE

This command may be issued from a Session or a Job. It may not be issued in BREAK or from a Program. It is Breakable (suspends execution).

OPERATION

This command compiles and prepares an RPG program to a program file on disc. If you do not specify *textfile*, MPE expects the source program to be entered from your standard input device. The USL file $OLDPASS, created during compilation, is a system-defined temporary file passed directly to the Segmenter. You can access it only if you do not use the $NEWPASS default for *progfile*. This is because the Segmenter also uses $OLDPASS to store the prepared program segments, overwriting any existing temporary files of that name.

EXAMPLE

The compile and prepare an RPG program entered from your standard input device, sending the listing to your standard list device, enter:

```
:RPGPREP
```

The USL file created during compilation is a temporary file passed directly to the Segmenter. You can access it under the name $OLDPASS only if the prepared program segments are not also stored in $OLDPASS (which will overwrite the USL file). Therefore, to save the compiled USL and the prepared program file, specify a non-existant file for *progfile* in the :RPGPREP command line and save the USL file $OLDPASS under another name. In the following example, the prepared program is saved as COMFL, and the USL file is renamed (and saved) to NUSL:

```
:RPGPREP,COMFL
:SAVE $OLDPASS,NUSL
```

Unless you have specifically created a permanent file to store the prepared program, the program file COMFL will be stored in the temporary file domain. To save it as a permanent file, use the :SAVE command:

```
:SAVE COMFL
```

Using the :BUILD command, you can create your own program file in the permanent file domain. When you do so, be sure to specify a file code of "PROG" or "1029" and a *numextents* parameter value of 1. Such a file is created in the next example. It is then used by the :PREP command.

```
:BUILD PROGFL;CODE=PROG;DISC=,1
:RPGPREP,PROGFL
```

To send the program listing to a device other than the default standard list device, use the :FILE command. In this example, the file equation assigns the filename LINEA to device class LP (your line printer). LINEA is then backreferenced in the :RPGPREP command line:

```
:FILE LINEA;DEV=LP
:RPGPREP,EDTDISC,COMFL,*LINEA
```

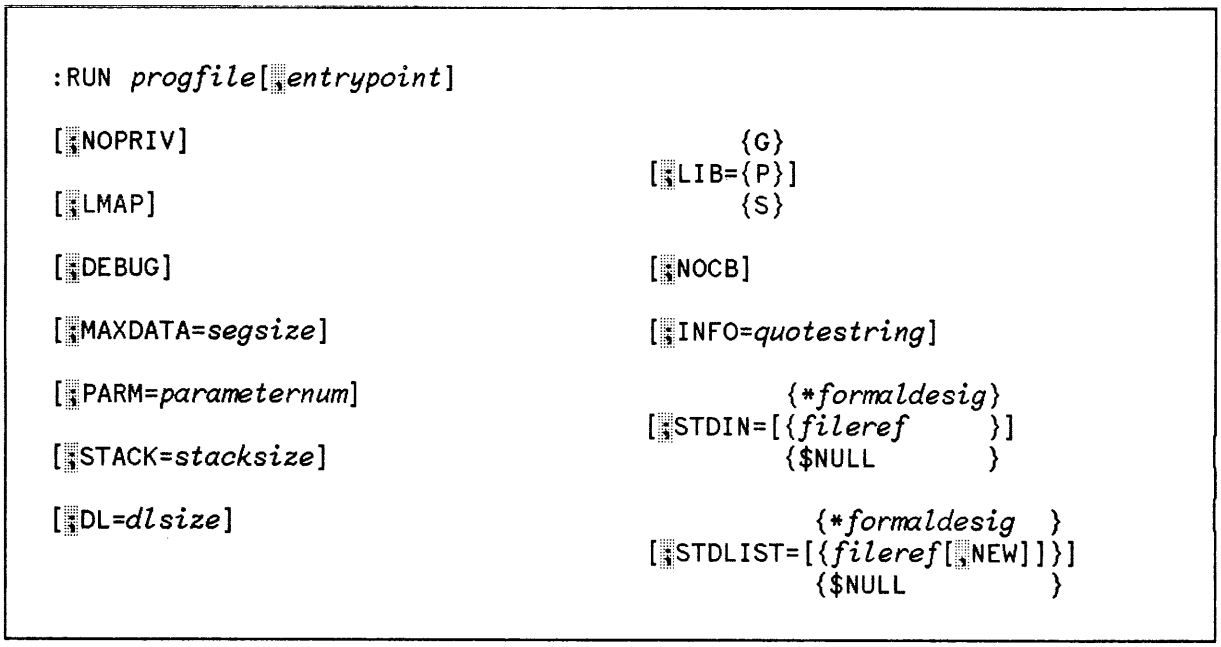
ADDITIONAL DISCUSSION

RPG/3000 Compiler Reference Manual (32104-90001)
MPE Segmenter Reference Manual (30000-90011)

:RUN

Executes a prepared program.

Syntax



PARAMETERS

progfile — Actual file designator of the program file that contains the prepared program.

entrypoint — Program entry point where execution is to begin. It contains a character string, terminated by a blank, specifying the entry point (label) in the program where execution is to begin when the program is executed. This point may be the primary entry point of program, or any secondary entry point in program's outer block. Default is primary entry point.

NOPRIV — Declaration that program segments will be placed in nonprivileged (user) mode. This parameter is for programs prepared with Privileged Mode capability and makes them accessible to nonprivileged users. Normally, programs containing privileged instructions are executed in Privileged Mode only if the program was prepared with Privileged Mode (PM) capability class. A program containing legally compiled privileged code, placed in nonprivileged mode, may abort when an attempt is made to execute it. If NOPRIV is specified in the :RUN command, all program segments are placed in nonprivileged mode. Library segments are not affected because their mode is determined independently. NOPRIV produces the same effect as omitting the $OPTION PRIVILEGED and OPTION PRIVILEGED entries in SPL source input. Default is that Privileged Mode programs will remain in Privileged Mode.

LMAP — Request to produce a descriptive listing of the allocated (loaded) program on the file whose formal file designator is LOADLIST. f there is no reference to LOADLIST in any :FILE command, listing is sent to $STDLIST. Default is no listing.

DEBUG

Request to issue a Debug call before the first executable instruction of the program. Unless you have READ and EXECUTE access to the program, this parameter is ignored. If assigned Privileged Mode (PM) capability, you are put in privileged mode Debug. If you do not have PM capability, you are put in user mode Debug.

segsize

Maximum stack area (Z-DL) size permitted, in words. This parameter is included if it is expected that size of DL-DB or Z-DB areas will be changed during program execution. Regardless what you specify, MPE may change *segsize* to accommodate table overflow conditions.

If you prepare your program with *segsize* less than the configured minimum, the value is rounded up to the minimum or to the amount needed by the program (as calculated by the Segmenter). The maximum actual *segsize* permitted a program is 31,232 words. If your specified *segsize* does exceed the maximum, it will be rounded down to 31,232 words.

parameternum

An integer containing a parameter to be passed to the new program (accessed through Q-4 of the outer block).

stacksize

Size of initial local data area, Z-Q (initial), in the stack. This value, if specified, must be between 511 and 32,767 words. It overrides the default stack size estimated by MPE Segmenter.

dlsize

DL-DB area to be initially assigned to stack. This area is of interest mainly in programmatic applications. To accommodate system logging requirements, the DL-DB area is always rounded upward in such a way that the distance from the beginning of the stack data segment to the DB address is a multiple of 128 words. This value must be within -1 and 32,767 words. The default is estimated by the MPE Segmenter.

G, P, or S

Searches the segmented procedure libraries of the program file's group and account. The G option first searches the group library, then the account library, then the system library. The P option searches the account library, then the system library. The S option searches the system library for external references to segmented procedures. Default is S.

NOCB

Request that file system not use the stack segment, PCBX, for its control blocks, even if sufficient space available. This permits expansion of the stack, by using the DLSIZE and ZSIZE intrinsics, to the maximum possible limit at a later time. However, NOCB causes the File Management System to operate more slowly.

quotedstring

A sequence of ASCII characters bounded by a pair of single quotation marks (apostrophes) or by double quotation marks. If you want a quotation to appear within *quotedstring*, the quotation and its quotation marks must also be bounded by quotation marks. For example, to insert "and" into a *quotedstring*, it must appear as ""and"". Similarly, 'and' must appear as "and". The maximum length of the string, including delimiters, is 255 characters. Refer to "EXAMPLES".

The string will be placed just after the global area of the new program's stack. A DB-relative byte pointer to the string in the new program's stack will be placed at Q-5 of the stack, where Q is the initial value of the Q-register at activation time. The length of the string in bytes will be placed at Q-6. If no string is specified, Q-5 and Q-6 will both contain 0. For more information refer to the CREATEPROCESS intrinsic in the MPE V Intrinsics Reference Manual (32033-90007).

STDIN

Use of this parameter allows the user to specify the file to be used as $STDIN by the program being executed. If this parameter is omitted, or if nothing is specified after the equal sign, as in ";STDIN=", then STDIN defaults to the standard input device for the job or session. This parameter must have one of the following subparameters:

**formaldesig* The formal file designator for a file previously specified in a file equation.

fileref The name of an existing permanent disc file.

$NULL The actual file designator of a system defined file that is always treated as an empty file. When referenced by a program as $STDIN, the program receives only an end-of-file indication when accessed. When referenced by a program as $STDLIST, the associated write request is accepted by MPE, but no physical output is actually performed. Thus, $NULL can be used to discard unneeded output from an executing program.

STDLIST

Use of this parameter allows the user to specify the file to be used as $STDLIST by the program being executed. If this parameter is omitted, or if nothing is specified after the equal sign, as in ";STDLIST=", then STDLIST defaults to the standard list device for the job or session. This parameter has the same subparameters as STDIN, but you may also specify the keyword NEW.

NEW The name to be assigned to a job/session temporary disc file created with the system defaults. The system defaults of the new file are fixed length ASCII 132-byte records with a maximum file size of 1023 records.

USE

This command may be issued from a Session or a Job. It may not be issued in BREAK or from a program. It is Breakable (suspends operation).

OPERATION

This command executes a program prepared in a program file. It permits searching segmented libraries (SLs), to satisfy external references, Relocatable libraries (RLS) are not searched.

If the volume set containing the file to be run is not mounted, this command implicitly causes that volume set to be mounted.

If the file is temporary, only the logon group and account libraries for the current session will be searched. Refer to MPE File System Reference Manual (30000-90236) for more information on file domains.

EXAMPLES

To list the segments of a loaded program, enter:

```
:RUN XLAB;LMAP
```

To run a program stored in the program file PROG4, beginning at the entry point SECLAB, enter:

```
:RUN PROG4,SECLAB
```

The following example runs a program "TESTPROG" with $STDIN set to an old disc file named INPUT and $STDLIST set to the line printer:

```
:FILE LPFILE;DEV=LP
:RUN TESTPROG; MAXDATA=10000; STDIN=INPUT; STDLIST=*LPFILE
```

The next example runs a program using the STDIN parameter, setting $STDIN to an existing disc file named INPUT, this time referenced through a file equation. In this example, $STDLIST is set to a temporary disc file named RESULTS that is automatically created by the :RUN command, enter:

```
:FILE INFILE=INPUT,OLD
:RUN TESTPROG;DEBUG;STDIN=*INFILE;STDLIST=RESULTS,NEW
```

The following example of the :RUN command uses the "INFO=" parameter to pass a string to the program:

```
:RUN MYPROG;MAXDATA=2000;INFO= "A test with ""and""  characters"
```

In *quotedstring*, "and" is bounded by an extra pair of quotation marks. As a result, the string passed to the program is:

```
A test with "and" characters
```

:SAVE

Saves file in permanent system file domain.

SYNTAX

```
:SAVE {$OLDPASS,newfilereference}
      {tempfilereference        }
```

PARAMETERS

$OLDPASS — A system-defined temporary file. After this file is saved, it can no longer be referenced by the name $OLDPASS .

newfilereference — New actual file designator assigned to $OLDPASS when it is made permanent. It format is: the format:

filename[/*lockword*][.*groupname*[.*acctname*]]

If *groupname* is used, it must indicate a group to which you have SAVE access, as defined by your Account Manager. If *groupname* is omitted, the logon group is assigned.

tempfilereference — Actual file designator of the temporary file to be made a permanent file under the same designator. The file is deleted from the job/session temporary file domain and entered into the system file domain. Its format is:

filename[/*lockword*][.*groupname*[.*acctname*]]

If *groupname* is used, it must indicate a group to which you have SAVE access, as defined by your Account Manager. If *groupname* is omitted, the logon group is assigned.

USE

This command may be issued from a Session or a Job, in BREAK, or from a Program. It is not Breakable.

OPERATION

The :SAVE command saves a temporary file by converting it to a permanent file in the system file domain. This command is necessary when the subsystem or program that created your file does not allow you to save it while the program is executing. The volume set for the group in which the file is to be saved must be mounted, because files cannot be saved across volume sets.

You must specify a new filename for $OLDPASS, because MPE does not allow $OLDPASS as a permanent filename. If, when saving $OLDPASS, there is a file in the temporary domain with the same name specified by *newfilereference*, MPE will attempt to save $OLDPASS by creating a new temporary file. This temporary filename, created by :SAVE, will start with "S" and be followed by seven digits: S*dddhhmm*, where *ddd* is the Julian day of the year, *hh* is the hour of the day, and *mm* is the minute. The new temporary file will then be saved under the filename specified by *newfilereference*, and be deleted from the temporary domain. If both temporary and permanent files exist under the same name specified by *newfilereference*, the temporary :SAVE file will be saved as a permanent file. In this case a printed error message will state the filename for the new :SAVE file. It can be renamed later using :RENAME.

This command applies only to temporary files on disc. It is similar to opening a file with the FOPEN intrinsic, and then closing it with the FCLOSE intrinsic, using a permanent file disposition.

EXAMPLES

To save the temporary file $OLDPASS, containing an object program, to the program file PROGFILE, enter:

```
:SAVE $OLDPASS,PROGFILE
```

To save the temporary file TEMPFL as a permanent file with the same name, enter:

```
:SAVE TEMPFL
```

To save the temporary file DATAFILE in the group GROUPX, enter:

```
:SAVE DATAFILE.GROUPX
```

To save a temporary file (other than $OLDPASS) and change its name, use the :SAVE and :RENAME commands. Only the logon group and account directories in the current session will be searched. For example:

```
:SAVE DATAFILE
:RENAME DATAFILE,DATABASE
```

:SECURE

Restores security provisions for a file that were previously suspended by a :RELEASE command.

SYNTAX

```
:SECURE filereference
```

PARAMETERS

filereference — Actual file designator of the disc file whose security provisions are to be restored. Its format is: format:

filename[/*lockword*][.*groupname*[.*acctname*]]

USE

This command may be issued from a Session or Job, in BREAK or from a program. It is Breakable.

OPERATION

The :SECURE command restores all security provisions for a file that were previously suspended by a :RELEASE command in this or another job.

This command can be used only for a permanent disc file whose file label identifies the user as the creator of that file. The :SECURE command will fail if the group's home volume set is not mounted. When the normal (default) MPE security provisions are in effect, the file must belong to the logon account and group.

EXAMPLE

To restore the security provisions previously in effect for the file named FILE1, enter:

```
:SECURE FILE1
```

Calls the MPE Segmenter.

SYNTAX

```
:SEGMENTER [listfile]
```

PARAMETERS

listfile Actual file designator of any ASCII output file that is to receive listed output from the MPE Segmenter. Formal file designator is SEGLIST. Default is $STDLIST. Usually this file is a line printer and must be defined in a :FILE command, backreferenced as follows:

```
:FILE LISTFL;DEV=LP
:SEGMENTER *LISTFL
```

NOTE

The formal file designator used in this command, SEGLIST, cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to "Implicit :FILE Commands For Subsystems" discussion of the MPE :FILE command.

USE

This command may be issued from a Session or a Job. It may not be issued in BREAK or from a Program. It is Breakable (suspends execution).

EXAMPLE

To call the MPE Segmenter from a session and transmit the output to a line printer instead of the standard list device, enter:

```
:FILE LISTFL;DEV=LP
:SEGMENTER *LISTFL
-
```

NOTE

You must have READ and LOCK capability to use a relocatable library with the Segmenter command.

ADDITIONAL DISCUSSION

MPE Segmenter Reference Manual (30000-90011)

Allows a job using a spooled $STDLIST to mark its standard list device for deletion when the job terminates.

SYNTAX

```
:SET STDLIST={DELETE}
             {SAVE  }
```

PARAMETERS

DELETE	Flags the job's $STDLIST for deletion at job termination.
SAVE	Cancels the effect of a previous :SET STDLIST=DELETE command. Default is SAVE.

USE

This command may be issued from a Job or a Program. It may not be issued from a Session or in BREAK. It is not Breakable.

OPERATION

The :SET command can be placed anywhere between the :JOB and :EOJ statements. It is most practical to place it at the end of a job stream since the command will not execute if the job fails. $STDLIST will then print, allowing you to study your listing, and locate the problem. The effect of a :SET STDLIST=DELETE can be reversed by entering :SET STDLIST=SAVE into the job stream. Note that the :SET command works only on jobs with a spooled $STDLIST.

EXAMPLE

```
!JOB EXAMPLE, MAC.TECHPUB,XGROUP
!CONTINUE
!RUN UPDATE.PUB.SYS;PARM=1;MAXDATA=16000
!IF JCW < FATAL THEN
!SET STDLIST=DELETE
!ENDIF
!EOJ
```

:SETCATALOG

Causes the Command Interpreter to search a catalog of user defined commands (UDCs) and establish an entry for each command, or to deactivate previously set catalogs.

SYNTAX

```
:SETCATALOG [catfilename[/lockword][,catfilename[/lockword]]...]...
   [,SHOW]
   [,{SYSTEM }]
   [,{ACCOUNT}]
   [,USER=user[.acct]]
```

PARAMETERS

catfilename	Catalog (UDC) filename. Normally, this file is of a file consisting of one or more User Defined Commands (UDCs). When more than one user defined command resides in the catalog file, the commands must be separated from each other by one line, the first character of which must be an asterisk. If you omit *catfilename*, all UDCs within that file will be deactivated at the user, account, system level to which you have access.
lockword	An optional lockword assigned by creator of the file.
SHOW	Lists catalogs and UDCs as the user defined commands are initialized. This parameter is useful for listing additional information if there is an error in UDC initialization. SHOW lists each UDC as it is checked. If an error occurs, it is listed after the erroneous UDC.
ACCOUNT or SYSTEM	Specifies the catalog file being defined or deactivated. ACCOUNT specifies that the catalog file being defined or deactivated reset is at the account level. ACCOUNT requires Account Manager (AM) capability. SYSTEM specifies that the catalog file being defined or deactivated is at the system level. SYSTEM requires System Manager (SM) capability. Default is the user level catalog.
user[.acct]	The username of the user whose catalog file will be defined or reset. Requires Account Manager (AM) capability.

USE

This command may be issued from a Session, Job or in BREAK. It may not be issued from a Program. It is not Breakable.

OPERATION

The :SETCATALOG command causes the Command Interpreter to search a specially prepared disc file for user defined commands. The Command Interpreter then establishes a directory entry for each command in the file.

A user must have READ and LOCK access to the UDC file in order to use the :SETCATALOG command and to logon while that UDC file is in effect. Therefore, for account and system level UDCs, the account and group file security under which the files reside must allow READ and LOCK access to all users. Lockwords specified using :SETCATALOG are put into COMMAND.PUB.SYS, so the user need not know the lockword to logon.

Entering :SETCATALOG with no parameters removes all your user level UDC files from the system catalog. Note that the files are not purged, but are deleted from the catalog. In order to purge or modify a particular UDC file, you must first remove from the system UDC catalog by issuing the :SETCATALOG command with a parameter list that does not include that particular UDC filename.

EXAMPLE

To create a user defined command (UDC) file named MYCMNDS with the Editor, and have the Command Interpreter establish a directory entry for it, enter:

```
:EDITOR
HP32201A.7.15 EDIT/3000  MON, MAY 14, 1985,  8:01 AM
(C) HEWLETT-PACKARD CO. 1985

/ADD
      1 S                    ** UDC name **
      2 SHOWJOB              ** MPE command **
      3 **                   ** Separates UDCs **
      4 R                    ** UDC name **
      5 RESUME               ** MPE command **
    6 //
...
/KEEP MYCMNDS
/E

END OF SUBSYSTEM
:SETCATALOG MYCMNDS;SYSTEM
                             ** Establishes a UDC directory at the system level **
                             ** for both commands. This requires SM capability **
```

Now, S and R are now established as system defined commands. If you enter S, the :SHOWJOB MPE command will execute; if you enter R, :RESUME will execute.

If, as Account Manager, you want to establish directory entries in the UDC file for users within your own account enter:

```
:SETCATALOG                  ** Requires AM capability **
NEWCMNDS;ACCOUNT
```

ADDITIONAL DISCUSSION

Refer to Section III of this manual.

Enables Stack Dump facility and initiates it on abort.

SYNTAX

```
          {DB}
:SETDUMP [{ST}[,...]]
          {QS}

   [,ASCII]
```

PARAMETERS

DB, ST, or QS	Dumps memory from the stack marker trace and registers at the time of abort in a specified job. DB dumps memory from DL to Q (initial) address. ST dumps memory from Q (initial) to S address. QS dumps memory from Q-63 to S address. The QS parameter is ignored if the ST parameter is used. Default is a display of the stack marker trace and all registers at the time of an abort.
ASCII	Displays ASCII conversion of octal values requested by DB, ST, or QS parameters, along with octal display, but only in a job. Default is octal.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable.

OPERATION

This command allows a display of the stack markers and all registers requested at the time of an abort. The stack marker trace is written to $STDLIST. In an interactive session all parameters are ignored, but if you have READ and WRITE file access to the program file, :SETDUMP will automatically initiate DEBUG.

EXAMPLES

To use :SETDUMP in an interactive session, enter:

```
:SETDUMP
```

Run the program ABC.MAC.TECHPUBS:

```
:RUN ABC.MAC.TECHPUBS
```

If the program ABC contains an error, it aborts, , writes the stack marker trace to the current list device, enters DEBUG, and displays the following error message:

```
ABORT :ABC.MAC.TECHPUBS.%0.%763
PROGRAM ERROR #18 :PROCESS QUIT .PARAM = 2
*** ABORT STACK ANALYSIS ***

S=001154    DL=177650    Z=003425
Q=001160 P=000763  LCST=  000  STAT=U,1,1,L,0,1,CCG   X=000035
Q=001153 P=177777  LCST= S026  STAT=P,1,0,L,0,0,CCG   X=000000

*DEBUG* PRIV.0.764
?E@                                          ** Terminates DEBUG **

PROGRAM TERMINATED IN AN ERROR STATE.  (CIERR 976)
:
```

To arm the Stack Dump Facility from a batch job, and display the memory area from the Q (initial) to S address, with accompanying ASCII conversion of the octal data, add the following line to your stream job:

```
!SETDUMP ST;ASCII
```

ADDITIONAL DISCUSSION

MPE Debug/Stack Dump Reference Manual (30000-90012).

:SETJCW

Creates or assigns a value to a Job Control Word (JCW).

SYNTAX

```
:SETJCW jcwname delimiter value [{+} value]
                                {-}
```

PARAMETERS

jcwname The name of a new or existing user or system defined Job Control Word (JCW). You can use "@" to specify all currently defined JCWs.

You may not specify the system reserved JCWs HPMINUTE, HPHOUR, HPDAY, HPDATE, HPMONTH, or HPYEAR.

delimiter One or more punctuation characters or spaces, except "%" and "-". Whatever character is used will delimit the name and value.

value One of the following:

1. An octal number between 0 and %177777, inclusive.
2. A decimal number between 0 and 65535, inclusive.
3. An MPE-defined JCW value mnemonic (OK for 0; WARN for 16,384; FATAL for 32,768; SYSTEM for 49,152) or an offset value of a mnemonic (OK3, etc).
4. The name of an existing JCW.

All specified values must be in the range of 0 to 65,535, inclusive. If the option "+" or "-" is used, the result of the indicated operation must also be within the range of 0 to 65,535, inclusive.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable.

JOB CONTROL WORDS

A Job Control Word (JCW) is a flag that allows information to be passed between processes within a single job or session. There are three forms of JCWs: system defined, user defined, and system reserved.

JCWs can be tested against specific values. The user can program conditional statements that act according to the results of these tests. The user defined JCWs can also be set to user-selected values so that they reflect the completion of steps within the setting process. System defined JCWs can be used to determine whether certain events have occurred within MPE.

The values in the system reserved JCWs can be inspected by the user, but not altered.

The contents of a JCW may be displayed using the :SHOWJCW command.

JCW Values and Mnemonics

JCWs may be assigned any positive integer value between 0 and 65,535 inclusive (%0 and %177777). These values are treated as 16-bit unsigned integers by MPE, since all 16 bits are used for numeric information, rather than using the most significant bit as a sign bit.

MPE treats the two most significant bits of a JCW in a special way: the bits define "bases" or "steps" of 16K each. Each of these steps is given a mnemonic to simplify references to it or to the numbers between steps. If the 14 least significant bits are considered to be zeros, the two "step" bits, step value (in decimal), and mnemonic have the following relationship:

Bit Value	Step Value	Mnemonic
00	0	OK
01	16,384	WARN
10	32,768	FATAL
11	49,152	SYSTEM

It is important to remember that these mnemonics are not the names of JCWs. They cannot be used as user defined JCW names.

You may use a combination of mnemonics and numbers to indicate numeric values between steps. If you specify a mnemonic and a number with no intervening spaces, an implied addition takes place. For example, "WARN3" has a value of 16,387, since it is "WARN" (16,384) plus 3. The value of the mnemonic plus the appended number value may not exceed 65,535. Again, no valid *value* of the form: mnemonic[*number*] may be used as a valid user defined *jcwname*. An explicit addition or subtraction can also be specified, using a "+" or "-" sign, as in "OK+7" (7) or fdWARN-4" (16,380). A mnemonic may also be added to another mnemonic: "WARN+FATAL".

The result of a mathematical operation must be in the range of 0 to 65,535; inclusive, if the number is out of range, an error message will be generated and the value of the JCW will remain unchanged. When the result of an operation is greater than the value of the next "step", the JCW value displayed by the :SHOWJCW command will be the mnemonic of the higher step plus any offset. For example, the value "OK+16385" will be displayed as "WARN1".

User Defined JCWs

User defined JCWs are created and initialized to a value by the :SETJCW command or PUTJCW intrinsic. The JCW name contains alphanumeric characters and must begin with an alphabetic character. The name can be up to 255 characters long. The value assigned to the JCW must be in the range of 0 to 65,535, inclusive.

The :SETJCW command scans MPE's JCW table for the name of the specified JCW (*jcwname*). If the specified name is found, the JCW is set to *value*. If the *jcwname* is not found, it is created and set to *value*. The term "*value*" as used here means the explicitly stated or the computed *value*.

You may not begin a JCW name with the mnemonic names OK, WARN, FATAL, or SYSTEM, unless you append a number to the mnemonic such that the computed value exceeds 65,535 (e.g. WARN999999, or SYSTEM20000). If the computed value exceeds 65,535, MPE will not recognize the term as a valid mnemonic, and will treat it as the name of a JCW. This restriction is intended to eliminate the possibility of an ambiguous JCW assignment. For example, it is unclear from the following two commands whether the JCW X is equal to 100 or to 0:

```
:SETJCW OK=100
:SETJCW X=OK
```

Naming a JCW with a mnemonic or predefined JCW value will result in an error message, as in the following example:

```
:SETJCW OK200=1982

JCWNAME CANNOT BE A VALID JCW VALUE (CIERR 1725)
```

Negative or out-of-range JCW values will cause the following error message to be displayed:

```
VALUE NOT IN RANGE--LEGAL RANGE IS 0 TO 65535 (CIERR 1712)
```

System Defined JCWs

MPE has two system defined JCWs that are created for each job and session: JCW and CIERROR. The JCW named "JCW" is always initialized to zero at the beginning of the job or session and will remain zero, unless fatal errors occur, or unless the user changes the value. There are two special values for the system defined JCW:

%140000 (System 0)	Program aborted per user request.
>%140000	Program terminated in an error state.

The "CIERROR" JCW tracks Command Interpreter (CI) errors. CIERROR is set to zero at the beginning of the job or session. If a Command Interpreter error occurs, CIERROR is updated to reflect the current CI error message number.

Users are advised not to alter the values of the "CIERROR" and "JCW" Job Control Words. User defined JCWs should be used for information the user wishes to control.

The following example shows the use of the CIERROR JCW:

```
:LISTTF
 ^
UNKNOWN COMMAND NAME.  (CIERR 975)
:SHOWJCW CIERROR
CIERROR = 975
:RUN
    ^
NO PROGRAM FILE SPECIFIED.  (CIERR 600)
:SHOWJCW CIERROR
CIERROR = 600
:
```

System Reserved JCWs

The system reserved JCWs are HPMINUTE, HPHOUR, HPDAY, HPDATE, HPMONTH, and HPYEAR. They contain system-assigned minute, hour, day, date, month, and year information. If the user attempts to assign values, an error message is displayed. You can retrieve the values in these JCWs with the FINDJCW intrinsic. The values can also be tested if the JCW is used with an :IF command. The names of the system reserved JCWs are reserved.

The following lists system reserved JCWs and possible values:

HPDAY	Day of the week. The possible integers are 1 through 7. Sunday is indicated by 1. Saturday is indicated by 7.
HPDATE	Day of the month. The possible integers are 1 through 31.
HPMONTH	Month of the year. The possible integers are 1 through 12. January is indicated by 1.
HPYEAR	Year of the century. The possible integers are 00 through 99.
HPHOUR	Hour of the day. The possible integers are 0 through 23.
HPMINUTE	Minute of the hour. The possible integers are 0 through 59.

Conditional Execution Using JCWs

JCWs are typically used to control the flow of batch jobs, based on events that take place within the job. You can use the MPE :IF/THEN and :ELSE statements to test JCW values.

The following example illustrates a conditional execution function. The sample job runs a program that edits, verifies, and counts valid transactions (CHEKPROG). If no fatal errors occur, the job runs the program SHIPPROG, which schedules shipments. The job then runs FINALRPT, which produces a final report. If fatal errors do occur, the CHECKPROG will set the value of the the JCW "CHEKPROGSTAT" to FATAL, and SHIPPROG is not run. Instead, ERRORRPT is run, which produces an error report. A final report is also produced by FINALRPT.

You can display the contents of a system defined JCW with the :SHOWJCW command only if you specify the *jcwname*. Otherwise, :SHOWJCW will not display system defined JCWs.

```
!SETJCW CHEKPROGSTAT=OK
!CONTINUE                          ** Prevents abort in case of errors.          **
!RUN CHEKPROG                      ** Edit, verify, and count valid transactions. **
!  IF CHEKPROGSTAT<FATAL THEN      ** If no fatal errors, schedule shipments.    **
!    RUN SHIPPROG                  ** Schedule shipments.                        **
!  ELSE
!    SHOWJCW CHEKPROGSTAT
!    RUN ERRORRPT                  ** Produce error report and reset JCW to 0.   **
!  ENDIF
!RUN FINALRPT                      ** Produce final report.                      **
```

EXAMPLES

To set the Job Control Word CURR1 to 100 , enter:

```
:SETJCW CURR1,100    ** The comma (,) is the delimiter and is used as an "=". **
```

To set CURR1 to the value of the mnemonic WARN , enter:

```
:SETJCW CURR1/WARN   ** The slash (/) is the delimiter and is used as an "=". **
```

To use an arithmetic operation to set a JCW, enter:

```
:SETJCW XX=5+6
```

To schedule a full backup job on Saturdays and a partial backup job on the other days of the week, enter:

```
:SETJCW FRIDAY=6
:IF HPDAY = FRIDAY THEN
  SCHEDJOB FULLBKUP;IN=1
:ELSE
  SCHEDJOB PARTBKUP;IN=1
:ENDIF
```

:SETMSG

Enables or disables the receipt of user or operator messages at the standard list device.

SYNTAX

```
:SETMSG {OFF}
        {ON }
```

PARAMETERS

OFF	Sets job or session to quiet mode and blocks the receipt of :TELL command messages from other users.
ON	Enables user or operator messages to be received and displayed at the standard list device.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable.

OPERATION

Allows job or session to receive or block :TELL messages from other users. :WARN messages from the System Operator override quiet mode and are received and displayed.

EXAMPLE

To block messages, enter:

```
:SETMSG OFF
```

To receive messages, enter:

```
:SETMSG ON
```

:SHOWALLOW

Displays which Operator commands have been allowed:

SYNTAX

```
                {user.acct}
:SHOWALLOW [    {user.@   }]
                {@.acct   }
                {@.@      }
```

PARAMETERS

user	Defines a particular user.
acct	Defines a particular account.
@	All users, if used in place of *user*, or all accounts, if substituted for *acct*.
	Default is that the logged-on user, account, and group are displayed.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is Breakable (aborts execution). The user must have Account Manager (AM) or System Manager (SM) capability to execute this command for other groups or accounts.

OPERATION

This command displays which Operator commands have been allowed to specific users if the *user.acct* form is entered. If the @.@ form is entered, the commands allowed to all users in all accounts are displayed. System Manager (SM) capability is required to specify @.@; Account Managers (AM capability) may specify all users in their own account. When :SHOWALLOW is executed from the system console, @ may be substituted for *user* and/or *acct*. In addition, :SHOWALLOW separately lists which Operator commands have been globally allowed.

EXAMPLE

To list the Operator commands allowed to the user MAC,PUB.SYS, enter:

```
:SHOWALLOW USER.SYS
#S86  USER.SYS
    USER HAS THE FOLLOWING COMMANDS ALLOWED:
ABORTIO          ACCEPT           DOWN             GIVE

THERE ARE NO GLOBAL ALLOWS DEFINED.
```

ADDITIONAL DISCUSSION

Refer to the :ALLOW command.

:SHOWCACHE

Displays a summary of disc caching performance. Disc caching software is not included in the Fundamental Operating Software, and must be purchased separately.

SYNTAX

```
:SHOWCACHE
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable.

OPERATION

The :SHOWCACHE command summarizes the performance of disc caching on your system, displaying nine columns of information (shown below). Since it only allows you to look at disc caching performance, and prohibits the modification of caching parameters, any user may execute :SHOWCACHE.

```
:SHOWCACHE

DISC    CACHE     READ WRITE          PROCESS                 % OF     CACHE
LDEV  REQUESTS    HIT%  HIT%  READ%   STOPS       K-BYTES   MEMORY   DOMAINS
-------------------------------------------------------------------------

1      21529       93    85    82     1705         686        11      218
2      35093       77    59    69     6155         2760       45      675

Total  56622       83    65    74     7860         3446       57      893

62% of user I/Os eliminated.
Data overhead is 229K bytes.
Sequential fetch quantum is 96 sectors.
Random fetch quantum is 16 sectors.
Block on Write = NO.
```

Each column of information provides the following information:

DISC LDEV	Logical device number(s) of the disc(s) being cached. Read across the row to see the other entries on a particular disc. The bottom row is a total across all discs.
CACHE REQUESTS	Tally of the number of logical disc I/O requests against caching. This is a double-word counter and resets to zero automatically when caching is started on the disc, or when the counter overflows.
READ HIT%	Cumulative percentage of logical disc I/O on read requests: read requests that were satified by data found in a main memory cache domain when the read request occurred.
WRITE HIT%	Cumulative percentage of logical disc I/O on writes requests: write requests that were accomplished in a main memory cache domain when the write request occurred.
READ%	Percentage of logical disc I/Os in the system which were read requests.
PROCESS STOPS	Number of times a process stopped in order to complete a cache I/O request. Process Stops occur due to read misses, or due to write hits in which the cached region is currently being posted to disc. This is a double-word counter which resets to zero if it overflows.
K-BYTES	Count of the number of thousands of bytes (1024-byte groups) which are currently being used for main memory cache domains.
% OF MEMORY	Percentage of the total memory in the machine which is currently allocated for use by cache domains.
CACHE DOMAINS	Number of cache domains in memory.

In addition, the :SHOWCACHE command displays information about data overhead, which is the current dynamic data structure overhead used to support disc caching. The sequential fetch and random fetch quantums, established by default or selected with the :CACHECONTROL command, specify the number of sectors read from the disc for sequential and random access files respectively. BLOCKONWRITE, which is also set with the :CACHECONTROL command, determines whether or not MPE will block the process until posting of the cache buffers to disc is complete.

ADDITIONAL DISCUSSION

The :SHOWCACHE command does not require any special capabilities. However, you are allowed only to look at disc caching performance and not to change any of the options or parameters. If you want to change the disc caching performance, refer to the :STARTCACHE, :STOPCACHE, and :CACHECONTROL commands in this manual. Refer also to the FSETMODE intrinsic in MPE V Intrinsics Manual (32033-90007).

:SHOWCATALOG

Lists user defined command (UDC) files.

SYNTAX

```
:SHOWCATALOG [listfile]
             [;USER=user[.acct]
```

PARAMETERS

listfile — An arbitrary filename, used to identify the UDC file when the output of :SHOWCATALOG is sent to a printer or stored on disc. Default is that UDC files are displayed on your terminal (the $STDLIST device).

user[.*acct*] — A particular user whose UDC filenames will be displayed on the screen or sent to a listfile; the contents of the files are not displayed.

The wildcard characters @, #, and ? may be used in *user*[.*acct*] as you would use them in the specification of any user or any account, where

@	One or more alphanumeric characters. Used by itself, @ denotes "all members of the set".
#	One numeric character.
?	One alphanumeric character.

Each wildcard character counts toward the eight character limit for *user* or *acct*.

USE

This command may be issued from a Session, Job, or in BREAK. It may not be issued from a Program. It is Breakable (aborts execution).

OPERATION

The :SHOWCATALOG command lists user defined command (UDC) files. Unless the output is redirected to another device with the :FILE command, as shown in "EXAMPLE", the listing appears on your terminal. It indicates at which level (user, account, or system) the files have been defined, so that you may display the current catalog before adding or deleting system level UDCs (if you are the System Manager) or account level UDCs (if you are Account Manager).

EXAMPLES

To list system, account, and user level UDC files on your terminal, enter:

```
:SHOWCATALOG
```

To store the current UDC file on disc as "LFILE" (which may also be printed) enter:

```
:BUILD LFILE
:FILE LFILE,OLD;DEV=DISC
:SHOWCATALOG *LFILE
```

ADDITIONAL DISCUSSION

Refer to the discussion of UDCs in Section III of this manual.

Displays status information about a communication device.

SYNTAX

```
:SHOWCOM ldev[;ERRORS][;RESET]
```

PARAMETERS

ldev	The logical device number of a communication system device.
ERRORS	Displays the full status list. If not specified, an abbreviated list is displayed.
RESET	Reset all status information to zero after it has been displayed.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

The status information generated by this command can be used to determine communication line activity and quality. It includes:

- Number of messages sent and received.
- Last recoverable and irrecoverable errors.
- Number of recoverable and irrecoverable errors.
- Number of retransmissions, response timeouts, clear-to-send losses, and underruns.
- Number of check character (BCC/CRC) errors, receive timeouts, carrier losses, and overruns.
- Line state (closed, connected or disconnected).

Note that clear-to-send losses and carrier losses will be valid only for communication lines configured as full duplex.

When the line is OPEN, connected or disconnected, the status information displayed is a duplicate of the statistics generated from the last opening of the line until :SHOWCOM command is invoked. When the line is CLOSED, the status information obtained at the close of the line reflects the statistics generated during the last open/close sequence.

EXAMPLES

To display the status of logical device 18, enter:

```
:SHOWCOM 18

                        LDN - 18
MESSAGES SENT          0              MESSAGES RECVD 0
            LAST RECOVERABLE ERROR          0
            LAST IRRECOVERABLE ERROR        201
            LINE IS CLOSED
```

To display the full status of logical device 18, enter:

```
:SHOWCOM 18;ERRORS

   TRANSMIT             LDN - 18             RECEIVE
MESSAGES SENT          0              MESSAGES RECVD 0
RETRANSMISSIONS        0              BCC/CRC ERRORS 0
RESPONSE TIMEOUTS      0              RECV TIMEOUTS  0
UNDERRUNS              0              OVERRUNS       0
CLR TO SEND LOSSES     0              CARRIER LOSSES 0
            # OF RECOVERABLE ERRORS
            LAST RECOVERABLE ERROR
            # OF IRRECOVERABLE ERRORS
            LAST IRRECOVERABLE ERROR
            LINE IS CLOSED
```

NOTE

The "LAST IRRECOVERABLE" and "LAST RECOVERABLE" error codes are specified in the Communications Handbook (30000-90105).

:SHOWDEV

Reports the status of input/output devices.

SYNTAX

```
:SHOWDEV [ldev      ]
         [classname]
```

PARAMETERS

ldev — Logical device number of device for which status information is to be displayed. This number is unique for each device. Default is that status information for all system devices on the system is displayed.

classname — Device class name of device(s) for which status information is to be displayed. This name may apply to several devices. Default is that status information for all devices on the system is displayed.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is Breakable (aborts execution).

OPERATION

Displays the status information for all input/output devices on the system. The display appears in the following format:

```
:SHOWDEV
LDEV AVAIL     OWNERSHIP        VOLID        DEN   ASSOCIATION

  1   DISC     43 FILES
  6   SPOOLED  SPOOLER OUT
  8   AVAIL
 20 A UNAVAIL  #S311:  2 FILES
```

The following items may appear in the listing, always displayed on the standard list device:

COLUMN	MEANING
LDEV	Includes the logical device number and may include one of the following:
	J Accepts jobs.
	D Accepts data.
	A Accepts jobs and data.
AVAIL	Lists the availability of devices and discs as follows:
	AVAIL — The device is available as a real, nonsharable device.
	SPOOLED — The device is available via input or output spooling.
	UNAVAIL — The device is not available (it is under the control of a job, session or a system process such as a spooler).
	DISC — The device is a disc and is always available.
	DISC (RPS) (G.00.00) — The device is a CS-80 disc on which Rotational Position Sensing (RPS) has been enabled.
OWNERSHIP	Includes device ownership and may include one of the following:
	SYS — Controlled by the system. If *#nnn* appears, it specifies the Process Identification Number (PIN) of the controlling process (program).
	SPOOLER IN — Input spooling in effect, controlled by spooler.
	SPOOLER OUT — Output spooling in effect, controlled by spooler.
	#Jnnn — Controlled by the indicated job.
	#Snnn — Controlled by indicated session.
	DIAG — Device has been allocated to diagnostic testing by System Operator via the :GIVE command.
	nn FILES — Indicates number of files currently in use on a disc.
	DOWN — Device is offline, requested by System Operator via the :DOWN command.
	DP — Device is being taken offline (:DOWN command operation pending).

COLUMN	MEANING	
VOLID	The volume identification and may include one of the following:	
	IBM	The named magnetic tape volume that has a label written in the IBM format.
	ANSI	The named magnetic taped volume that has a label written in the ANSI format.
	NOLABEL	The named magnetic tape volume that has no label. Default.
DEN	Density of the tape and may include one of the following:	
	6250	Density of 6250 BPI (bytes-per-inch).
	1600	Density of 1600 BPI or the density of the tape is unrecognizable.
ASSOCIATION	Indicates the logical devices by device class which have been established by the user with the :ASSOCIATE command.	

EXAMPLES

To display the status of the device identified by logical device number 5 enter:

```
:SHOWDEV 5
LDEV  AVAIL     OWNERSHIP          VOLID          DEN   ASSOCIATION

  5   SPOOLED   SPOOLER OUT
```

To display the status of all devices of the device class CARD, enter:

```
:SHOWDEV CARD
LDEV  AVAIL     OWNERSHIP          VOLID          DEN   ASSOCIATION

  6 A AVAIL
```

:SHOWIN

Reports the status of input devicefiles.

SYNTAX

```
          [#Innn              ]
:SHOWIN   [STATUS             ]
          [SP                 ]
          [item[;item[;...]]]
```

PARAMETER

#Innn — Identifier of particular input devicefile for which information is to be displayed. Default is that MPE displays information for all input devicefiles used by the logon job or session.

STATUS — Summarizes the status information for all current input devicefiles. Default is that MPE displays information for all input devicefiles used by the logon job or session. The information appears in following format:

```
8 FILES
   0 ACTIVE
   0 READY;INCL 0 SPOOFLES, 0 DEFERRED
   8 OPENED; INCL 0 SPOOFLES
   0 SPOOFLES; 0 SECTORS
0 LOCKED; INCLUDING 0 SPOOFLES
```

SP — Displays the status information for the currently spooled input devicefiles associated with the logon job or session. Default is a display of status information for all input devicefiles.

item — Displays the status of current input devicefiles as identified. Default is that MPE displays status information for all input devicefiles used by this job. The syntax appears on the next page.

Syntax for Item

```
[DEV=ldev      ]

     {@J      }
     {@S      }
[JOB={@       }]
     {[#]Jnnn}
     {[#]Snnn}

 {ACTIVE   }
[{OPENED   }  ]
 {READY    }
```

Parameters for Item

ldev

Displays the status of input devicefiles identified by logical device number *ldev*.

JOB=

Displays the status of input devicefiles. JOB= may be one of the following options:

@J	Displays the status of input devicefiles for all jobs.
@S	Displays the status of input devicefiles for all sessions.
@	Displays the devicefiles for all jobs and sessions.
[#]*Jnnn*	Displays the status of all input devicefiles for a specified job.
[#]*Snnn*	Displays the status of all input devicefiles for a specified session.

Default is @.

ACTIVE, OPENED, or READY

Displays the status of all input files in a specified state. ACTIVE displays the status of ACTIVE devicefiles. OPENED displays the status of OPENED devicefiles. READY displays the status of READY devicefiles.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is Breakable (aborts execution).

OPERATION

This command displays the status information about one or more currently defined input devicefiles. This information reflects the status at the time the command is entered, and always appears on the standard list device. Except for the keyword STATUS, which has its own format (refer to "Parameter"), the format of the information is:

```
DEV/CL   DFID     JOBNUM  FNAME    STATE FRM SPACE RANK PRI #C
10      #I1088    #J133   $STDIN   OPENED
```

The information displayed in this format is defined as follows:

COLUMN	MEANING
DEV/CL	Logical device number of device.
DFID	Device file identification in the form *#Innn*.
JOBNUM	Job or session number (*jsnum*) of job/session using the devicefile, if not used for READY or ACTIVE data. Otherwise, the job/session name appears on the line following standard device information.
FNAME	File name associated with the devicefile.
STATE	One of the following:
	ACTIVE — Input being read from a spooled device to a disc.
	READY — Input spooling completed: file is now ready for use by a program.
	OPENED — A file is being accessed by a program.
FRM	Forms message indicator. The letter "F" will appear only if a forms alignment message applies to the devicefile. Does not apply to input files.
SPACE	Approximate disc space currently used (in sectors), for jobs only.
RANK	The order in which the file is entered into the system with respect to other files of the same priority and class name or logical device.
	A "D" following "RANK" indicates a deferred file for spooled devicefiles only. A file can be deferred if its priority is less than or equal to the system outfence or the outfence of a specific device.
PRI	The outpriority of the devicefile, requested by the user or adjusted by the System Operator. Specified for spooled output devicefiles only.
#C	The number of copies needed, specified for spooled output devicefiles only.

EXAMPLES

The following is an example of how to determine the status of an individual input devicefile:

```
:SHOWIN #I80


DEV/CL  DFID  JOBNUM FNAME     STATE  FRM SPACE RANK PRI #C
43      #I80  #S37   $STDIN    OPENED           8
```

If you do not know the devicefile identification number (DFID) of the devicefile whose status you want to determine, you may request the status display by entering either the logical device number or the device class name of the device on which the file originated:

```
:SHOWIN DEV=43

DEV/CL  DFID  JOBNUM FNAME     STATE   FRM SPACE RANK PRI #C
43      #I80  #S37   $STDIN    OPENED
```

You may also request displays of devicefile information using various combinations of qualifications (devices, jobs/sessions, and states). For example, to display information about all OPENED input devicefiles used by all sessions (but not jobs) in the system, enter:

```
:SHOWIN JOB=@S;OPENED

DEV/CL  DFID  JOBNUM FNAME     STATE  FRM SPACE RANK PRI #C
7       #I81  #S38   MASTER    OPENED
26      #I36  #S18   $STDIN    OPENED
32      #I85  #S41   $STDIN    OPENED
34      #I58  #S26   $STDIN    OPENED
42      #I64  #S28   $STDIN    OPENED
43      #I80  #S37   $STDIN    OPENED
50      #I84  #S40   $STDIN    OPENED
51      #I35  #S17   $STDIN    OPENED

8 FILES (DISPLAYED):
    0 SPOOFLES:  0 SECTORS
```

:SHOWJCW

Displays current state of one or more Job Control Words (JCWs).

SYNTAX

```
:SHOWJCW [jcwname]
```

PARAMETERS

jcwname The name of a valid Job Control Word (JCW). Default is that all user and system defined JWCs are displayed.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is Breakable (aborts execution).

OPERATION

The :SHOWJCW command is used to display the current state of one or more Job Control Words, JCWs. Specifying a particular JCW (user or system defined , or system reserved) displays the state of that particular JCW. If you do not specify a particular JCW, user defined and system defined JCWs are displayed. The value of the third type of JCW, system reserved JCW, is displayed only if you specifically enter its *jcwname*.

You may retrieve the value assigned a JCW with the FINDJCW intrinsic, then test it with an :IF command. In this way, the value of a given JCW can be used to conditionally execute another instruction or set of instructions. For example:

```
!CONTINUE
!SPL MYPROG,MYUSL
!IF JCW>=FATAL THEN
!   TELL MAC.TECHPUBS;COMPILE FAILED
!ELSE
!   TELL MAC.TECHPUBS;COMPILE COMPLETED
!ENDIF
```

EXAMPLE

To show the current state of all user and system defined JCWs, enter:

```
:SHOWJCW
JCW = 0
CIERROR = 0
```

To display the current state of a valid user defined Job Control Word named JCW1, enter:

```
:SHOWJCW JCW1
JCW1=3
```

To display the contents of a system reserved JCW, enter:

```
:SHOWJCW HPDAY
HPDAY=4
```

ADDITIONAL DISCUSSION

Refer to :SETJCW in this section.

:SHOWJOB

Displays the status information about jobs/sessions.

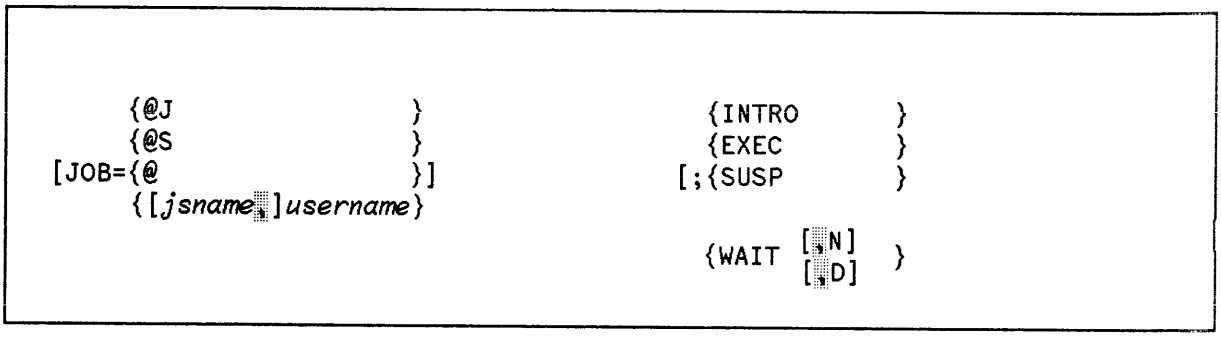
SYNTAX

```
                [[#]Snnn              ]
                [[#]Jnnn              ]
:SHOWJOB        [STATUS               ]          [;*listfile]
                [SCHED                ]
                [item[;item[;...]]]
```

PARAMETERS

[#]S*nnn*	The session number (assigned by MPE) of the session for which the status information is to be displayed. The information appears in Type I format, described under "OPERATION". Default is that the status information for all jobs/sessions is displayed.
[#]J*nnn*	The job number (assigned by MPE) of the job for which status information is to be displayed. The information is in Type I format, described under "OPERATION". Default is that the status information for all jobs/sessions is displayed.
STATUS	Lists the number of jobs and sessions in each processing state and the current job fence and job/session limits. This information is in Type II format, described under "OPERATION". Default is that the status information for all job/sessions is displayed.
SCHED	Displays only the scheduled jobs. The information is in Type III format, described under "OPERATION".
**listfile*	Formal file designator of the file on which the output listing is written. A backreference to a :FILE equation is required. The *listfile* is a temporary file with record size of 256 bytes, blocked one record per block, with Carriage Control (CCTL), with the time and date displayed. You can override the default characteristics of *listfile* with the :FILE command. Default is $STDLIST.
item	A list of jobs/sessions whose status is displayed. Default is that the status information for all jobs/sessions is displayed. The syntax appears below.

Syntax for Item



Parameters for Item

JOB=

A list of jobs/sessions for which status information is to be displayed. Use one of the following options:

@J	Displays status information for all jobs.
@S	Displays status information for all sessions.
@	Displays status information for all jobs and sessions.
[*jsname*.]*.username* *.acctname*	The *jsname* is an optional name given to the session or job by the user. The *username* parameter is the username established by the Account Manager. This name may consist of one to eight alphanumeric characters beginning with an alphabetic character. The *accountname* parameter is the name of the account established by the System Manager. This name may consist of one to eight alphanumeric characters beginning with an alphabetic character. The "@" can be used to replace the *jsname* or *username* in a specified account.

INTRO, EXEC, SUSP, or WAIT

Displays the status of all jobs or sessions in a specified state. INTRO means that the job is introduced. In this case the spooler process validates the :JOB command and, if the job is legitimate, copies the job input records to disc. EXEC means that the job is executing. SUSP means that the job or session is suspended, because table entries or system resources are unavailable. WAIT means that there are no available list devices for the job. WAIT has the following subparameters:

N	Displays the status of nondeferred READY devicefiles.
D	Displays the status of deferred READY devicefiles.

If information for only one devicefile is displayed, output is in Type I format; if information for more than one devicefile is displayed, output is in Type I followed by Type II format. (Format types are described under "OPERATION".)

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is Breakable (aborts execution).

OPERATION

This command enables you to determine the number of jobs and sessions in each processing state, the current job fence and job/session limits, and allows you to keep track of individual spooled and streamed jobs that are entered in the system. The output appears in two possible formats or in a combination of both formats:

Type I:

```
JOBNUM  STATE IPRI  JIN  JLIST    INTRODUCED JOB NAME

#S16     EXEC         45    45     MON  7:08A TEST.PUBS

JOBFENCE= 0;   JLIMIT = 3;   SLIMIT= 16
```

Type II:

```
7 JOBS:
   0 INTRO
   0 WAIT; INCL 0 DEFERRED
   7 EXEC; INCL 7 SESSIONS
   0 SUSP
JOBFENCE= 0;   JLIMIT= 3;   SLIMIT= 16
```

If the :SHOWJOB SCHED command is used, the output is displayed as shown below. The STATE field shows that the job is scheduled. The SCHEDULED-INTRO field shows the time and date the job will be introduced to the system. Note that the scheduled jobs are listed in the order in which they will be introduced to the system. If you enter only the :SHOWJOB command, the formatted output for jobs and sessions in the INTRO, WAIT, and EXEC states will be displayed first in the Type I and Type II formats. The formatted data for jobs in the SCHED state is displayed last and is in the Type III format.

Type III:

```
CURRENT: 5/13/85 1600

JOBNUM  STATE IPRI  JIN  JLIST    SCHEDULED-INTRO   JOB NAME

#J38       SCHED   3    10   6       5/16/84 11:24     NOTHING,JON.OSE
#J23       SCHED   8    10   PP      5/25/84  8:01     REPORT,MGR.OSE
#J25       SCHED   8    10   LP       7/4/84 18:05     FIREWORK,UNCLE.SAM

3 SCHEDULED JOB(S)
```

EXAMPLES

To determine the number of jobs and sessions in each processing state, the current job fence and the job/session limits, enter:

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

To get a report on all jobs and sessions in the system, enter:

```
:SHOWJOB

JOBNUM   STATE IPRI  JIN  JLIST    INTRODUCED JOB NAME
#S745    EXEC         29  29       MON  2:53P DL,SPL.ALANG
#S746    EXEC         26  26       MON  2:53P CLI.AOPSYS

2 JOBS:
    1 INTRO
    0 WAIT; INCL 0 DEFERRED
    2 EXEC; INCL 2 SESSIONS
    0 SUSP
JOBFENCE= 2; JLIMIT= 1; SLIMIT= 16
```

The following example of a :SHOWJOB command sequence illustrates an override of the default characteristics of *listfile* with the :FILE command, and shows the output produced with the new *listfile* characteristics:

```
:FILE A;REC=40,1,F,ASCII;NOCCTL
:SHOWJOB;*A
:SAVE A
:FCOPY FROM=A;TO=

HP32212A.3.17 FILE COPIER (C) HEWLETT-PACKARD CO. 1982

MON, MAY 7, 1985,  7:54 AM

JOBNUM   STATE IPRI  JIN  JLIST    INTRODUCED JOB NAME
#S46     EXEC         20  20       MON  7:14A OPERATOR.SYS
#S45     EXEC         47  47       MON  6:37A MAC.PUBS
#S47     EXEC         10S LP       MON  7:26A SUPPORT.DOC
#S48     EXEC        102  102      MON  7:28A MAC.TECH
#J19     EXEC         28  28       MON  6:41A JON.OSE
#S49     EXEC*        34  34       MON  7:31A FLASH.G
#J21     EXEC         10S LP       MON  7:15A DELIVER,MAIL.MAIL
#J22     EXEC         10S LP       MON  7:14A RSPOOLJ,RSPOOL.SYS

8 JOBS (DISPLAYED):
    0 INTRO
    0 WAIT; INCL 0 DEFERRED
    8 EXEC; INCL 5 SESSIONS
    0 SUSP
```

```
JOBFENCE= 6;   JLIMIT= 4;   SLIMIT= 50
EOF FOUND IN FROMFILE AFTER RECORD 17

18 RECORDS PROCESSED *** 0 ERRORS

END OF SUBSYSTEM
:
```

The :SHOWJOB command reports a job or session as being in EXEC* when it is initializing. After initialization is complete, the state will change to EXEC . The number shown in the EXEC state is the sum of the jobs and sessions in both EXEC and EXEC*.

:SHOWLOG

Displays the number of system's current log file and the percentage of disc space used.

SYNTAX

```
:SHOWLOG
```

PARAMETERS

None.

USE

This command may be issued from Session, Job, in BREAK, or from a Program. It is not Breakable. The user must have System Supervisor or (OP) capability, or issue the command from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

The log file number, "*xxxxx*", and percentage of file space used, "*yy*", is displayed in the format:

```
LOG FILE LOG xxxxx IS yy% FULL
```

If the Logging System is disabled, MPE displays a "NO LOGGING" message. If logging is enabled but currently suspended due to an error, both messages appear.

EXAMPLE

To display the current log file status, enter:

```
:SHOWLOG

LOG FILE LOG2075 IS 20% FULL
```

ADDITIONAL DISCUSSION

Refer to the discussion of system logging in "MPE LOGGING FACILITIES" in Section of the MPE V System Operator and Resource Management Reference Manual (32033-90005).

:SHOWLOGSTATUS

Displays status information about currently opened user logging files assigned to a logging identifier.

SYNTAX

```
:SHOWLOGSTATUS [logid]
```

PARAMETERS

logid — Displays the logging identifier created by the :GETLOG command. Default is that the status of all logging identifiers is displayed.

USE

This command may be issued from a Session, Job, or in BREAK. It may not be issued from a Program. It is not Breakable.

OPERATION

This command lists the status of currently running logging processes. The status includes the total number of records written by the process and the number of users accessing the logging file. This command gives the following information about all currently running logging processes. To display the status of an open log file named LEN, enter:

```
:SHOWLOGSTATUS LEN

LOGID     CHANGE  AUTO  USERS  STATE     CUR-RECS  MAX-REC  %USED  CUR-SET

DUMMYLOG    NO     NO      4   INACTIVE     100      1000     10%      1
TAPELOG     YES            1   INACTIVE    5738                       1
DISCLOG     YES    YES     2   INACTIVE     500      1000     50%      2
```

COLUMN	MEANING
LOGID	The name of the logging process.
CHANGE	Whether the :CHANGELOG command is permitted (whether the name of the first logging file ends in "001".
AUTO	Whether an automatic :CHANGELOG has been enabled (whether the AUTO parameter has been specified for the :ALTLOG or :GETLOG command).
USERS	The number of users accessing the logging file.
STATE	ACTIVE, INACTIVE, INITIALIZING, or RECOVERING.

	INACTIVE is displayed when a process is waiting for information from user processes that involve intrinsics. INITIALIZING starts the log process. RECOVERING is displayed immediately after a WARMSTART recovery.
CUR-RECS	The number of records in the log file.
CUR-RECS	The number of records in the log file.
MAX-REC	The maximum number of records permitted.
%USED	The percentage of the maximum used.
CUR-SET	The current set number.

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32002-90005)

:SHOWME

Reports job/session status.

SYNTAX

```
:SHOWME
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is Breakable (aborts execution).

OPERATION

To display the status of the current job/session enter:

```
:SHOWME
USER: #S485,MGR.DSUSER,PUB       (NOT IN BREAK)
MPE VERSION: HP32033G.02.00.  (BASE G.02.00).
CURRENT: MON, MAY  7, 1984, 11:09 AM
LOGON:   MON, MAY  7, 1984, 11:08 AM
CPU SECONDS: 3          CONNECT MINUTES: 1
$STDIN LDEV: 88          $STDLIST LDEV: 88
                           (WELCOME TO SYS1)      ** Local system's welcome message **
```

The :SHOWME command reports information pertaining to your job or session. Using the example above, the following information is given:

ITEM	MEANING
#S485	This is the session number. It may also be a job number.
(NOT IN BREAK)	An "(IN PROGRAM)", "(IN BREAK)", or "(NOT IN BREAK)" message to indicate whether :SHOWME was executed programmatically, in BREAK, or directly from the MPE command.
HP32033G.02.00	This is user defined version of the software that you are running. You can modify this number by using the :SYSDUMP command. In this example the "E" is the user defined version, the first "00" the user defined update number, and the last "00" the user defined fix number.

(BASE G.02.00) This number is the base MPE *version.update.fix* numbers. It is the official Hewlett-Packard version number of the operating system software and cannot be modified by :SYSDUMP.

CURRENT Shows the current time and date.

LOGON Shows the logon time.

CPU SECONDS Shows the Central Processor Time (CPU) used by this job/session.

NOTE

:SHOWME calculates CPU usage by adding the local CPU usage of the current process to the accumulated total of all terminated processes. The CPU usage listed for a programmatic :SHOWME, therefore, would rarely agree with that for a :SHOWME executed during BREAK.

CONNECT MINUTES The amount of time job/session has been connected.

$STDIN LDEV The logical device number of the job or session's standard input device.

$STDLIST LDEV The standard list device number.

(WELCOME TO SYS1) Shows the current welcome message, if the System Operator has entered one.

Displays the status of output devicefiles.

Syntax

```
                {#Onnn              }
:SHOWOUT [      {STATUS             }]
                {SP                 }
                {item[;item[;...]]}
```

PARAMETERS

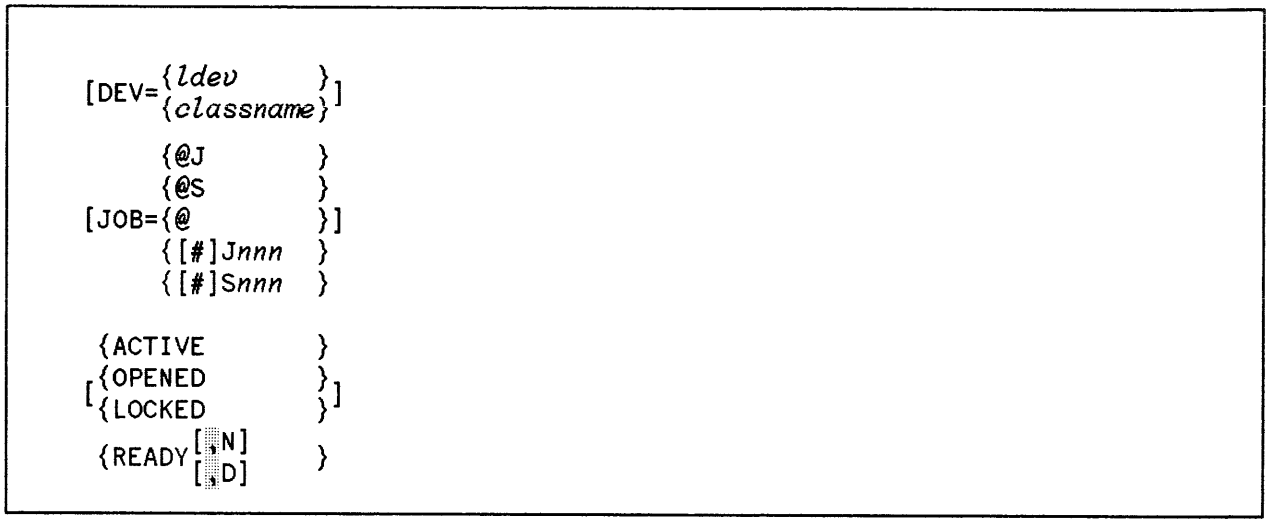
#Onnn — Identifies a particular output devicefile for which information is to be displayed. The information is displayed in Type I format, described under "OPERATION". Default is a display of status information for all output devicefiles used by the logon job or session.

STATUS — Summarizes the status information for all current output devicefiles. The information is displayed in Type II format, described under "OPERATION". Default is a display of status information for all output devicefiles used by the logon job or session.

SP — Displays the status information for currently spooled output devicefiles associated with the logon job or session. The information is displayed in Type I followed by Type II format, described under "OPERATION". Default is a display of status the information for all output devicefiles used by the logon job or session.

item — Displays the status of all current output devicefiles as identified. If information for only one devicefile is displayed, output is in Type I format; if information for more than one devicefile is displayed, output is in Type I followed by Type II format. Refer to "OPERATION".) The syntax appears on the next page.

Syntax for Item



Parameters for Item

ldev or *classname*

Displays the status of output devicefiles. The *ldev* parameter displays the files residing on the device identified by the logical device number. The *classname* parameter displays the status of the output devicefiles residing on all devices in a class name.

JOB=

Displays the status of output devicefiles using one of the following options.

@J	Displays the status of output devicefiles for all jobs.
@S	Displays the status of output devicefiles for all sessions.
@	Displays the output devicefiles for all jobs and sessions.
[#]J*nnn*	Displays all output devicefiles for specified job.
[#]S*nnn*	Displays the status of all output devicefiles for a specified session.

ACTIVE, OPENED, READY, or LOCKED

Displays status of all output files in a specified state. An ACTIVE file is one that is currently being produced on your printer, plotter, or punch card. Only one output spoolfile can be ACTIVE at any one time. OPENED files are those being accessed by a program. A spoolfile will be OPENED when a spooler process is writing the file to disc; during that time, however, the file is not ready to be printed, plotted or punched. READY files are completely spooled and ready to be output. A LOCKED file is READY but cannot be accessed until the system relinquishes it exclusive use of the file.

READY files may include one of the following:

N	Displays the status of nondeferred READY devicefiles.
D	Displays the status of deferred READY devicefiles.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is Breakable (aborts execution).

OPERATION

This command displays the status information for one or more currently defined output devicefiles. The information reflects the status at the time the command is entered and always appears on the standard list device. Two types of spooling queues are maintained in MPE: one output queue for each logical device configured on the system, and one additional queue for all device classes. Within each queue, files are linked according to the following parameters and listed in descending order of importance: output priority, device class, and rank. If the priorities are equal, the spooler alternates between queues. To list information about an individual output devicefile, you may specify its devicefile identifier (DFID) in the :SHOWOUT command:

```
:SHOWOUT #O11470

DEV/CL   DFID     JOBNUM  FNAME     STATE FRM SPACE RANK PRI #C
EPOC     #O11470 #J242   $STDLIST  READY          36    D 1   1

OUTFENCE = 6
```

COLUMN	MEANING
DEV/CL	Logical device number or device class name of the device.
DFID	Devicefile identification, in the form *#Onnn*.
JOBNUM	The job/session number (*jsnum*) of job/session using the devicefile if not used for READY or ACTIVE data. Otherwise, the job/session name appears online following standard device information.
FNAME	Filename assigned to devicefile.
STATE	The status, indicated by one of the following subparameters:
	ACTIVE — The spooled devicefile on disc is actually being written to a printer, plotter, or card punch.
	OPENED — The devicefile on disc is being accessed by a program. If the devicefile is spooled, a program is currently writing to the disc.

READY	READY, but the system has exclusive access to the file.
LOCKED	The spooled devicefile on disc is ready for output.
FRM	The forms message indicator (the letter "F") appears only if a forms alignment message applies to this devicefile.
SPACE	The approximate disc space currently used in sectors. This applies only to spooled output devicefiles.
RANK	The ranking of the file: its order in the system with respect to other files of the same output priority and *classname* or *ldev*. A time stamp activated by the FCLOSE command determines the file's rank. A "D" following "RANK" indicates a deferred file. This applies only to spooled devicefiles. A file can be deferred if its priority is less than or equal to system outfence or to the outfence of a specific device.
PRI	The output priority requested by user or as adjusted by the System Operator for spooled devicefiles only. A priority of 1 is lowest, and 13 is highest.
#C	Number of copies needed, for spooled devicefiles only.

The output may appear in two possible formats or in a combination of the two formats:

Type I:

```
DEV/CL  DFID  JOBNUM FNAME     STATE   FRM SPACE RANK PRI #C
45      #032  #S16   $STDLIST  OPENED
OUTFENCE=6
```

Type II:

```
19 FILES
    0 ACTIVE
    2 READY; INCLUDING 2 SPOOFLES, 2 DEFERRED
   17 OPENED; INCLUDING 1 SPOOFLE
    0 LOCKED; INCLUDING 0 SPOOFLES
    3 SPOOFLES:  1572 SECTORS
OUTFENCE = 6
OUTFENCE = 2 FOR LDEV 13          ** LDEV 13 has been assigned an outfence of 2 **
```

EXAMPLES

To display the total number of output devicefiles currently existing, the number of those that are spooled, and the states that they are in, enter:

```
:SHOWOUT STATUS
11 FILES:
    1 ACTIVE
    1 READY; INCL 1 SPOOFLES, 0 DEFERRED
    9 OPENED; INCL 1 SPOOFLES
    0 LOCKED; INCL 0 SPOOFLES
    3 SPOOFLES:  7212 SECTORS
OUTFENCE= 2
:
```

You can also request information about a specific output devicefile, device number or device class name of the device for which the file is destined in the :SHOWOUT command:

```
:SHOWOUT DEV=43

DEV/CL  DFID  JOBNUM FNAME     STATE   FRM SPACE RANK PRI #C
43      #089  #S37   $STDLIST  OPENED

OUTFENCE= 2
:
```

:SHOWQ

Displays process scheduling data and the contents of each subqueue.

SYNTAX

```
:SHOWQ
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, in BREAK. It may not be issued from a Program. It is Breakable. (aborts execution). It requires System Supervisor (OP) capability.

OPERATION

The process scheduling and subqueue information is displayed in three major columns: "DORMANT", "WAITING", and "RUNNING". RUNNING processes are these that currently require the CPU in order to continue, or that will require it in the immediate future. CPU time is automatically allocated to the highest-priority process that is ready to run.

The processes listed in "DORMANT" and "WAITING" columns are waiting on longer-term events. DORMANT processes must wait a shorter amount of time than WAITING processes, for example, until a code segment moves into memory. WAITING processes wait for a longer event, such as an I/O completion. They will contend for the CPU once those events have occurred.

On occasion, a process will appear in more than one column. This indicates that the process was being moved one column to the right when you executed of :SHOWQ.

Below is a sample of the output listing produced by the :SHOWQ command. The symbols that may appear in such a listing are explained in the remainder of the discussion.

```
 DORMANT              WAITING              RUNNING
Q  PIN   JOBNUM      Q  PIN   JOBNUM      Q  PIN   JOBNUM

L   1                                     C  M163  #S263
L   2                                     C  U215  #S256
L   3
L   4
D  U29   #J30
C  M37   #S234
C  M55   #S248
```

Each entry in the three columns displays the following information for a single process:

```
{ L }
{ C }  [ M ]  pin [  #Jnnn  ]
{ D }  [ U ]      [  #Snnn  ]
{ E }
```

Where:

L	a linearly scheduled process on the AS, BS, or master queue.
C	a circularly scheduled process on the CS queue.
D	a circularly scheduled process on the DS queue.
E	a circularly scheduled process on the ES queue.
M	a job main process.
U	a user process.
pin	the Process Identification Number for this process.
J nnn	the process executing in a batch job.
S nnn	the process executing from a session.

The Process Identification Number (PIN) may appear with or without an M or U label. Processes without an M or U label are system processes.

In addition, :SHOWQ prints the scheduling parameters currently in effect along with the minimum memory manager algorithm cycle value (*minclockcycle*).

```
CQ MINQUANTUM=0, MAXQUANTUM=300, BASEPRI=152, LIMITPRI=200
DQ MINQUANTUM=1000, MAXQUANTUM=1000, BASEPRI=202, LIMITPRI=238
EQ MINQUANTUM=1000, MAXQUANTUM=1000, BASEPRI=153, LIMITPRI=154
MINIMUM CLOCK CYCLE=1000
```

EXAMPLE

To display the queues of processes within MPE, enter:

```
:SHOWQ
```

The system will display a listing of DORMANT, WAITING, and RUNNING processes similar to the one explained in the preceding "OPERATION" section.

:SHOWTIME

Prints current time and date.

SYNTAX

```
:SHOWTIME
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable.

OPERATION

Prints current time and date, as indicated by system clock.

EXAMPLE

To display the time and date, enter:

```
:SHOWTIME
MON, JUL 24, 1985, 8:47 AM
```

=SHUTDOWN

Initiates a shutdown of MPE.

SYNTAX

```
=SHUTDOWN
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, in BREAK. It may not be issued from a Program. It is not Breakable. It can be issued only at the Console.

OPERATION

The =SHUTDOWN command performs an implicit =LOGOFF of all sessions, including the session logged at the system console. All system processes are stopped in an orderly fashion. This includes the completion of all pending system activity and any processing necessary to ensure that the integrity of all system tables and directories is maintained. Once these procedures are complete, "SHUT" is displayed on the console, the CPU halts, and console interrupt (A^C) is ineffective.

Device configuration changes that were made after the preceding load (:UP, :DOWN, :GIVE, :TAKE, :ACCEPT, :REFUSE and spooling commands) will not be retained. Configuration changes made during cold load are permanently recorded and retained until the next tape cold load. Newly assigned or released global Resource Identification Numbers (RINs) are permanently recorded.

All communication lines must be closed before issuing a =SHUTDOWN command. Otherwise, a manual halt of the system may be necessary. Note that data will be lost if a transmission is in progress when the halt is performed. Also, if DSN/DS lines were left open before issuing the =SHUTDOWN, lines to the remote system will remain open and any remote sessions will be effectively "hung". In this case, the remote system's Operator may need to issue :ABORTIO commands for the hung sessions and then abort the sessions themselves.

Spooled devices will stop operation immediately upon receiving a =SHUTDOWN command. A WARMSTART will keep spoolfiles so that they may be printed when the system is returned online.

EXAMPLE

To shut the system down, first issue a warning to all users to allow them time to log off, and then execute =SHUTDOWN as shown below:

```
:WARN @;SYSTEM WILL SHUTDOWN IN FIVE MINUTES. PLS LOG OFF.

A^C             ** Press A and CNTL simultaneously **

=SHUTDOWN
10:49/#S40/25/LOGOFF
10:49/20/ALL JOBS LOGGED-OFF
```

:SHUTQ

Closes the spool queue for the specified logical device or device class. This command is available on the G.00.00 and later releases of the operating system.

SYNTAX

```
:SHUTQ {ldev     }
       {devclass}
```

PARAMETERS

ldev	The logical device number of the spooled device.
devclass	The device class name of the spooled device.

USE

This command may be issued from a Session, Job, in BREAK, or from a program. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

:SHUTQ closes the spool queue for a logical device or device class configured in the system. The spooler process, however, does not need to be running for the device. If the spooler process is running, it is unaffected by shutting the queue.

This command also serves as an option to the :STARTSPOOL command, which is also documented in this section.

EXAMPLE

To shut the queue for device class LP, enter:

```
:SHUTQ LP
```

:SPEED

Changes the operating speed of the terminal.

SYNTAX

```
:SPEED newinspeed,newoutspeed
```

PARAMETERS

newinspeed The new input speed in characters-per-second (CPS). For Series 39/40//42/44/58/64/68 the speed must be 15, 60, 120, 240, 480, or 960. Series 39/40/42/44/48 also supports 10 CPS. If you are using the ATP driver, the speed 1020 is also supported. The *newinspeed* must be the same as newoutspeed.

newoutspeed The new output speed in characters-per-second. For Series 39/40/42/44/58/64/68 the speed must be 15, 60, 120, 480, or 960. Series 39/40/42/44/48 also supports 10 CPS. If you are using the ATP driver, the speed 1020 is also supported. The *newoutspeed* must be the same as *newinspeed*.

USE

This command may be issued from a Session, but not from a Job. It may be issued in BREAK or from a Program. It is not Breakable.

OPERATION

MPE automatically senses the input/output speed of a terminal when you logon at that terminal. If your terminal has speed adjustment controls, you can change the input and output speeds after logon with the :SPEED command. This command is not valid for terminals that operate at only one speed.

When the command is entered, MPE displays the following message at the old output speed:

```
CHANGE SPEED AND INPUT "MPE":
```

Manually change the speed control on the terminal and verify the new speed by entering:

```
MPE (RETURN)
```

If the characters "MPE" cannot be verified, the system assumes that the terminal is to continue at the old speed. (To continue, you must reset the terminal control to the old speed.) Note that on Hewlett-Packard terminals the "Baud Rate" is Characters Per Second (CPS) multiplied by 10. When you select the Baud Rate at which you choose to operate you must, therefore, divide the rate by 10, and enter that value with the :SPEED command.

You can also change the terminal speed programmatically by using the FCONTROL intrinsic. Refer to the MPE V Intrinsics Reference Manual (32033-90007).

Refer to Appendix A "TERMINALS SUPPORTED BY MPE V" for a list of the terminal types and the speeds that they support.

EXAMPLES

To change the input and output speeds to 240 CPS (2400 baud), enter:

```
:SPEED 240,240

CHANGE SPEED AND INPUT "MPE":
{
```

Manually change the speed and type in MPE. The "{" is a random character.

```
{MPE RETURN
```

ADDITIONAL DISCUSSION

Fundamental Communications Handbook (5957-4634)

:SPL

Compiles an SPL program. SPL is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:SPL [textfile][,[uslfile][,[listfile][,[masterfile][,newfile]]]]
     [;INFO=quotedstring]
```

PARAMETERS

textfile — Actual file designator of the input file from which the source program is read. This can be any ASCII input file. The formal file designator is SPLTEXT. Default is $STDIN.

uslfile — Actual file designator of the User Subprogram Library (USL) file to which the object code is written. This can be any binary output file created with a file code of USL or 1024.. Its formal file designator is SPLUSL. If the *uslfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS. If the *uslfile* parameter is entered, it indicates that the file was created in one of four ways:

- By using the MPE :SAVE command to save the default USL file created during a previous compilation.
- By building the USL with the MPE Segmenter -BUILDUSL command. Refer to the MPE Segmenter Reference Manual (30000-90011).
- By creating a new USL file with the MPE :BUILD command and specifying a file code of USL or 1024.
- By having the statement $CONTROL USLINIT in your program.

listfile — Actual file designator of the file to which the program listing is written. This can be any ASCII output file. The formal file designator is SPLLIST. Default is $STDLIST.

masterfile — Actual file designator of the master file with which *textfile* is merged to produce a composite source. This can be any ASCII input file. The formal file designator is SPLMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified.

newfile — Actual file designator of the file created by merging *textfile* and *masterfile*. This can be any ASCII output file. Formal designator is SPLNEW. Default is that no file is written.

NOTE

The formal file designators used in this command (SPLTEXT, SPLUSL, SPLLIST, SPLMAST, and SPLNEW) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the :FILE command.

quotedstring A sequence of ASCII characters bounded by a pair of single quotation marks (apostrophes) or by double quotations marks. If you want a quotation to appear within *quotedstring*, the quotation and its quotation marks must also be bounded by quotation marks. For example, to insert "and" into a *quotedstring*, it must appear as ""and"". Similarly, 'and' must appear as 'and'. The maximum length of the string, including delimiters, is 255 characters. Refer to "OPERATION".

For SPL to recognize *quotedstring*, a dollar sign ($) must follow the quotation marks at the beginning of the *quotedstring*. This feature is used to specify compiler options which will appear at the beginning of the source listing. For more information, refer to the :RUN command in this section.

USE

This command may be issued from a Session or a Job, but not in BREAK or from a Program. It is Breakable (suspends execution).

OPERATION

This command compiles an SPL program into a User Subprogram Library (USL) file on disc. If *textfile* is not specified, MPE expects the source program to be entered from your standard input device. If *listfile* is not specified, the program output will be sent to your standard list device.

EXAMPLES

The following example compiles an SPL program entered from your standard input device into an object program in the USL file $OLDPASS, and writes the listing to your standard list device:

```
:SPL
```

The next example compiles an SPL program contained into the disc file SOURCE and stores the object code into the USL file OBJECT. The program listing will be sent to the disc file LISTFL:

```
:SPL SOURCE,OBJECT,LISTFL
:SAVE OBJECT
```

ADDITIONAL DISCUSSION

Systems Programming Language Reference Manual (30000-90024)

:SPLGO

Compiles, prepares, and executes an SPL program. SPL is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:SPLGO [textfile][,[listfile][,[masterfile][,newfile]]]
       [;INFO=quotedstring]
```

PARAMETERS

textfile — Actual file designator of the input file from which the source program is read. This can be any ASCII input file. The formal file designator is SPLTEXT. Default is $STDIN.

listfile — Actual file designator of the file to which the program listing is written. This can be any ASCII output file. The formal file designator is SPLLIST. Default is $STDLIST.

masterfile — Actual file designator of the master file that is merged against *textfile* to produce a composite source. This can be any ASCII input file. Formal file designator is SPLMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified. If two files being merged have identical line numbers, the lines from *textfile* or from $STDIN will overwrite those in masterfile.

newfile — Actual file designator of the file produced by merging *textfile* and *masterfile*. This can be any ASCII output file. The formal file designator is SPLNEW. Default is that no file is written.

NOTE

The formal file designators used in this command (SPLTEXT, SPLLIST, SPLMAST, and SPLNEW) cannot be backreferenced as actual file designators in the command parameter list. For further information, refer to the "Implicit :FILE Commands For Subsystems" discussion of the :FILE command.

quotedstring — A sequence of ASCII characters bounded by a pair of single quotation marks (apostrophes) or by double quotation marks. If you want a quotation to appear within *quotedstring*, the quotation and its quotation marks must also be bounded by quotation marks. For example, to insert "and" into a *quotedstring*, it must appear as ""and"". Similarly, 'and' must appear as "and". The maximum length of the string, including delimiters, is 255 characters.

For SPL to recognize *quotedstring*, a dollar sign ($) must follow the quotation marks at the beginning of the *quotedstring*. This feature is used to specify compiler options which will appear in front of the source listing. For more information, refer to the :RUN command in this section.

USE

This command may be issued from a Session or a Job, but not in BREAK or from a Program. It is Breakable (suspends execution).

OPERATION

This command compiles, prepares, and executes an SPL program. If *textfile* is omitted, MPE expects input from your standard input device. This command creates a temporary User Subprogram Library (USL) file ($NEWPASS) that you cannot access, and a temporary program file that you can access under the name $OLDPASS. For more information, refer to "System Defined Files", Section I of this manual.

EXAMPLES

To compile, prepare, and execute an SPL program entered from your standard input device, and have the program listing sent to your standard list device, enter:

```
:SPLGO
```

To compile, prepare, and execute an SPL program read from the disc file SOURCE and send the resulting program listing to the disc file LISTFL, enter:

```
:SPLGO SOURCE,LISTFL
```

ADDITIONAL DISCUSSION

Systems Programming Language Reference Manual (30000-90024)
MPE Segmenter Reference Manual (30000-90011)

:SPLPREP

Compiles and prepares an SPL program. SPL is not part of the HP 3000 Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:SPLPREP [textfile][,[progfile][,[listfile][,[masterfile][,newfile]]]]
         [;INFO=quotedstring]
```

PARAMETERS

textfile Actual file designator of the input file from which source program is read. This can be any ASCII input file. Formal file designator is SPLTEXT. Default is $STDIN.

progfile Actual file designator of the program file to which the prepared program segments are written. You will get the best results by omitting the *progfile* parameter. When you omit *progfile*, the Segmenter creates the program file, which will reside in the temporary file domain as $OLDPASS. If you do create your own program file, you must do so in one of two ways:

- By using the MPE :BUILD command, specifying a file code of 1029 or PROG, and a *numextents* value of 1. This file is then used by the :PREP command.
- By specifying a nonexistent file in the *progfile* parameter, in which case a job/session temporary file of the correct size and type is created.

listfile Actual file designator of the file to which program listing is written. This can be any ASCII output file. Formal designator is SPLLIST. Default is $STDLIST.

masterfile Actual file designator of the master file that is merged against *textfile* to produce composite source. This can be any ASCII input file. The formal file designator is SPLMAST. Default is that the master file is not read; input is read from *textfile*, or from $STDIN if *textfile* is not specified. If two file being merged have identical line numbers, the lines form *textfile* or from $STDIN will overwrite those in *masterfile*.

newfile Actual file designator of the file produced by merging *textfile* and *masterfile*. This can be any ASCII output file. The formal file designator is SPLNEW. Default is that no file written.

NOTE

The formal file designators used in this command (SPLTEXT, SPLLIST, SPLMAST, and SPLNEW) cannot be backreferenced as actual file designators in the command parameter list. For further information refer to the "Implicit :FILE Commands For Subsystems" section of the :FILE command.

quotedstring

A sequence of ASCII characters bounded by a pair of single quotation marks (apostrophes) or by double quotation marks. If you want a quotation to appear within *quotedstring*, the quotation and its quotation marks must also be bounded by quotation marks. For example, to insert "and" into a *quotedstring*, it must appear as ""and"". Similarly, 'and' must appear as 'and'. The maximum length of the string, including delimiters, is 255 characters. Refer to "OPERATION".

For SPL to recognize *quotedstring*, a dollar sign ($) must follow the quotation marks at the beginning of the *quotedstring*. This feature is used to specify compiler options which will appear at the beginning of the source listing. For more information, refer to the :RUN command in this section.

USE

This command may be issued from a Session or a Job, but not in BREAK or from a Program. It is Breakable (suspends execution).

OPERATION

Compiles and prepares an SPL program into a program file on disc. If *textfile* is not specified, MPE expects you to enter your source program from your standard input device. If you do not specify *listfile*, your program output will be sent to your standard list device.

The User Subprogram Library (USL) file created during compilation, $OLDPASS, is a temporary file passed directly to the Segmenter. It can be accessed only if you do not use the default for *progfile*. This is because the Segmenter also uses $OLDPASS to store the prepared program segments, overwriting the USL file of the same name.

EXAMPLES

To compile and prepare an SPL program entered from your standard input device and send the output to your standard list device, enter:

```
:SPLPREP
```

The following example compiles and prepares an SPL source program from the disc file SFILE into the program file MYPROG. The program listing will be sent to your standard list device:

```
:SPLPREP SFILE,MYPROG
```

In the next example, the first first positional parameter is omitted. This indicates to MPE that you intend to enter the source text from your standard input device. The object code will be stored in the default USL file $OLDPASS, and the prepared program segments will be stored in FILEZ. OLDPASS is then saved in the permanent file domain under the new name NUSL.

```
:SPLPREP,FILEZ
:SAVE $OLDPASS, NUSL
```

ADDITIONAL DISCUSSION

System Programming Language Reference Manual (30000-90024)
MPE Segmenter Reference Manual (30000-90011)

:STARTCACHE

Enables caching on a single disc. Disc caching software is not included with the Fundamental Operating Software and must be purchased separately.

SYNTAX

```
:STARTCACHE ldev
```

PARAMETERS

ldev	The logical device number of the disc to be used for caching.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It requires System Supervisor (OP) or System Manager (SM) capability.

OPERATION

The System Supervisor uses this command to enable caching on a single disc. When the system is first started, caching is disabled, by default. Each disc to be cached requires a separate invocation of the command.

EXAMPLE

To enable caching on *ldev* 3, enter:

```
:STARTCACHE 3
CACHING STARTED ON LDEV 3.
```

NOTE

If caching is not installed on your system, an error message will be displayed:

```
CIERR 4417 DISC CACHING NOT ENABLED ON
THIS SYSTEM
```

:STARTSESS

Creates a session on the specified device, provided that the command's user has Programmatic Sessions (PS) capability. (G.01.00 or later releases)

SYNTAX

```
:STARTSESS ldev[sessionname,]user[/userpass].acct[/acctpass]
      [,group[/grouppass]]
   [;TERM={termtype}]
          {termname}

   [;TIME=cpusecs ]

        {BS}
   [;PRI={CS}]
        {DS}
        {ES}

   [{;INPRI=inputpriority}]
   [{;HIPRI               }]

   [;NOWAIT]
```

PARAMETERS

ldev

The logical device number of the target terminal. This terminal must be a real physical device and cannot be a virtual terminal or a DS pseudo terminal. The terminal must be configured as TYPE 16 and as SUBTYPE 0 or 4.

NOTE

All required passwords must be specified in the command invocation; STARTSESS will not prompt the caller or the *ldev* for passwords. Omitting a missing password where one is required will cause the command to fail.

sessionname

Arbitrary name used in conjunction with the *user* and *acct* parameters to form a fully qualified session identity. The name may contain from one to eight alphanumeric characters, beginning with an alphabetic character. Default is that no session name is assigned.

user

User name, established by the Account Manager, that allows you to logon under this account. The name may contain from one to eight alphanumeric characters, beginning with an alphabetic character.

userpass

User password, optionally assigned by the Account Manager. The password may contain from one to eight alphanumeric characters, beginning with an alphabetic character.

acct

Account name established by the System Manager. The name may contain from one to eight alphanumeric characters, beginning with an alphabetic character. A period (.) must precede the *acct* parameter.

acctpass

Account password, optionally assigned by the System Manager. The password may contain from one to eight alphanumeric characters, beginning with an alphabetic character. A slash (/) must precede the *acctpass* parameter.

group

Group name to be used for the local file domain and the CPU time charges established by the Account Manager. The name may contain from one to eight alphanumeric characters, beginning with an alphabetic character. Default is the specified users home group if you are assigned one by the Account Manager. The parameter is required if a home group is not assigned.

grouppass

The *grouppass* parameter is not needed when the user logs on under the user's home group, even if a password has been established. The *grouppass* is needed when the user logs on under any other group for which a password has been established. A slash (/) must precede the parameter *grouppass*.

termtype or *termname*

Determines terminal-type characteristics. The value of the *termtype* parameter determines the type of terminal used for input. MPE uses this parameter to determine device-dependent characteristics such as delay factors for carriage returns. The value must be a number from 0 to 22. The default value for *termtype* is assigned by the System Supervisor during system configuration. This parameter is required to ensure correct listings if your terminal is not the default *termtype*.

The *termname* parameter is the name of the file containing the desired terminal-type characteristics. The file must not have a lockword or reside on a Private Volume.

Users of the Workstation Configurator are allowed to create terminal-type files. The proper and efficient operation of a specific device by a user-created terminal-type is the responsibility of the user. The Workstation Configurator utility allows the user to specify the following characteristics of the terminal: data flow control, block mode, read trigger, special characteristics, echo, line feed, parity, and printer control.

If group and/or account names are omitted, the proposed logon group and/or account name is substituted. For further information on the terminal-type characteristics, refer to the Communications Handbook (30000-90105) and the Workstation Configurator Reference Manual (30000-90001). Refer also to Appendix A, "Terminals Supported by MPE", in Section III of this manual.

cpusecs

Maximum CPU time that a session must use, entered in seconds. When the limit is reached, the session is aborted. It must be a value from 1 to 32767, provided that it does not exceed any limit imposed by the System or Account Manager. To specify no limit, enter a question mark (?), or UNLIM, or omit the parameter. Default is no limit. BS, CS, DS, or ES The execution priority queue that the Command Interpreter uses for your session, and also the default priority for all programs executed within the session. BS is highest priority, ES is lowest. If you specify a priority that exceeds the highest permitted for your account or username by the system, MPE assigns the highest priority possible below BS. DS and ES are intended primarily for batch jobs; their use for sessions is generally discouraged. Care should be used in assigning the BS queue because processes in this priority class must lock out other processes. For information on the guidelines for these priority queues, refer to the :TUNE command in this section. Default is CS.

inputpriority or HIPRI

Determines the input priority of the job or session. The *inputpriority* option is the relative input priority used in checking against access restrictions imposed by the job fence. The *inputpriority* option takes effect at logon time and must be from 1 (lowest priority) to 13 (highest priority). If you supply a value less than or equal to current job fence set by System Operator, session is denied access. Default is 8.

The HIPRI option is used for two different purposes when logging on. It can be used to override the system job fence or it can be used to override the session limit. When using the HIPRI option to override the job fence, the system will first check to see if you have System Manager (SM) or System Operator (OP) capability. The user who has either of these capabilities, will be logged on and the INPRI will default to the system's job fence and execution limit. If you do not have either of these capabilities, the system will attempt to log you on using INPRI=13 and will succeed if the job fence is 12 or less, and the session limit is not exceeded. In attempting to override the session limit (to logon after the maximum number of sessions set by the operator has been reached), you can specify HIPRI, but in this case you must have either SM or OP capability. The system will not override the the session limit automatically. Use of the HIPRI option without SM or OP capability causes the following warning to be displayed:

```
MUST HAVE 'SM' OR 'OP' CAP. TO SPECIFY HIPRI,
 MAXIMUM INPRI OF 13 IS USED (CIWARN 1460)
```

NOWAIT

Request that the session starts executing immediately without waiting for a RETURN on the terminal. If this parameter is specified and the target terminal is the System Console, System Manager (SM) capability is required.

USE

This command is available from a session, job, in BREAK, or from a Program. It is not Beakable. It requires Programmatic Sessions (PS) capability.

OPERATION

The MPE user employs this command to create a session at any terminal on the system. The effect will be the same as if a user had logged on at the target terminal.

NOTE

1. The target terminal must be available; no other user must be logged on, and the terminal must be turned on.

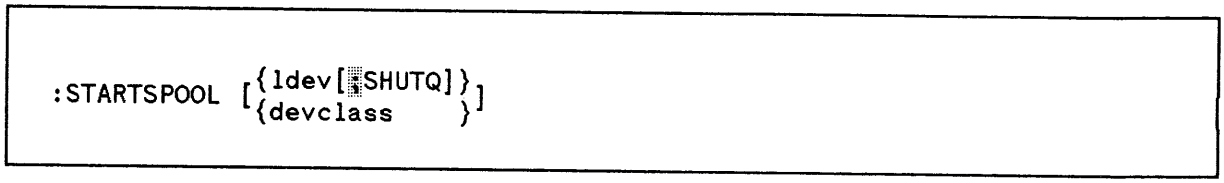
2. No speed sensing will be done for the target terminal, so it must be set at the configured baud rate.

3. When the session is created, nothing will be printed to the target terminal until its carriage return key is pressed unless NOWAIT was specified.

:STARTSPOOL

Initiates the spooler process for a device.

Syntax



PARAMETERS

ldev	The logical device number of a spooled device. When the spooler gains control of the specified device, it controls spooling to it as well as to all device classes which reference the device.
devclass	The device class specified in the I/O configuration. Only this device class becomes spooled; it will not affect other device classes or any devices in the class.
SHUTQ	The spooler will print files waiting in the queue for device *ldev*, but prevents the creation of new spoolfiles. The SHUTQ parameter is valid for *ldev* only.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It may be issued only at the Console.*

* Unless distributed to users with the :ALLOW or ASSOCIATE commands.

OPERATION

To start the spooling process for a specified device, and for any and all device classes associated with it, issue the :STARTSPOOL command with the *ldev* parameter. When *devclass* is used with :STARTSPOOL, only the specified device class will be controlled by the spooler. The logical device itself will not be controlled, unless a :STARTSPOOL has also been issued for the corresponding *ldev*.

The HP 2680A and HP 2688 Laser Printer operates only as a spooled device. If spooling is enabled only on the Laser Printer's *ldev*, and spooling stops as a result of an I/O error, no new spoolfiles will be created. To avoid this, issue the :STARTSPOOL command twice for an HP 2680 Laser Printer: once for the *devclass* associated with the printer, and a second time for the *ldev* assigned to it.

NOTE

If the *ldev* referenced in a :STARTSPOOL command is an HP 2894 card reader/punch, the following message will be generated on the system console:

```
IS THIS AN INPUT OR OUTPUT SPOOLER
(IN/OUT)?
```

You must then enter the appropriate reply: "IN", if cards are to be read; "OUT", if cards are to be punched.

EXAMPLES

To start spooling all output to logical device 6 and all device classes that reference logical device 6, enter:

```
:STARTSPOOL 6
```

To start spooling all output to device class LP, enter:

```
:STARTSPOOL LP
```

To start spooling on logical device 6 while preventing the creation of any new spoolfiles, enter:

```
:STARTSPOOL 6;SHUTQ
```

ADDITIONAL DISCUSSION

Refer to the discussion of "SPOOLING" in the MPE System Operation and Resource Reference Manual (32033-90005).

:STOPCACHE

Disables caching on a single disc. Disc caching software is not included in the Fundamental Operating Software, and must be purchased separately.

SYNTAX

```
:STOPCACHE ldev
```

PARAMETERS

ldev	The logical device number of the disc for which you intend to disable caching.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It requires System Supervisor (OP) or System Manager (SM) capability.

OPERATION

The System Supervisor uses this command to disable caching on a single disc. If your system uses more than one disc for caching, each *ldev* must be disabled with a separate invocation of :STOPCACHE.

EXAMPLE

To disable caching on *ldev* 3, enter:

```
:STOPCACHE 3
```

Terminates spooling to a specified device or device class.

Syntax

```
:STOPSPOOL [{ldev[;OPENQ]}]
           {devclass    }
```

PARAMETERS

ldev — The logical device number of a spooled device. The spooler process gives up ownership of the spooled device. If the OPENQ parameter is omitted, the device becomes available only for non-spooled I/O. When a logical device is assigned to more than one device class, spooling may be restarted for a specific device class by issuing an explicit :STARTSPOOL request for that class.

devclass — The device class specified in system I/O configuration. Subsequent I/O directed to this device class will not take place to/from a spoolfile. I/O will go directly to/from a logical device if one is available within the device class. If none is available, the program will be unable to open the file.

OPENQ — May be specified with the *ldev* parameter only. The spooler process will leave the queue OPEN , or will OPEN the queue if previously shut. Default is SHUTQ .

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

You may use the :STOPSPOOL command to stop spooling for a single logical device, or for all devices assigned a common device class. Using the *devclass* parameter in a :STOPSPOOL command will shut the queue for that device class. When you specify *ldev*, however, you may shut the spooling queue or leave it open. Default is SHUTQ.

When spooling is stopped for a particular device without using OPENQ, that device is conversationally referred to as "hot" (a "hot printer", for example). In this case, one job has exclusive access to that device preventing other programs from accessing it.

EXAMPLES

To terminate spooling to logical device number 6 and cause the spooler process to relinquish control of that device, enter:

```
:STOPSPOOL 6
```

Spooling will also terminate for any device class that references this device unless a :STARTSPOOL has been issued for a specific device class.

To stop directing output for device class LP to a spoolfile (provided a :STOPSPOOL 6 has also been issued), enter:

```
:STOPSPOOL LP
```

To terminate spooling on device 6 and leave the queue open, enter:

```
:STOPSPOOL 6;OPENQ
```

ADDITIONAL DISCUSSION

Refer to the discussion of "SPOOLING" in the MPE V System Operation and Resource Management Reference Manual (32002- 90005). manual.

Copies disc files onto magnetic tape or serial disc in a manner that is recoverable by :RESTORE . Two parameters are supported only on the G.00.00 and later releases of MPE. These are noted by "(G.00.00) and later".

SYNTAX

```
:STORE [{fileset[-fileset][,fileset[-fileset]][,...]}][;{[;*storefile]}]
       [{!indirectfile                            }]  {[;storefile ]}

       [;SHOW[=[=showparm[,showparm[,...]]]]

       [;FILES=maxfiles]

       [;DATE[{<=accdate}]
              {>=moddate}

       [;ONERR[OR]=[{QUIT}]
                   {REDO}
       [;PURGE]

       [;PROGRESS[=#minutes]
```

PARAMETERS

fileset

Specifies the set or sets of files to be stored. You must have READ and LOCK access to any files which you wish to store. This positional parameter has the form:

filename[.*groupname*[.*acctname*]]

If you specify nothing for the group or account, MPE assumes the logon group and account. Default is @ (stores all files in logon group). You must use the characters @, #, and ? can be used as wildcard characters in the *fileset* parameter. The wildcard characters count toward the eight character limit for the user, group, account, and filenames. These wildcard characters have the following meanings:

@	Specifies zero or more alphanumeric characters. When used by itself, @ denotes "all members of the set".
#	Specifies one numeric character.
?	Specifies one alphanumeric character.

The characters can be used as follows:

n@	Stores all items starting with the character "n".
@n	Stores all items ending with the character "n".
n@x	Stores all items starting with the character "n" and ending with the character "x".
n#######	Stores all items starting with the character "n" followed by seven digits.
?n@	Stores all items whose second character is "n".
n?	Stores all two-character items starting with the character "n".
?n	Stores all two-character items ending with the character "n".

-fileset Name of a file set or subfile set to be excluded from the store. You cannot nest the *-fileset* parameter. There can only at most one *-fileset* (excluded subset) for each *fileset* (stored set of files). A negative sign (-) must precede this parameter.

!indirectfile An ASCII file that contains the desired parameters of the :STORE command. Each line of the *indirectfile* may contain *filesets* and keywords. An exclamation point (!) must precede this parameter.

*storefile The name of the destination tape or serial disc file on which the stored files are to be written. It can be any magnetic tape or serial disc from the output set. This parameter is required for the E.01.00 release of MPE. This file may be named in a :FILE command and backreferenced by using an asterisk (*).

SHOW Produces a list all files stored. For each file stored, the following information is printed on the user's terminal (default): filename, group name, account name, logical device number, file size (in sectors) and file code. As each file is stored, a message is printed on the user's terminal. Error messages are displayed in the same way. Thus, if a particular file is not stored due to its being "in use", disc error, etc, you may elect to abort the store process. The listing is sent to $STDLIST (formal file designator is SYSLIST) unless a :FILE command is entered to send the listing to some other device. If the SHOW parameter is not specified, only the names of the files not stored, and the number of files stored and not stored, are listed.

showparm May be SHORT, LONG, DATES, SECURITY, or OFFLINE. You cannot specify both LONG and SHORT, but you may use more than one *showparm* if they are separated by commas. If neither SHORT nor LONG is selected, the default is SHORT if $STDLIST is less than 99 characters. The default is LONG otherwise.

SHORT	Causes SHOW to display the filename, group name, account name, logical device number, sector address, reel number, and file size in sectors of all files stored.

LONG — Causes SHOW to display the same information as SHORT, plus: record size, file type, end-of-file, file record limit, blocking factor, extents allocated, and maximum extents of all files stored.

DATES — Causes SHOW to display the creation date, last access date, and last modification date of all files stored.

SECURITY — Causes SHOW to display the creator and file access matrix of all files stored.

OFFLINE — Causes the SHOW output to be sent to the system line printer as well as to on the user's terminal. The line printer output can be redirected to another output device by using a :FILE command. (The formal file designator is OFFLINE.)

maxfiles — Maximum number of files that may be stored. If more than *maxfiles* are found, :STORE will print an error message and quit. Absolute maximum allowed is 32,767. Default is 4000.

<*=accdate* or >*=moddate* — Stores only those files that have not been accessed or that have been modified since the date specified. The *accdate* parameter stores only files not accessed since date specified. The *moddate* stores only those files that were modified on or after date specified. The date is expressed in the numeric form *mm/dd/yy*. When no date is specified, files specified by the *filesets* will be stored.

ONERR[OR] — Causes STORE to quit or continue when a tape error occurs. You may shorten ONERRPR to ONERR when specifying this parameter.

QUIT — Terminates the :STORE program when a tape error is encountered.

REDO — Causes :STORE to close the current tape reel, rewind it, and mark it as bad, whenever a tape error is encountered. A message is sent tp the Console advising that the current tape contains an error. After the tape has rewound, a request for a new reel is sent to the Console. Writing to the new reel begins at that point in the *fileset* list where the rejected (bad) reel began. Thus, the new (good) reel will contain precisely the files that would have been written to the previous (bad) tape. REDO cannot be specified for labeled tapes.

Default: — QUIT for labeled tapes.

REDO for unlabeled tapes.

USE

This command may be issued from a Session or Job, or a Program, but not in BREAK. It may be issued from a Program. It is Breakable. The user must have System Manager (SM), System Supervisor (OP), or Priviledged Mode (PM) capability to execute this command for priviledged files.

OPERATION

This command stores one or more disc files onto magnetic tape or serial disc. The :STORE command will store only those files whose home volume sets are mounted.

Users with System Manager (SM) or System Supervisor (OP) capability can store any userfile in the system. Users with Account Manager (AM) capability can store any file in their account, but those files with negative file codes unless the users have Privileged Mode (PM) capability.

Before entering a :STORE command, you may identify *storefile* as a magnetic tape or serial disc file with a :FILE command (referred to as a file equation).

The :STORE command may be invoked with the :RUN command. The "INFO=" parameter of the :RUN command can be used to specify the store option, *filesets*, and keywords. If no "INFO=" parameters are specified, the " <-- " prompt will appear. Acceptable responses are: a complete :STORE command, a complete :RESTORE command, an MPE command preceded by a colon, or :EXIT.

If [BREAK] is pressed during a store operation, the operation continues while you interact with the command interpreter. Both :ABORT and :RESUME are usable within BREAK.

EXAMPLE

To copy all files from the group GP4X in your logon account to a tape file named T, enter:

```
:FILE T;DEV=TAPE
:STORE @.GP4X;*T;SHOW
```

The System Console will receive a request to mount the tape identified as T. A listing of the files stored will appear on your standard list device as each is stored. If a file is in use by the system, the filename will be displayed, along with the message "BUSY". The following is a sample listing of the files stored:

```
FILENAME.GROUP    .ACCOUNT    LDN   ADDRESS REEL  SECTORS CODE
LABEL    .MAC     .TECHPUBS     3%01146363    1      257 FIG
MAIL     .MAC     .TECHPUBS     1%00120467    1       40
MANPLAN4.MAC      .TECHPUBS     2%01546471    1      844 RASTR
MANPLAN5.MAC      .TECHPUBS     3%01102561    1      544 RASTR
FILES STORED:          4
:
```

To store all files that were modified on or after 1/30/84, enter:

```
:STORE @.@.@;*T;DATE>=1/30/84;ONERROR=QUIT
```

Note that no error recovery has been specified in the example above.

To store all files that are not in any PUB group and show the information on the standard list device, enter:

```
:STORE @.@.@ - @.PUB.@;*T;SHOW
```

To store file ABC and show all possible information on both the line printer and $STDLIST, enter:

```
:STORE ABC.GROUPX.TECHPUBS;*T;SHOW=LONG,DATES,SECURITY,OFFLINE


FILENAME.GROUP   .ACCOUNT   LDN  ADDRESS REEL  SECTORS CODE
          SIZE TYP      EOF    LIMIT R/B #X/MX  CREATED ACCESSED MODIFIED

ABC      .GROUPX TECHPUBS           3%00075371    1        4
            88B FA         4        4   4  1/01  3/09/84  3/21/84  3/09/84
          MAC  (R:ANY; A:ANY; W:ANY; L:ANY; X:ANY) RELEASED

FILES STORED:          1
FILES NOT STORED:      0
:
```

To store using the specifications in the indirect file INDSTORE:

```
:STORE !INDSTORE;*T
```

where the file INDSTORE might contain:

```
@.PUB.SYS,@.@.TECHPUBS;FILES=32767
@.@.SUPPORT;SHOW=LONG,DATES
```

To store files using the :RUN command, enter:

```
:RUN STORE.PUB.SYS

<--STORE ABC
STORE/RESTORE, VERSION 2    (C) 1981  HEWLETT-PACKARD CO.
MON, APR 16, 1984,  6:27 AM


FILES STORED:          1
FILES NOT STORED:      0
<--
```

You may use the :RUN command with the :FILE command:

```
:FILE OFFLINE;DEV=PP;CCTL
:RUN STORE.PUB.SYS;INFO= "STORE @.@.@;*T;SHOW"
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)

:STREAM

Spools batch jobs or data from a session or job. The optional time-related parameters of the :STREAM command may be used to schedule jobs.

SYNTAX

```
:STREAM filename [,char]

    [;AT=timespec]

          {day-of-week}
   {[;DAY= {day-of-month}[;AT=timespec]]}
          {days-until-month}

    [;DATE= mm/dd/yy[;AT=timespec]]

   {[;IN= [days][,hours[,minutes]]]}
```

PARAMETERS

filename — The EDITOR file containing the commands of the job.

The first character of the first record is assumed to be the replacement character for the expected colon (:) that identifies MPE commands. The user must have READ or EXECUTE access.

char — Character used in place of colon (:) to identify MPE commands within the input file. When the input file is entered on a device configured to accept jobs, sessions, or data input via the :DATA command, this character can be any ASCII special (non alphanumeric) character except a colon. Default is an exclamation point (!).

AT — Absolute time specification.

timespec — Time specification. This is the absolute time of day in the format HH:MM where HH is the hour of the day (0<=HH<=23) and MM is the minuter of the hour (0<=MM<=59). If DAY or DATE are not specified, *timespec* is the time of the current day. If *timespec* is earlier than the current time, the job logs on immediately with an explanatory message.

DAY — Absolute day specification.

day-of-week — Day-of-Week. Allowable values are:

SUN[DAY]
MON[DAY]
TUE[SDAY]
WED[NESDAY]
THU[RSDAY]
FRI[DAY]
SAT[URDAY]

day-of-month Day-of-Month. The integers 1 through 31. It indicates the calendar day of the month. If *dom* is greater than or equal to the current day-of-month, the current month is indicated. If *dom* is less than the current day-of-month, the next month is indicated. An error message is generated if the day-of-month does not correspond to the month (e.g. 31 is entered for February). If *dom* is omitted, the current date is used.

days-until-monthend Day-from-End-of-Month. The negative integers -31 through -1. It indicates the calendar day from the end of the specified month on which the job will run. For example, a -1 value represents the last day of the month. If the specified day from the end of the month indicates a day earlier than the current day, the next month is assumed. For example, if today is the seventh day from the end of the month and a -8 value is entered, the job will be scheduled for the eighth day from the end of the next month.

DATE Absolute date specification.

cusdate Date, specified in the format MM/DD/YY, where MM is the month (1<=MM<=12), DD is the day (1<=DD<=31), and YY is the year. If omitted, the current date is used.

IN Relative date or time specification.

days Days. A positive integer indicating the offset in days from the current date.

hours Hours. A positive integer (0<*hours*<=23) indicating the offset in hours from the current time. If omitted, zero is used.

minutes Minutes. A positive integer (0<=*minutes*<=59) indicating the offset in minutes from the current time. If omitted, zero is used.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is Breakable (aborts execution and any partially streamed job).

OPERATION

With this command you can initiate jobs while in an interactive session by constructing your job from your terminal, or by reading records from a card, disc, or tape file. When the job is read, MPE spools it onto a disc file, assigns it a job number, and processes it independently as an entity completely separate from your session. In the meantime, MPE allows you to continue with your session.

You can initiate jobs in this way only if the System Operator, or a user who has been given operator capabilities, has enabled the MPE Stream Facility by entering the console command :STREAMS. The console command :STREAMS also specifies a streaming device, which to MPE appears to be the source of your job input regardless of the device you actually use for this input. As a result, the listing device that corresponds to the streaming device (and not your terminal) will display the job number assigned by MPE and the listing generated by the job. To stream data to disc the input device used must be configured to accept the :DATA command.

When you enter :STREAM without an input file (that is, with the terminal as the default input device), during a session or a job, MPE prompts you for input by displaying a greater than (>) character. When you enter :STREAM for a device other than your terminal, MPE does not print the prompt character.

How to Stream Jobs

Begin each job in the input file with the !JOB command and terminate it with the !EOJ command. Begin each data file with the !DATA command and terminate this file with the !EOD command. Begin all commands with an appropriate substitute (other than colon) *character*, as in !JOB or #DATA. When the input file is spooled to disc, MPE replaces the substitute command identifier with a colon, so that the data files are properly interpreted when executed.

After reading the !EOJ command that terminates the job, MPE assigns each job a unique job number (JobID). MPE also assigns each job a preset priority, unless you specify otherwise in the :JOB command, and processes the job independently of the initiating job. Regardless of which device you use to submit the input file, all jobs in that file are treated as though they originated on the unique streaming device designated by the System Operator (with the :STREAMS command). The listing for each spooled job and the job number are written to the standard list device that corresponds to the streaming device. You may, however, use the "OUTCLASS=" parameter of the :JOB command to direct the listing to another device.

How To Time Schedule Jobs

You may specify the time a job is to enter the WAIT state in absolute or relative time.

Absolute	The user supplies an exact time for the job using the AT parameter with or without the DAY or DATE parameter.
Relative	The user specifies a time offset from the current time using the IN parameter.

If the time specified is the same as the current time, the specified job will logon immediately. If the time specified is earlier than the current time, a warning message is generated. Otherwise, any time in the current century can be specified.

If no errors are detected, a JobID is displayed on the user's screen. If more than one job is included in the *inputfile*, each job is assigned a unique JobID and all of the jobs will be scheduled at the same time. If the *inputfile* also contains :DATA files, the files will be immediately added to the Input Device Directory (IDD). There is a limit of 64 jobs and :DATA files in any one *inputfile*.

When a job is scheduled for a future time, it enters the SCHED state. When the specified time is reached, the job enters the WAIT state and will be executed when system variables allow.

Terminating Streamed Jobs

To terminate interactive job input, enter a colon (:). In response, MPE ceases prompting for batch job input and instead prompts you for another MPE command:

```
>:          ** Denotes end of batch job input.      **
:           ** MPE prompts for next command.       **
```

Pressing [BREAK] aborts the :STREAM command and any job currently being entered through the command. Incompletely spooled disc space is returned to the system.

If you make an error while entering the MPE :JOB command, you will receive an error message on your job listing device. The System Operator, however, receives no indication of the job or the error.

Terminating Time Scheduled Job

Jobs that have been scheduled for :STREAM execution can be terminated with the :ABORTJOB command. Refer to the MPE System Operation and Resource Management Reference Manual (32033-90005), for information on using the :ABORTJOB command to terminate time scheduled jobs.

Special Considerations

In order to :STREAM a file, you must have READ and LOCK access or EXECUTE access to that file. However, READ and LOCK access would allow general users to obtain security information within the file, such as passwords and lockwords. To allow general users to :STREAM the file without giving them access to secure information, you may allow EXECUTE access only. For more information on file security, refer the MPE V System Operation and Resource Management Reference Manual (32033-90005).

NOTE

Scheduled jobs will survive a WARMSTART. Any other system starts will cause scheduled jobs to be deleted. If the system is brought down for any reason, execute a :SHOWJOB command to show the scheduled jobs. Then reschedule the jobs when the system is brought back up.

A scheduled job uses an entry in the JMAT table. Because of the limited recoverability of scheduled jobs, it is recommended that jobs be scheduled no more than a few days in advance.

If a user specifies a day or date for a job, but does not specify a time, the job will not enter the WAIT state at midnight on the specified day. Instead, it will use the time that the :STREAM is executed, and will enter the WAIT state at that time on the specified day.

EXAMPLES

To stream a job from a disc file, you must name the input file in the :STREAM command:

```
:STREAM ABC
```

If you use a character other than an exclamation point (!) as the substitute command identifier in your job input, you must identify that character in the :STREAM command. Because you enter this character as the second positional parameter in this command, you must always precede it with a delimiting comma, even when you omit the input filename (the first parameter):

```
:STREAM,*                    ** Asterisk used as substitute command identifier. **
>*JOB MAC.TECHPUBS
>*FORTGO MYPROG
>*EOJ
#J74
>:
:
```

If your job input file contains subsystem commands, such as commands directed to the editor, do not enter any command identifier *character* at the beginning of these commands. For instance, when using the Editor, enter the subsystem commands as follows:

```
:STREAM EXAMPLE                 ** The job input file is EXAMPLE.                **
!JOB WXYZ,WRITER.TECHPUBS,MAC   ** Initiates the job WXYZ.                       **
!EDITOR                         ** Invokes the Editor.                          **
TEXT ABC                        ** Subsystem commands without (!) as an identifier. **
LIST ALL,OFFLINE
EXIT
!EOJ                            ** Terminates job.                             **
#J87                            ** Job number supplied by MPE.                 **
:
```

If you want the job listing to appear on a device other than the standard listing device associated with the streaming device, you can specify this other device in the MPE :JOB command. Enter:

```
:STREAM
>!JOB MAC.TECHPUBS;OUTCLASS=12
```

The following section contains additional examples of :STREAM command usage. For these examples, assume that the current date and time are Monday, April 16, 1984, 12:00 p.m. Also assume *jobfile* contains a valid :STREAM job.

:STREAM JOBFILE	JOBFILE will be introduced immediately.
:STREAM JOBFILE;AT=8:00	JOBFILE will be introduced at 8:00 a.m., Tuesday, April 17.
:STREAM JOBFILE;AT=20:00	JOBFILE will be introduced at 8:00 p.m., Monday, April 16.
:STREAM JOBFILE;IN=,8	JOBFILE will be introduced in eight hours, at 8:00 p.m., Monday, April 16.
:STREAM JOBFILE;IN=1,8	JOBFILE will be introduced in one day plus eight hours, at 8:00 p.m., Tuesday, April 17.

:STREAM JOBFILE;DAY=MON;AT=8:00	Since the time specified (8:00 a.m.) is earlier than the current time, JOBFILE will be introduced at 8:00 a.m., Monday, April 23.
:STREAM JOBFILE;DAY=MONDAY;AT=20:00	Since the time specified (8:00 p.m.) is later than the current time, JOBFILE will be introduced at 8:00 p.m., Monday, April 16.
:STREAM JOBFILE;DAY=17;AT=20:00	Since the day of the month (17) is later than the current day of the month (16), the current month is assumed. JOBFILE will be introduced on Tuesday, April 17, at 8:00 p.m.
:STREAM JOBFILE;DAY=5	Since the day of the month (5) is earlier than the current day (16), the next month is assumed. Since no time was specified, JOBFILE will be introduced on Saturday, May 5, at 12:00 p.m.
:STREAM JOBFILE;DAY=31	Since there is no April 31, the next month is assumed. Since there is a May 31, this is a legal command. JOBFILE will be introduced on Thursday, May 31, at 12:00 p.m. If there was no May 31, this would result in an error.
:STREAM JOBFILE;DAY-2	The "-2" means the second to last day of the month, and since no time was specified, the current time is used. JOBFILE will be introduced on Sunday, April 29, at 12:00 p.m.
:STREAM JOBFILE;DAY-20	The "-20" means the twentieth day from the end of the month. If one assumes the current month, that implies April 11, but April 11 is earlier than the current day; therefore, the next month is assumed. JOBFILE will be introduced on Saturday, May 12, at 12:00 p.m.
:STREAM JOBFILE;DATE=4/16/84;AT=8:00	Since the specified time is earlier than the current time, this command is not legal and will result in an error.
:STREAM JOBFILE;DATE=4/16/84;AT=20:00	The specified time is later than the current time, so this command is legal. JOBFILE will be introduced on Monday, April 16, at 8:00 p.m.

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)

:STREAMS

Enables or disables the streams device: it permits users to submit job/data streams, or prevents them from doing so.

SYNTAX

```
:STREAMS {ldev}
         {OFF }
```

PARAMETERS

ldev	The logical device number of the STREAMS device. This device must also have an output device number or class that references logical devices of type 32. Any input device, except the system console or terminals, may be used, providing that it was configured as job-accepting in the SYSDUMP dialogue.
OFF	Disables the STREAMS facility.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

The Operator executes this command after a startup to enable the STREAMS facility. The :STREAMS device must be enabled each time the system is brought back online in order to to allow users to stream jobs. (Streamed jobs are processed separately by MPE, allowing users to continue with other work at their terminal. If the streamed job is submitted on a tape drive rather than from a terminal, MPE will process it without requiring the user's attention.) Any attempt to stream a job when the :STREAMS facility is disabled generates the following message:

```
STREAM FACILITY NOT ENABLED: SEE OPERATOR.  (CIERR 82)
```

The device normally configured as the streams device is *ldev* 10. However, *ldev* 10 may not correspond to an actual device, such as a tape drive or card reader, physically connected to the computer. If this is the case, then the STREAMS device is considered a "psuedo-device". Regardless of whether the device physically exists or not, it must be entered into the I/O Configuration table as a legitimate logical device. It must be assigned the device class JOBTAPE.

EXAMPLES

To enable jobs and data streams on logical device number 10, enter:

```
:STREAMS 10
```

To disable data streams, enter:

```
:STREAMS OFF
```

ADDITIONAL DISCUSSION

Refer to the discussion of "JOB" and to the :STREAM command in this manual.

:SUSPENDSPOOL

Suspends output to a spooled device.

SYNTAX

```
:SUSPENDSPOOL ldev[;FINISH]
```

PARAMETERS

ldev	The logical device number of a spooled device.
FINISH	Directs the device to complete the currently active spoolfile and then stop.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

When the spooler process is suspended, the message "SP# *ldev* SPOOLER SUSPENDED" is displayed on the Console. You may also determine the spooler's status by entering ":SHOWOUT SP;JOB=@". If suspended, any spoolfiles listed will be READY for printing; none will be ACTIVE, and a :SHOWDEV of the spooled device will indicate that the device is still spooled. Refer to the :SHOWOUT command in this manual.

When suspending an ACTIVE spoolfile, you should first take the output device offline. This gives you time to enter the command and determine that the ACTIVE file is the one being printed. If you issue :SUSPENDSPOOL without taking the device offline, that file might finish printing while you enter the command and another file might start.

When your instruction has been sent to the spooler process, MPE will return a colon prompt (:). The command will not be executed, however, until the output device is returned online. Only then will you receive the "SPOOLER SUSPENDED" message.

EXAMPLES

To suspend printing on logical device 6 (the line printer), enter:

```
:SUSPENDSPOOL 6
```

To suspend printing on logical device 6 once the currently active spoolfile is completely printed, enter:

```
:SUSPENDSPOOL 6;FINISH
```

:SWITCHLOG

Closes the current system log file, then creates and opens a new one.

SYNTAX

```
:SWITCHLOG
```

PARAMETERS

None.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It requires System Supervisor (OP) capability or must be issued from the Console.*

* Unless may be distributed to users with the :ALLOW command.

OPERATION

When the :SWITCHLOG command is executed, MPE displays the previous system log file number (*xxxxx*), the percentage of file space used (*yy*), and the current open log file (*zzzz*), as shown in the following example:

```
LOGFILE LOG xxxxx IS yy% FULL
LOG FILE NUMBER zzzz ON
```

If logging is enabled but currently suspended, MPE displays the following message:

```
NO LOGGING
LOG FILE xxxxx IS yy% FULL
```

NOTE

You should not :BUILD new log files, since MPE creates them automatically. If you do use :BUILD to create a new log file, and then attempt to switch the current log file to the file you created, user logging will suspend in an error state. The following message will be displayed:

```
LOG FILE NUMBER xxxxx ERROR #100.
LOGGING SUSPENDED.
```

EXAMPLE

To switch logging to a new Log File, enter:

```
:SWITCHLOG
```

ADDITIONAL DISCUSSION

Refer to :RESUMELOG in this section, and to the discussion of "MPE LOGGING FACILITIES" in the MPE System Operation and Resource Management Reference Manual (32033-90005).

:SYSDUMP

Starts the Configurator dialogue, and copies all system and user files to magnetic tape or serial disc and creates a backup of your system, user files, and configuration data.

SYNTAX

```
:SYSDUMP {$NULL}
         {dumpfile} [auxlistfile]
```

PARAMETERS

dumpfile	The destination file of the modified or duplicate system, previously defined with the :FILE command. The *dumpfile* must backreference a magnetic tape or serial disc file.
$NULL	Allows the operator to step through the configuration dialogue and alter system parameters without actually changing the system.
auxlistfile	Actual file designator of the output devicefile to which all listings requested during the Configurator/User dialogue are written. The formal file designator used by the MPE SEGMENTER for this file (when it is invoked to add or replace SL segments) is SEGLIST. Default is $STDLIST. It may backreference a magnetic tape or serial disc file.

USE

This command may be issued from a Session or a Job, but not in BREAK or from a Program. It is Breakable (suspends execution). It requires System Supervisor (OP) capability.

OPERATION

When the :SYSDUMP command is executed, you must direct MPE to write the output to a specific device. To do so, precede the :SYSDUMP command line with two file equations. Then back- reference these files by preceding the *dumpfile* parameter and the *auxlistfile* parameter with an asterisk (*) on the :SYSDUMP command line. To associate *dumpfile* with a tape drive, for example, enter:

```
:FILE DUMP;DEV=TAPE
```

Similarly, to produce the output listing from the SYSDUMP command on a line printer, enter:

```
:FILE LIST;DEV=LP
```

In general, use the following format for the :FILE command:

:FILE *formaldesignator*[=*filereference*];DEV=*device*[;REC=*recsize*[DEN=*density*]]

Where:

- *Device* specifies is the device class name or *ldev* number of a tape drive or serial disc.
- *Recsize* determines the size of the blocks on the store tape. The record size may be any multiple of 256 words, with a maximum record size of 8192 words for tape or serial discs.
- *Density* specifies the tape density in characters-per-inch (CPI). This parameter is only used for variable-density tape drives. The values allowed for tape drives are: HP 7976A - 1600, 6250; HP 7974 - 800, 1600; HP 7978 - 1600, 6250.

Once you have designated the peripheral devices that *dumpfile* and *auxlistfile* refer to, issue the :SYSDUMP command, preceding each filename with an asterisk (*) to indicate a backreference to the definitions in the :FILE commands.

The :SYSDUMP command invokes the :STORE command after writing system information to the beginning of the tape. Progress messages will appear once the system begins writing user files to the tape.

When using :SYSDUMP, you must specify the ;PROGRESS keyword during the :SYSDUMP dialog. Specify the ;PROGRESS keyword in response to the "ENTER DUMP FILE SUBSETS" prompt. Refer to "EXAMPLE". You may choose the default, one minute, for # *minutes*, or you may specify an interval. Refer to the :STORE command in this section.

NOTE

To start the Configurator dialogue without coping data to tape, use the system defined file $NULL as the *dumpfile*. MPE will accept this file specification, and open the file as though it were a standard file, but will perform no corresponding write operation. :SYSDUMP $NULL allows you to examine the current configuration, and even make "changes" to it, without affecting the system in any way. It will terminate with an error message, which you may ignore.

EXAMPLE

Enter :SYSDUMP to begin dialogue with the MPE Configurator. Once the dialogue is complete, configuration changes are copied to the "DUMP" file, and any listings requested during the dialog are output to the file named LIST. In the example below, the first file equation directs the output of the :SYSDUMP command to a tape drive. The second file equation will send any listings generated during the Configuration dialogue to a line printer.

```
:FILE DUMP;DEV=TAPE
:FILE LIST;DEV=LP
:SYSDUMP*DUMP,*LIST
```

During :SYSDUMP dialog you must specify an interval, in minutes, for progress messages (the default is one minute):

```
:SYSDUMP *T
ANY CHANGES? NO
ENTER DUMP DATE? 8/15/85
ENTER DUMP FILE SUBSET(S) @.@.@;FILES=20000;PROGRESS=3
```

ADDITIONAL DISCUSSION

For a full discussion of the :SYSDUMP command, refer to the sections titled "THE SYSTEM SUPERVISOR" and "CONFIGURING YOUR SYSTEM" in the MPE V System Operation and Resource Management Reference Manual (32033-90005).

Releases a device previously assigned to diagnostics so that it may be brought online.

SYNTAX

```
:TAKE ldev
```

PARAMETERS

ldev	The logical device number of the device taken from diagnostics.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

After diagnostic testing, the :TAKE command must be issued to release the device from diagnostic control. Only then can it be brought online with the :UP command.

EXAMPLE

To take logical device number 35 from diagnostics and bring it back online enter:

```
:TAKE 35
:UP 35
:SHOWDEV 35

LDEV  AVAIL     OWNERSHIP      VOLID       ASSOCIATION

 35 A AVAIL     UP
```

Sends a message to another session.

SYNTAX

```
        {[#]Snnn              }
        {username.acctname    }
:TELL   {@                    }[[;]text]
        {@.acctname           }
        {@S                   }
```

PARAMETERS

[#]S*nnn*	The session number as assigned by MPE. This session number will receive the TELL message.
username.accountname	The name of the session or user to receive the message, and the account name to which the message is directed. This parameter is the same as the session identity entered with the :HELLO command. Issuing a SHOWJOB commands lists all the *user.accountnames* to which you may direct a TELL message. Users who have issued a :SETMSG OFF command will be listed as being in QUIET mode and will not receive your TELL message. If several users are running under the same session identity, MPE will send the message to all of them.
@	All sessions.
@.*acctname*	All sessions under the account name established by the System Manager.
@S	All sessions. This is the same as the @ parameter.
text	Message text, preceded by a space or a semi-colon (:) and consisting of any string of ASCII characters. The default is that no text is printed; however, MPE still prints the "FROM" message as follows: FROM/*sessionid*

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable.

OPERATION

This command transmits a message from the sender's job or session to one or more sessions currently running. The message appears on the receiving session list device. Messages sent with this command may include escape and control characters that invoke bells or inverse video. If a message is sent to a terminal that is currently interacting with a program, MPE queues the message as high as possible among the current input/output requests but does not interrupt reading or writing in progress. If the session or user designated to receive the message is not running, or if the job is spooled, the transmitting job/session receives a system message indicating this. MPE blocks the :TELL command if the receiving device is operating in the quiet mode (refer to :SETMSG command) and informs the sender with:

```
Snnn username.acctname NOT ACCEPTING MESSAGES
```

You cannot send TELL messages to a job or to yourself. If you try to send a message to a job the following warning is issued:

```
TELL TO A JOB NOT ALLOWED. NO MESSAGE SENT TO QUALIFYING JOB(S).
 (CIWARN 1627)
```

EXAMPLES

To send a message to a user identified as BROWN logged on under account A running a session named BROWNSES telling him to use a particular file, enter:

```
:TELL BROWNSES,BROWN.A USE FILEX
```

To send a message asking all users logged on in account A to log off, enter:

```
:TELL @.A PLEASE LOG OFF
```

:TELLOP

Sends a message to System Console.

SYNTAX

```
:TELLOP [[;]text]
```

PARAMETER

text Message text, preceded by a space or a semi-colon (:) and consisting of any string of ASCII characters. Default is that no text is printed; however, MPE still prints the "FROM" as follows:

FROM/*sessionid*

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable.

OPERATION

This command sends a message to the System Console. The message text appears on the System Console, preceded by the time it was transmitted and your job/session number. Like messages transmitted between users (:TELL command), this message is printed as soon as possible without interrupting any console input/output currently in progress. The message can be sent to the System Console, even if no session is logged on.

EXAMPLE

To send a message to the System Operator instructing him to mount a tape, enter:

```
:TELLOP PLS MOUNT MYTAPE,VERSION 1
```

:TUNE

Changes the filter and/or priority limits of circular subqueues; it also changes the value of the memory manager algorithm threshold.

SYNTAX

```
                        {CQ}
:TUNE minclockcycle;{DQ}=[base[,limit[,min[,max[;...]]]]]
                        {EQ}
```

CAUTION

Misuse of this command can significantly degrade system operating efficiency.

PARAMETERS

minclockcycle Minimum number of milliseconds allotted for the replacement algorithm to cycle through memory. If it cycles through memory in less time, memory allocation is delayed and prevents disc thrashing.

base Priority at which processes that execute in the CS, DS, or ES subqueue begin in their respective queues. These processing queues, also referred to as "circular subqueues", are used by MPE to differentiate processes according to the amount of CPU processing time allotted to each, among other things. A process executing in the ES subqueue, for example, will be initiated only if all higher-priority processes are not ready to run. As a result that a process in the ES subqueue it may execute more slowly. In general, the more critical the process, the higher the subqueue it should be assigned.

limit Highest priority which processes executing in the CS, DS, or ES subqueues can attain.

min The minimum number of milliseconds (filter value) that a process may use the CPU before its priority is reduced.

max The maximum number of milliseconds (filter value) that a process may use the CPU before its priority is reduced. The value of *max* must be greater than or equal to the value of *min*.

USE

This command may be issued from a Session, Job, or in BREAK, but not from a Program. It is Breakable (aborts execution). It requires System Supervisor (OP) or System Manager (SM) capability.

OPERATION

The System Supervisor uses the :TUNE command to change the filter or priority limits of a circular subqueue and/or the value of the manager algorithm threshold. The command is used primarily in online tuning, so that the system can manage the current processing load efficiently.

For processes executing in the CS subqueue, the parameters *min* and *max* refer to the absolute limits of the "average short transaction time", or filter. This filter is recomputed after every transaction is complete, and then compared against the CPU time of each process to determine whether the priority of the process should be decreased. By contrast, processes executing in the DS or ES subqueues use a fixed value rather than an average. The same value is used for both queues and is specified by setting *min* and *max* equal to the desired value. Both *min* and *max* must be specified, but if they are not given the same value, the value of *max* is used.

At least one of the parameters (*minclockcycle*, *base*, *limit*, *min*, or *max*) must be specified. The values for *base* and *limit* may range from 150 to 255 . Parameters are positional: *min* is changed without specifying *base* and *limit*, it must be preceded by two commas. If *minclockcycle* is omitted, the queue specification (CQ , DQ , or EQ) must be preceded by a semicolon (;).

When a process begins executing in the CS subqueue, it is given a priority of CBASE. When a process stops (for disc I/O, terminal I/O, preemption, etc), its new priority is determined so that it may be requeued for the CPU. If the process has completed a transaction, defined as the time between terminal reads, its priority becomes CBASE , and the value of an "average short transaction" is recalculated. If the CS process has not completed a transaction, and if the process has exceeded the filter value since its priority was last reduced, the priority is decreased (but will not exceed the CLIMIT).

DS and ES processes begin at DBASE and EBASE respectively, and are rescheduled according to the same criteria used for CS process. However, a fixed value (the value of *max*, specified for the subqueue) is used in place of the "average short transaction time", which is used for CS processes only. If the values specified for *max* in the CS , DS , and ES queues are too large, system response may become erratic. If they are too small, excessive memory management may occur, because process-swapping occurs too frequently. In either case, system performance is degraded. The recommended settings for *min* and *max* for the CS subqueue are 0 and 300 respectively, since short transactions are favored in most cases. A value for *min* and *max* of 1000 usually produces efficient system operation for the DS and ES subqueues.

The values assigned for *limit* specify the lowest priority (largest number) that the system can assign to a process in a particular queue.

The *minclockcycle* parameter is used by the memory manager to determine when thrashing (excessive memory management activity) is occurring. Making this value smaller increases the probability of thrashing. The recommended value for this parameter is 1000 .

The following default settings are in when the system is cold loaded from the system disc (a WARMSTART or COOLSTART).

minclockcycle: 1000

CQ *base*:	152	DQ *base*:	202	EQ *base*:	240
limit:	200	*limit*:	238	*limit*:	253
min:	0	*min*:	1000	*min*:	1000
max:	300	*max*:	1000	*max*:	1000

EXAMPLE

To set the minimum clock cycle to 1000, CQ *base* to 152, *limit* to 200, and CQ maximum filter to 300; and DQ *base* to 202, DQ *limit* to 238, DQ (and EQ) minimum and maximum filter to 1000, enter:

```
:TUNE 1000;CQ=152,200,0,300;DQ=202,238,1000,1000
```

ADDITIONAL DISCUSSION

For additional information about process scheduling, refer to the discussion of "PROCESS SCHEDULING" in Section III of the MPE V Intrinsics Reference Manual (32033-90007).

Returns a particular device to its normal function on the system; cancels a :DOWN command that had been issued for the device.

SYNTAX

```
:UP ldev
```

PARAMETERS

ldev	The logical device number of the device being returned to service online.

USE

This command may be issued from a Session, Job, in a BREAK, or from a Program. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW or :ASSOCIATE commands.

OPERATION

This command makes available to users a device previously taken offline with the :DOWN command. Ownership of the device is not affected by the :UP command: If a device is owned by the system at the time it is downed, the system will retain ownership even after the :UP command is executed.

EXAMPLE

To allow logical device number 10 to function again, enter:

```
:UP 10
:SHOWDEV 10
LDEV   AVAIL    OWNERSHIP    VOLID   ASSOCIATION

10   A AVAIL
```

ADDITIONAL DISCUSSION

Refer to the :DOWN command in this manual.

:VINIT

Runs the VINIT subsystem to format, initialize, and label private volumes, serial discs, and foreign discs.

SYNTAX

```
:VINIT [listdevice]
```

PARAMETERS

listdevice — Name of the device to receive output generated by the VINIT subsystem. This parameter must backreference a :FILE command that names the device. Default is $STDLIST.

FUNCTIONS

>COND *ldev*
[;RECOVER]

Condenses free disc space and recovers lost disc space for private volumes.

>COPY *fromldn*,*toldn*
[;GEN=[*genindex*]]
[;VER[IFY]]
[;BUF=*size*]

Copies the contents of one volume to another.

>DSTAT [*ldev*]
[ALL]
[@]

Displays status of disc drives.

>DTRACK *ldev*

Processes suspect tracks in the Defective Track Table and defective sectors in the Defective Sectors Table. In doing so, you can either recover (i.e. ignore), delete, or reassign the defective tracks.

>EXIT

Exits the VINIT subsystem.

>FOREIGN *ldev*

Makes a disc volume foreign by filling the label area of a volume with zeros.

>FORMAT *ldev*
[;IBM[:*volid*]]

Allows online formatting of disc packs.

>HELP

Prints a listing of the VINIT commands, including syntax and descriptions, at $STDLIST.

>INIT *vname*,*ldev*
[,*vsname*,*group*,*acct*]
[;GEN=*genindex*]

Completes the conditioning of a private volume previously formatted by the SLEUTHSM program or the >FORMAT command.

>PDEFN [* .*group*.*account*]
[*vsname*.*group*.*account*]

Lists the specified volume set. The asterisk (*) specifies the home volume set. Default is the home volume set.

>PDTRACK *ldev*	Prints the contents from the Defective Tracks Table or the Defective Sectors Table at $STDLIST.
>PFSPACE {*ldev*[,ADDR]} {ALL }	Prints the Disc Free Space Map at $STDLIST.
>PLABEL *ldev*	Prints the disc label at $STDLIST.
>SCRATCH *ldev* [,RESET]	Writes (resets) a scratch-type label.
>SERIAL *ldev*	Writes a serial disc-type label.
>VERIFY *ldev* [,DTT]	Verifies (reads) data.

Each of the parameters referenced in the above :VINIT functions is defined below and discussed in detail under the heading "PRIVATE VOLUMES" in the MPE V System Operation and Resource Management Reference Manual (32033-90005).

ldev	The logical device number of the disc drive on which the volume is mounted.
fromldn	The logical device number of the disc from which data will be copied. Unless the device is a system domain disc, you must first remove it from normal system use by issuing a :DOWN command for the device.
toldn	The logical device number of the disc volume to which data will be copied. The device must be in the DOWN state. You must first remove the device from normal system by issuing a :DOWN command for the device.
genindex	A value from 0 to 32767, inclusive, indicating the generation index of the disc volume. If not specified, the *genindex* of *toldn* will be one greater than the *genindex* of *fromldn*. This parameter is valid only for private volumes.
VER[IFY]	Checks the readability of data on the destination drive. If an error is detected, VER[IFY] directs the :VINIT subsystem to attempt to recover the track. If it is unrecoverable, the track is reassigned and :VINIT will attempt to copy data once again. The VERIFY keyword may be abbreviated as VER.
size	The internal buffer size, which may be a value from 1 to 32, inclusive.
IBM[:*volid*]	Indicates that the disc is a single-sided IBM-formatted diskette. If *volid* is specified, it denotes an IBM volume identifier other than the default *volid* of IBMIRD.
vname	The name of the disc volume which is a member of the volume set.
vsname.group.acct	The name of the volume set.
ADDR	Displays the Disc Free Space entries for the specified logical device.

ALL or A	Displays the Disc Free Space Map in table format for all systems and private volumes which are physically mounted.
RESET	Resets a volume from scratch to non-scratch.
DTT	Directs :VINIT to verify only suspect sectors/tracks as recorded in the Defective Sectors/Tracks Table. This option is not applicable to serial or foreign discs or cartridge tapes.

USE

This command may be issued from a Session, or a Job, but not in BREAK or from a Program. It is Breakable (aborts execution). It requires System Supervisor (OP) or System Manager (SM) capability.

OPERATION

The :VINIT command accesses the VINIT subsystem, which includes commands for online initialization, labeling, and formatting of Private Volumes.

EXAMPLE

```
:FILE LIST;DEV=LP
:VINIT *LIST
```

NOTE

In JOB mode, the VINIT subsystem can be invoked only as follows:

```
!RUN PVINIT.PUB.SYS
```

ADDITIONAL DISCUSSION

Refer to VINIT in "SUBSYSTEMS AND UTILITIES" of the MPE V System Operation and Resource Management Reference Manual (32033-90005).

Enables or disables MPE's Private Volumes Facility.

SYNTAX

```
:VMOUNT {ON[,AUTO]} [,ALL]
        {OFF      }
```

PARAMETERS

ON or ON,AUTO	Enables the Private Volume Facility so that all valid user :MOUNT and Operator :LMOUNT requests are allowed. When "ON " is used without "AUTO ", the Operator must reply to all :MOUNT requests. When ON,AUTO is used, MPE attempts to satisfy user :MOUNT and Operator :LMOUNT requests without Operator intervention.
OFF	Requests to use the Private Volumes Facility will be rejected.
ALL	Prints all Private Volumes mount-related console messages, including those not requiring Operator intervention, on the Console.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

If the Private Volumes Facility is enabled when you issue a :VMOUNT OFF command, users having mounted volume sets will be unaffected; the command will be satisfied when the last access is complete.

Once the Private Volumes facility has been enabled, use the :DSTAT command to determine which volumes are actually mounted, and the :VSUSER command to determine which users are using particular volume sets. Refer to the :DSTAT and :VUSER commands in this section.

The Private Volumes Facility is disabled immediately following a system startup (the setting is equivalent to :VMOUNT OFF;ALL). However, you will still receive Console messages concerning private volume requests.

The Operator will have the greatest interactive control over the use of private volumes by using :VMOUNT ON;ALL. The command that least interrupts the Operator when users are accessing private volumes is :VMOUNT ON,AUTO.

EXAMPLES

To disable the Private Volumes Facility so that no messages are sent to the Console when users attempt to mount private volumes (the default condition) enter:

```
:VMOUNT OFF
```

To disable the Private Volumes Facility and still receive messages on the Console when users attempt to mount private volumes, enter:

```
:VMOUNT OFF;ALL
```

ADDITIONAL DISCUSSION

Refer to the discussion of "PRIVATE VOLUMES" in the MPE V System Operation and Resource Management Reference Manual (32033-90005).

:VSUSER

Lists all users of a currently mounted, nonsystem volume set.

SYNTAX

```
:VSUSER [vsname]
```

PARAMETER

vsname	A fully qualified volume set name. Default is that information for all currently mounted volume sets are listed.

USE

This command may be issued from a Session, Job, or in BREAK, but not from a Program. It is not Breakable.

OPERATION

The :VUSER command lists all users who have explicitly or implicitly mounted a nonsystem volume set. It also displays the volume set name, job number, and the group and account names of all users currently performing a mount function. The :MOUNT/ :DISMOUNT commands enable users to perform an explicit mounting; the FOPEN/CLOSE intrinsics enable them to perform an implicit mounting. You must have UV (Use Volumes) or CV (Create Volumes) capability to use this command. Refer to the :MOUNT and :DISMOUNT commands in this section.

EXAMPLE

To list all of the currently mounted volume sets, enter:

```
:VSUSER
VOLUME SET NAME      JOBNUM      JOBNAME
---------------      ------      ----------
USER.MANAGER         #S260       NORMA.MPEM
```

ADDITIONAL DISCUSSION

MPE V System Operation and Resource Management Reference Manual (32033-90005)

MPE V/R Intrinsics Manual (32033-90007)

Sends an urgent message to jobs/sessions.

SYNTAX

```
        {@                      }
:WARN   {[#]Jnn                 }
        {[#]Snnn                }[;message]
        {[jsname,]user.acct     }
```

PARAMETERS

@	All users receive the message (QUIET and nonquiet).
#J*nn*	A job number (assigned by MPE) for the job that is to receive the message.
#S*nn*	A job number (assigned by MPE) for the job that is to receive the message. Only jobs submitted on interactive devices can receive messages.
jsname,user.acct	The names of the job/session and user to receive the message, and the account name under which they are funning. (These names are the same as those entered with the :JOB or :HELLO command.) If several users are running under the same job/session identity, MPE will send the message to all of them.
text	The message text, consisting of any string of ASCII characters no more than 67 characters. The message is terminated by a (RETURN). Default is that no message is printed.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

Sends an urgent message, interrupting any current pending read or write in progress. The message will appear on the list devices of all sessions (even those that are QUIET) as:

OPERATOR WARNING: *text*

Messages sent with the :WARN command will be received by jobs only if the job was submitted on an interactive device.

The user has the option of running his session in QUIET mode. In that case, :TELL messages from other users will be suppressed. :WARN messages generated at the system console, however, override QUIET mode.

NOTE

Use caution when sending a warning to users. The :WARN command overrides a block mode screen.

EXAMPLE

To send a WARN message to all sessions, followed by a WARN message to session #S51 , enter:

```
:WARN @;THE SYSTEM WILL SHUTDOWN IN 5 MINUTES. PLS LOG OFF.
:WARN #S51;LAST CHANCE TO LOG OFF GRACEFULLY.
```

:WELCOME

Defines the welcome message.

SYNTAX

```
:WELCOME [welcfile]
```

PARAMETERS

welcfile	An ASCII file containing the new welcome message.

USE

This command may be issued from a Session, Job, in BREAK, or from a Program. It is not Breakable. It may be issued only from the Console.*

* Unless distributed to users with the :ALLOW command.

OPERATION

The Operator uses the :WELCOME command to compose the message that will be transmitted to users when they initiate jobs and sessions. The message is retained when you WARMSTART or COOLSTART the system, or choose the UPDATE restart option.

To define the welcome message, enter ":WELCOME" and press "RETURN". When the "#" prompt is displayed, begin entering the text of the message. Any number of lines may be used, but the length of any line cannot exceed 72 characters. To terminate the message and complete the command, enter RETURN at the "#" prompt.

To define the welcome message from an editor file merely specify the *welcfile* parameter. Subsequent changes to the editor file will not affect the welcome message until another :WELCOME command is issued with the filename specified.

NOTE

If no parameter is specified, you are prompted to enter the new message interactively.

To delete the old welcome message, issue the :WELCOME command and enter RETURN at the # prompt.

EXAMPLE

To create a multi-line welcome message, enter:

```
:WELCOME
# WELCOME TO THE HP3000 COMPUTER SYSTEM.
# FILES WILL BE STORED EACH DAY BETWEEN 6AM AND 7AM.
# RETURN
```

USER DEFINED COMMANDS

SECTION III

MPE allows you to define your own commands to fit specific needs. To build a user defined command, or UDC, you may use Editor, or another ASCII text editor, to combine several standard MPE commands into a single list. Assign a name to that list and save it. To invoke the UDC, enter its name at an MPE prompt and press RETURN: all of the commands in the UDC list will execute. A user defined command may be used in most instances where a standard MPE command may be used, even within another UDC. User defined commands cannot, however, be executed through the COMMAND intrinsic, or from within subsystems, unless you are in BREAK.

The UDC definitions are stored in one or more files. These files can exist at one of three different levels: user, account, or system. UDCs may be created by any user for personal use, by the Account Manager for anyone using the account, or by the System Manager for user on the system. System wide UDCs must be defined in a file that is released for all users to use.

When identically named UDCs occur at different levels, the user level UDC takes precedence over the account level UDC, which in turn takes precedence over the system level UDC. For this reason, it is a good practice to give your UDCs unique names. Exceptions to this rule, nested and logon UDCs, are discussed later in this section.

You must have special capabilities to establish account or system level UDCs. For information on System Manager, Account Manager, and System Supervisor capabilities, refer to the MPE V System Operation and Resource Management Reference Manual (32002-90005).

NOTE

No UDC name may consist of, or start with, the characters RFA, since RFA is reserved for internal use. Names like RFAST, as well as the name RFA, are thus illegal. Although the HELP subsystem will show such UDCs in their correct form, the command interpreter will not recognize them and will issue the error message: "UNKNOWN COMMAND NAME (CIERR 975)".

The :SHOWCATALOG command lists the UDCs currently in effect, the level at which they are defined, and the file in which they reside:

```
:SHOWCATALOG                    ** Name of file containing UDC definitions    **
                                ** UDC name and level                         **

UDCU.TOM.MPE
   WELCOME              USER
   T                    USER
UDCA.PUB.MPE
   LIST                 ACCOUNT
   FCOPY                ACCOUNT
UDCS.PUB.SYS
   R                    SYSTEM
```

If a UDC name that you do not recognize appears in the catalog, the :HELP command can be used to discover the content of that UDC, unless OPTION NOHELP has been specified for that UDC. Refer to "EXECUTION OPTIONS", later in this section. Suppose, for example, that :SHOWCATALOG reveals an unfamiliar UDC named SJ. To find out exactly what that UDC contains, enter:

```
:HELP SJ      **The UDC name used as a parameter of HELP**

USER DEFINED COMMAND:

SJ
SHOWJOB
:
```

SYNTAX OF UDCs

A user defined command has of two parts: a header section containing control information, and a body consisting of MPE commands.

Header

The control information in the header section of a UDC consists of:

- The command name, composed of alphanumeric characters (16 maximum) and beginning with an alphabetic character. You cannot use the reserved word RFA as a UDC name or at the beginning of a UDC name.
- Parameters and their optional default values (maximum 16).
- Execution options (LIST/NOLIST, BREAK/NOBREAK, LOGON/NOLOGON, HELP/NOHELP).

THE COMMAND NAME. The first line (including continuation lines) in the header of a UDC gives the Command Interpreter the name of the command, any parameters, and the defaults for those parameters. An minimum first line could be:

```
A
```

In this example, the command is named A. It has no parameters. UDC headers may be more complex, if you choose to specify options or parameters. A UDC header may extend over more than one physical line and may contain as many has 320 characters. Each line to be continued must end with an ampersand (&) as the last nonblank character.

PARAMETERS. UDC parameters are variables that are passed from the calling command (the UDC name) to the body of the UDC. This allows a general UDC to serve different specific purposes.

EXECUTION OPTIONS. There are several options that you may wish to specify in the UDC header. The options, or their defaults are invoked when the UDC executes. The options available for use in a UDC declaration are explained in the following pages.

List/Nolist. The LIST option of a command allows the text of the UDC body to be listed on the standard list device after all parameters have been substituted. With the LIST option, errors are reported with a caret (^) under the error. Default is NOLIST. An example is:

```
TDPOFFLINE
OPTION LIST
FILE LP;DEV=EPOC;CCTL;ENV=LP2.ENV2680A.SYS
RESUME
```

When the UDC is used the text of the body is printed:

```
:TDPOFFLINE
FILE LP;DEV=EPOC;CCTL;ENV=LP2.ENV2680A.SYS
RESUME
READ pending
```

Break/NoBreak. If NOBREAK is specified, the command will be nonBREAKable. If the NOBREAK option is not specified, all BREAKable commands within the UDC remain BREAKable. Default is BREAK.

If a nonprogram command is broken, the UDC terminates. If a program command is broken and you use the :RESUME command to resume the program, the UDC will also resume. An exception to this occurs if the :SETCATALOG command is used while you are in the BREAK facility. In this case, the UDC will not be resumed. Refer to "USING THE :SETCATALOG COMMAND", this section.

The UDC NOBREAK option overrides the CAUSEBREAK intrinsic. A program containing CAUSEBREAK will not BREAK if it is executed from a UDC that specifies the NOBREAK option.

Suppose your UDC file contained the following:

```
S
SHOWJOB
***
SJ
OPTION NOBREAK
SHOWJOB
***
```

If you use the "S" UDC, you could BREAK the listing as follows:

```
:S
JOBNUM  STATE IPRI  JIN  JLIST    INTRODUCED JOB NAME
#S55    EXEC         20  20       MON  2:49P FIELD.SUPPORT
#S57    EXEC                                  ** [BREAK] pressed here. **
:
:RESUME
COMMAND ONLY ALLOWED IN BREAK. (CIWARN 1686) ** [BREAK] aborts :SHOWJOB.**
:
```

If you used the "SJ" UDC, you could not BREAK the listing. Enter:

```
:SJ

JOBNUM  STATE IPRI  JIN  JLIST    INTRODUCED JOB NAME

#S55    EXEC         20  20       MON  2:49P FIELD.SUPPORT
#S57    EXEC         52  52       MON  2:50P BOB.DATAMGT

2 JOBS:
    0 INTRO
    0 WAIT; INCL 0 DEFERRED    ** (BREAK) pressed here has no effect **
    2 EXEC; INCL 2 SESSIONS
    0 SUSP
JOBFENCE= 2; JLIMIT= 2; SLIMIT= 16
```

Logon/Nologon. The LOGON option specifies that the UDC will automatically executed when you logon. There are three levels in the UDC hierarchy: user, account, and system. One logon, at most UDC will be executed at each level. For users who have logon UDCs in each of the levels, the system logon UDC will execute first, then the account logon UDC, and finally the user logon UDC. If more than one logon UDC exists within any one particular level, only the first logon UDC will execute for that level without affecting the execution of logon UDCs at any other level. Default is NOLOGON. The first UDC at any level is that UDC named first in the :SETCATALOG command for that level. The UDC file *ufile01* will be the first after the command :SETCATALOG ufile01, *ufile01*, *ufile01* is executed.

For example, suppose your UDC was:

```
S
OPTION LOGON
SHOWJOB
***
```

When you logon, "S" automatically executes:

```
:HELLO BARBARA.USERS


HP3000 / MPE V  G.02.00 (BASE G.02.00).  MON, MAY 14, 1984,  8:43 AM


JOBNUM  STATE IPRI  JIN  JLIST    INTRODUCED JOB NAME

#S55    EXEC         20  20       MON  6:43A FIELD.SUPPORT
#S57    EXEC         52  52       MON  6:45A BARBARA.USERS

2 JOBS:
    0 INTRO
    0 WAIT; INCL 0 DEFERRED
    2 EXEC; INCL 2 SESSIONS
    0 SUSP
JOBFENCE= 6; JLIMIT= 7; SLIMIT= 40
```

Help/Nohelp. The HELP subsystem may be used to obtain information about user defined commands, unless NOHELP is specified as an option. The default is HELP. Default is HELP. The MPE HELP subsystem does not contain a listing of UDCs. Therefore, entering the HELP subsystem first will defeat your purpose in trying to list the contents of a UDC. Instead, entering :HELP *udcname* forces the HELP subsystem to search out cataloged UDCs.

The HELP subsystem will respond with a listing of the UDC:

```
:HELP GRAPHOUT
USER DEFINED COMMAND:

GRAPHOUT
FILE GRAPHOUT;DEV=EPOC
RUN GRAPH.PUB.SYS
:
```

The following example illustrates the result of an attempt to use the :HELP command on a UDC for which the NOHELP option has been specified:

```
:HELP S    ** The "S" UDC contains a NOHELP option **

Can't find anything under this command or in Table of Contents.
```

NOTE

If you specify both the NOLIST and NOHELP options and an error occurs, the error will be reported, but the line containing the error will not be echoed.

Body

As in the UDC header, any logical line in a UDC header may continue over more than one physical line and may contain as many as 320 characters. Each line to be continued must end with an ampersand (&) as the last nonblank character. The body portion of a UDC contains MPE commands which are executed when the UDC name is entered in place of an MPE command. In addition to MPE commands, you may also include other user defined commands (nested UDCs) and comment lines. You may not include any other noncommand material such as data for MPE subsystems or user programs.

Several other restrictions also apply. The :REDO command is not allowed from within a UDC. Using :DATA, :JOB, or :HELLO within a UDC will cause your current job or session to log off, thereby terminating the UDC execution; no new job or session will be logged on. The body may also contain a parameter list which supplies values for the parameters defined in the UDC command header. Refer to "USING PARAMETERS" in this section.

USING THE :SETCATALOG COMMAND

The :SETCATALOG command informs MPE that a specified file name contains user defined commands to be made available to the specified user, account, or system. The Command Interpreter then searches the file and establishes a directory of all commands contained in the file. The filename is stored in a system catalog of all UDC users (COMMAND.PUB.SYS).

If you want to carry out such a procedure for a file named MYUDC1, for example, enter:

```
:SETCATALOG MYUDC1
```

You may have more than one file to be entered into the directory. In this case, separate the files by commas. For example enter:

```
:SETCATALOG KEYUDCS,UDCFILE
```

The two files will be entered into the system catalog as a complete and separate set, and will replace any set of UDC files already existing in the catalog.

You cannot use the :PURGE command on any UDC file that anyone is accessing. Nor can you use Editor to modify and keep a file under the same filename while anyone whose directory contains that file is logged on. Any attempt to do either of these two operations will result in an error message: "EXCLUSIVE VIOLATION".

Entering :SETCATALOG with no parameters removes all your UDC files from the system UDC catalog. The files are not purged, but are deleted from the catalog. To purge or modify a particular UDC file, you must remove it from the system UDC catalog by issuing the :SETCATALOG command with a parameter list that does not include that particular UDC file.

To re-enter UDC files into the catalog, enter the :SETCATALOG filename command for these files. Refer to the :SETCATALOG command in Section II.

Removing UDCs with the :SETCATALOG command (for example, :SETCATALOG;SYSTEM to remove system wide UDCs) has no immediate effect upon jobs/sessions that are still logged on, except for the job/session that enters the :SETCATALOG command. Another job/session may continue to access the canceled UDCs until that job or session terminates, or until each job or session forces the creation of a new UDC directory with a :SETCATALOG command.

A user who does not have READ and LOCK access to a UDC file cannot use the :SETCATALOG command on that file, or logon while the file remains active in the system catalog. Therefore, security for the account and group in which the file resides must allow READ and LOCK access to all users (R, L; ANY) at the account and system level. The default file access for the PUB group only gives READ access to users with AL, GL, or home group (GU) capability. To override the default, you should use ":RELEASE *filename*" where *filename* is the system level UDC file. Refer to the :RELEASE command in Section II.

If the UDC file has a lockword associated with it, that lockword may be specified when using the :SETCATALOG as in the following example:

```
:SETCATALOG file1/lockword
```

The file and its associated lockword will then reside in COMMAND.PUB.SYS, so users need not know the lockword to logon or to access the UDC. If you fail to specify the lockword when using the :SETCATALOG command, MPE will prompt you for it. If you respond with an invalid lockword, the file will not be entered into COMMAND.PUB.SYS.

The USER option of the :SETCATALOG command allows those with Account Manager (AM) capability to assign UDC catalogs to any user in their account. It also allows those with System Manager (SM) capability to assign UDC catalogs to any user in the system. Any changes will take effect the next time the user logs on.

If the :SETCATALOG command is part of the user defined command, the :SETCATALOG command will be the last command executed in the UDC body; any subsequent commands in the UDC definition will not be executed. Further, if the UDC performing the :SETCATALOG is a nested UDC (i.e. invoked by another UDC), all levels of UDC execution are terminated, and you are returned to the Command Interpreter.

If you execute a UDC that invokes a program or subsystem and you press (BREAK) while the program is running, you may execute a :SETCATALOG during the BREAK (with a direct command or through another UDC). Executing a :RESUME command will continue the program from the point at which the BREAK occurred. However, the UDC will fail to execute subsequent program/subsystem calls.

For example, suppose you have a UDC which first calls the Editor subsystem, then performs a :SHOWOUT when you exit the Editor. You invoke the UDC and enter the Editor. While in the Editor, you press (BREAK) and enter the :SETCATALOG command to disassociate your UDC file from your account. The :RESUME command will return you to the Editor, but the :SHOWOUT will not occur when you exit.

The system UDC catalog, COMMAND.PUB.SYS, must exist for the :SETCATALOG command to execute properly. If this file does not exist on your system, the System Manager or Supervisor must build it. The file should normally be built with a record size of 20. The maximum number of records in the file COMMAND.PUB.SYS is determined by the sum of:

the number of usernames with user level UDCs; plus
the number of UDC catalog files specified for each user; plus
the number of account level UDCs; plus
the number of UDC catalog files for each account; plus
one, if system level UDCs are in effect; plus
the number of UDC catalog files specified for system UDCs.

The System Manager could build a file for the system catalog in the following way:

```
:BUILD COMMAND.PUB;REC=20,32;DISC=2000
```

To secure COMMAND.PUB.SYS so that all system users may have UDCs, but only the System Manager may read or modify COMMAND.PUB.SYS. The System Manager should enter:

```
:ALTSEC COMMAND.PUB; (X:ANY;R,L,W,A:CR)
```

Doing so will provide EXECUTE (X) access to all users (ANY), and READ (R), LOCK (L), WRITE (W), and APPEND (A) access are available only to the Creating User (CR), in this case the System Manager. For more information on file security specifications, refer to the :ALTSEC command in Section II of this manual.

BUILDING AND MODIFYING A UDC FILE

Once you have decided upon a command, or set of commands, that you would like to save, you may enter them into an ASCII disc file using the Editor. This file will have a default record length of 80 bytes for numbered files (standard Editor file), or 72 bytes for unnumbered files. You may create several such files, each consisting of one or more user defined commands. If a file contains more than one user defined command, the commands must be separated from each other by one line containing one or more asterisks, starting with an asterisk in the first column.

The following is an example showing how to create user defined Commands with the Editor:

```
:EDITOR
HP32201A.7.12 EDIT/3000  MON, MAY 7, 1984, 3:36 PM
(C) HEWLETT-PACKARD CO. 1982
/ADD
1      S          ** Header **
2      SHOWJOB    ** List of Commands **
3      //
...
/KEEP UDC1
/DELETE ALL
/ADD
1      L
2      LISTF
3      ****       ** Separates UDCs **
4      R
5      RESUME
6      //
...
/KEEP UDC2
/EXIT

END OF SUBSYSTEM
:SETCATALOG UDC1,UDC2
```

After setting the catalog, log off. Then log back on. Now you can use the UDCs you created for :SHOWJOB, :LISTF, and :RESUME. Using "S" to invoke :SHOWJOB is illustrated below:

```
:S

JOBNUM  STATE IPRI  JIN  JLIST    INTRODUCED JOB NAME

#S55    EXEC         20  20       MON  2:49P FIELD.SUPPORT
#S57    EXEC         52  52       MON  2:50P BOB.DATAMGT

2 JOBS:
    0 INTRO
    0 WAIT; INCL 0 DEFERRED
    2 EXEC; INCL 2 SESSIONS
    0 SUSP
JOBFENCE= 2; JLIMIT= 2; SLIMIT= 16
:
```

The Editor can be used to modify commands stored in a UDC file if that file is not being accessed. Note, however, that once a UDC file has been entered into the system UDC catalog with the :SETCATALOG

command, it cannot be modified and kept under the same name while any user whose directory contains that UDC file is logged on.

For example, suppose you are the only user with the file UDC1, as defined on the previous page, as your UDC file. To modify UDC1:

```
:SHOWCATALOG

UDC1.SMR.OSE
   S                    USER
:EDITOR

HP32201A.7.12 EDIT/3000  MON, MAY 7, 1984, 3:36 PM
(C) HEWLETT-PACKARD CO. 1982
/T UDC1
/A 1.1
    1.1   OPTION NOBREAK
    1.2   //
...

/K UDC1
--F-I-L-E---I-N-F-O-R-M-A-T-I-O-N---D-I-S-P-L-A-Y--
|  ERROR NUMBER:  90     RESIDUE:  0               |
|  BLOCK NUMBER:  0              NUMREC:  0        |
-------------------------------------------------
*41*FAILURE TO OPEN KEEP FILE   (90)
EXCLUSIVE VIOLATION: FILE BEING ACCESSED (FSERR 90)
/                    ** [BREAK] pressed here. **
:SETCATALOG          ** File disassociated from system UDC catalog. **
:RESUME
READ PENDING
K UDC1
UDC1 ALREADY EXISTS - RESPOND YES TO PURGE OLD AND KEEP NEW
PURGE OLD?Y
/E
END OF SUBSYSTEM
:SETCATALOG UDC1
```

DETAILS OF UDC OPERATION

UDCs are initialized at logon, provided that the user, account, or system level UDCs exist and have been associated by the :SETCATALOG command, or whenever the user executes a :SETCATALOG command. The system file, COMMAND.PUB.SYS, holds the UDC filenames associated with each user, account, and system UDCs, and includes pointers to the files in which the UDCs are defined. At initialization time, all the UDC files needed by the user are opened, and a UDC directory is created for the user.

The position of one UDC in the directory determines which other UDCs can call it and which others it can call. The UDC hierarchy imposed by the system depends on how the UDCs entered into the directory are found in the file. The files are opened and scanned in the order they appeared in the last :SETCATALOG command for that UDC level. If, for example, the UDC files CAT1 and CAT" are enabled with :SETCATALOG CAT1, CAT2, then any UDCs in CAT1 may call any UDCs in CAT2. The reverse is not true: UDCs in CAT2 cannot call UDCs in CAT1. All user UDCs are entered into the directory first, then all account UDCs, and all system UDCs are entered last.

If the UDC file has a lockword, you do not need to know the lockword to logon or to access the UDC, since the lockword is kept in COMMAND.PUB.SYS. The System Manager can release COMMAND.PUB.SYS, so that all system users may access and execute the UDCs. But only the System Manager may read or modify the COMMAND.PUB.SYS by entering the :ALTSEC command. For more information, refer to "USING THE :SETCATALOG COMMAND" in this section.

When you are in the Command Interpreter (CI) and have a colon prompt (:), or a remote prompt (#), you can execute a UDC by entering the command name, followed by a parameter list (if needed). The UDC directory is scanned for that command, starting at the beginning of the directory. If the command is found, the UDC body for that command is read one line at a time, and parameters are substituted into the line where appropriate.

The CI is re-entered at a special internal entry point. This UDC entry is treated as a new command string, and is handled much the same as all other command strings: the UDC directory is scanned for a UDC that matches the command string. This time, however, the UDC directory scan begins at the directory entry that follows the UDC currently being executed. In this way, UDCs can call other UDCs as long as the called UDC is defined after the UDC being executed. If the Command Interpreter fails to find a match for the command string in the UDC directory, it then tries to match the command string with an MPE command. This implementation allows some UDCs to call other UDCs, but does not allow any kind of UDC recursion. This linear scan of the UDC directory is the heart of UDC invocation. This particular scan will allow UDCs with the same name to be executed at different times.

Note that the names of UDCs may be composed of alphanumeric characters, but must begin with an alphabetic character. MPE commands, however, contain only alphabetic characters. A string such as "SHOWDEV6" is treated first as a UDC, because the last character is numeric. If no UDC exists under that name, the MPE command :SHOWDEV is passed to the parameter. Normal MPE command syntax requires a space between the command and the parameter. The coincidence of names in this example is unusual and produces unexpected results: "SHOWDEV" is a valid MPE command, and MPE is capable of separating the command from the parameter and executing them accordingly.

NESTING UDCs

User defined commands can be nested; that is, they can be so defined that entering one UDC will cause several UDCs to execute. For example:

```
DOALL
L
S
****
L
LISTF
****
S
SHOWJOB
```

If the preceding set of UDCs is entered into a UDC directory, then each time DOALL is entered, the MPE commands :LISTF and :SHOWJOB will execute.

Note, however, that a UDC can call another UDC only if the UDC being called is defined after the calling UDC.

If, in the example, DOALL is put in the UDC file after the L and S user defined commands, the UDC DOALL will not execute, and an error message will result:

```
L
LISTF
***
S
SHOWJOB
***
DOALL
L
S
```

When the command DOALL is entered, the UDCs that it calls will not be found:

```
:DOALL
L
^
UNKNOWN COMMAND NAME.  (CIERR 975)
:
```

System level UDCs are global to account level UDCs, which in turn are global to user level UDCs. This means that an account level UDC can call a system level UDC, and a user level UDC can call either account or system level UDCs.

If a system and a user level UDC have the same name, and an account level UDC calls a UDC by this name, the system level UDC will be called instead of the user level UDC, because the system level UDC is global to the user level UDCs.

ERRORS IN A USER DEFINED COMMAND

A user defined command contains an error if the execution of an MPE command or a user defined command within the body of the UDC results in an error; or if the expansion of the UDC parameters results in an error. If an error or warning occurs as a result of executing a UDC, MPE will:

1. Print the appropriate error message, always.

2. Print a caret (^) pointing to the error, unless the NOHELP option is specified in the UDC.

3. Print the line in which the error occurs, unless the NOHELP and NOLIST options are specified.

Every time a user logs on, the system initializes the user's UDC directory, as discussed earlier in this section. If an illegal UDC is encountered during initialization, or if the UDC file cannot be opened (perhaps it has been purged), only the UDC level in which the error occurs will fail to be initialized. If, for example, the user level UDC file encounters an error at logon, the user will still have access to account and system level UDCs. Similarly, if an account level UDC encounters an error at logon, the user will still have access to user and system level UDCs.

If you anticipate difficulty from a particular command line, you may insert a :CONTINUE command in the line immediately preceding the potentially troublesome line.

The CONTINUE command permits the MPE to bypass the error and proceed with subsequent command lines. In a job, a UDC that generates an error will terminate the job, unless the UDC contains a :CONTINUE command at one of the current nesting levels.

Consider the following UDC file:

```
A
LISTF A@
B
SHOWJOB
***
B
BUILD FILEX;CODE=XXXX
SHOWTIME
***
```

Suppose the UDC A is executed:

```
:A

FILENAME

APPA        APPB        APPB1VE     APPB2       APPB2VE     APPB3
APPB3VE

BUILD FILEX;CODE=XXXX
                    ^
UNKNOWN FILE CODE TYPE.  (CIERR 253)
```

UDC B contains an error because the :BUILD command contains an invalid file code. Therefore, the UDC B terminates and the :SHOWTIME following the :BUILD command is not executed. Because the execution of UDC B results in an error, :SHOWJOB in UDC A is not executed.

A :CONTINUE may be placed before the :FILE command in UDC B to allow both UDCs to execute

```
A
LISTF A@
B
SHOWJOB
***
B
CONTINUE
BUILD FILEX;CODE=XXXX
SHOWTIME
***
```

Suppose the UDC A is executed:

```
:A

FILENAME

APPA        APPB        APPB1VE     APPB2       APPB2VE     APPB3
APPB3VE

BUILD FILEX;CODE=XXXX
                  ^
UNKNOWN FILE CODE TYPE.  (CIERR 253)
MON, MAY  7, 1984,  7:39 AM

JOBNUM  STATE IPRI JIN  JLIST    INTRODUCED  JOB NAME

#S118   EXEC        20  20       MON  9:02P  OPERATOR.SYS
#S48    EXEC QUIET 223  223      MON  1:51P  TEXT.USER
#S69    EXEC       179  179      MON  2:31P  SUPPORT.DOC

3 JOBS:
    0 INTRO
    0 WAIT; INCL 0 DEFERRED
    1 EXEC; INCL 3 SESSIONS
    0 SUSP
JOBFENCE= 2; JLIMIT= 5; SLIMIT= 40
```

USING PARAMETERS

The name given to a parameter in the header of a UDC is the formal name by which the parameter is known to the commands in the body of the UDC. "Required" parameters are those for which no default values are supplied in the command header. If a value for a required parameter is not supplied in the parameter list, an error is reported, and the UDC is not executed. If there are no required parameters in the UDC header, a parameter list does not have to be supplied when the command is executed.

Types of Parameters

A "keyword" parameter is one in which the formal name of the parameter appears in the parameter list exactly as it appears in the parameter declaration. The Command Interpreter makes no distinction, though, between upper and lower case alphabetic characters. A "keyword" parameter may appear in any position or in any order in the parameter list.

A "positional" parameter requires the substitution of a name or value in the parameter list in place of the formal name in the parameter declaration. "Positional" parameters must appear in the parameter list in the same relative position occupied by the formal name in the parameter declaration.

When supplying values for UDC parameters you can use either keyword parameters or positional parameters, but not both at the same time. MPE permits both decimal and octal numbers as command parameters. Octal numbers are preceded by a percent sign (%). The following are examples of different parameter lists for a UDC named Z, with this header:

```
Z PARM1,PARM2=FTN06,PARM3="REC=40,16,F,ASCII"
```

EXAMPLE 1:

```
:Z FTN05
```

The specified required parameter PARM1 would take the value of FTN05; PARM2 would default to FTN06, and PARM3 would default to REC=40,16,F,ASCII.

EXAMPLE 2:

```
:Z FTN05,FTN77
```

PARM1 would take the value FTN05, and PARM2 the value FTN77. The default value for PARM2 would not be used because the call contained a new value. The default value (REC=40,16,F,ASCII) would be used for PARM3 because PARM3 was not specified. Both FTN05 and FTN77 are positional parameters because they substitute for PARM1 and PARM2.

EXAMPLE 3:

```
:Z PARM2=FTN77,PARM1=FTN05
```

Example 3 illustrates entering the command Z with keyword parameters. Note that this call is identical in its effect to the call in Example 2. PARM3 would take its default value (REC=40, 16,F,ASCII).

EXAMPLE 4:

```
:Z FTN05,,"REC=-88"
```

Example 4 illustrates double quotation marks (" ") to delimit a parameter when that parameter contains embedded blanks or special characters. Also note that the second parameter is not specified (position is denoted by the adjacent commas), thus the default value FTN06 is used for PARM2, and (REC=-88,16,F,ASCII) is used for PARM3.

The body of the user defined command follows the header and specifies the MPE commands that the user defined command Z is to execute:

```
FILE INPUT=!PARM1,OLD
FILE LISTFILE=!PARM2,NEW;!PARM3;DISC=1000
RUN ABC
```

Thus, the complete user defined command Z is:

```
Z PARM1,PARM2=FTN06,PARM3="REC=-40,16,F,ASCII"          ** Header **
FILE INPUT=!PARM1,OLD                                   ** Body **
FILE LISTFILE=!PARM2,NEW;!PARM3;DISC=1000
RUN ABC
```

Parameter Delimiters

Placing an exclamation mark (!) before a formal parameter name appearing in the body of the command instructs MPE to substitute the value of that parameter into the command. If a parameter name does not follow the exclamation point, then the string is retained literally, with no error reported to the user.

The first line of the UDC body tells MPE that !PARM1 will take the value of an input file to be specified later:

```
FILE INPUT=!PARM1,OLD
```

Similarly, the second line of the UDC body informs MPE that you wish to substitute an output file called FTN06 (a default value in this example) for !PARM2:

```
    FILE LISTFILE=!PARM2,NEW;!PARM3;DISC=1000
```

Notice that the command body of UDC Z contains only the MPE commands :FILE and :RUN, and parameters for these commands:

```
FILE INPUT=!PARM1,OLD                          ** Body of UDC Z **
FILE LISTFILE=!PARM2,NEW;!PARM3;DISC=1000
RUN ABC
```

In the UDC, PARM1 is a required parameter (it has no default value), so its value must be supplied in the parameter list. The parameters PARM2 and PARM3, however, do have specified default values and so are optional. They will be passed to the command body either as default values, or as values you enter in the parameter list with the commands:

```
Z PARM1,PARM2=FTN06,PARM3="REC=40,16,F,ASCII"
FILE INPUT=!PARM1,OLD
FILE LISTFILE=!PARM2,NEW;!PARM3;DISC=1000
RUN ABC
```

Note that the parameters DISC=1000 (for the :FILE LISTFILE command) and ABC (for the :RUN command) cannot be changed by the user-entered command call.

An exclamation mark in a UDC normally signifies a parameter name. There might be occasions when you want to retain the exclamation mark without signifying a parameter name. Doubling the exclamation mark (!!) prevents MPE from treating the characters that follow as a parameter name. An odd number of exclamation marks preceding a set of characters causes MPE to expect a value to substitute for the character set. An even number of exclamation marks preceding a set of characters tells MPE that the character set is not a parameter and tells it not to attempt a value substitution:

```
TESTUDC PARM
OPTION LIST
COMMENT"!!PARM"
COMMENT"!!!PARM"
```

If the user executes this UDC with ":TESTUDC ABC", the two comments will be printed as:

```
COMMENT "!!PARM"
COMMENT "!!ABC"
```

The UDC parameter can be delimited by the use of double quotes ("). Doing so allows the user to insert the parameter into the middle of another string. The quotes are not part of the command, although other blank or nonblank characters within the quotation marks may be. Imagine, for example, having application program to compile and prepare daily:

SOURCE	USL	PROGRAM
APPL1SRC	APPL1USL	APPL1

You could easily create s UDC to combine the steps to prepare any of the above three source files into a program file (note the commented delimiters on the parameters):

```
COMPAPPL MODULE
OPTION LIST
PURGE APPL!MODULE                                    ** Blank **
SPL   APPL!"MODULE"SRC, APPL!"MODULE"USL             ** Quotes **
PREP   APPL!"MODULE"USL, APPL!MODULE;~PMAP           ** Semicolon **
SAVE  APPL!MODULE                                    ** Blank **
```

After adding the COMPAPPL UDC to the catalog, you may compile and prepare APPL1SRC by entering:

```
:COMPAPPL 1
```

Entering this UDC is the same as entering the following MPE commands:

```
:PURGE APPL1
:SPL   APPL1SRC, APPL1USL
:PREP  APPL1USL, APPL1; PMAP
:SAVE  APPL1
```

INCREASING UDC FUNCTIONALITY

UDCs can perform functions considerably more complex than those already covered in this section by including sophisticated command sequences within the body of the UDC. In the following example, the UDC Y2 contains the command sequence normally entered to initiate a remote session:

```
Y2 WHO=" USER/PASS.ACCT "
DSLINE Y2
REMOTE HELLO !WHO      ** Remote logon to Y2. **
REMOTE
```

Using UDCs in this way can save you from typing frequently used command sequences. Using regular characters, control characters and escape sequences in UDCs allows you to build and control inverse video fields on the terminal. Softkey and terminal key functions can also be manipulated with advanced UDCs. The terminal keys and softkeys can then be dynamic throughout your session. If you use Editor functions, for example, you can establish a UDC to program the softkeys to execute Editor commands.

USING UDCs WITH JOB CONTROL WORDS

UDCs may be used with Job Control Words (JCWs) to further increase UDC functionality. UDCs that use JCWs within their structure, for example, are useful for monitoring and maintaining the status of your session. The following example of an advanced UDC expands upon the idea of using UDCs to initiate remote sessions. It incorporates a JCW, Y1JCW, which has been previously set to zero in a logon UDC, to first check whether you already have a remote session.

```
CHECKY1 WHO ="USER.ACCT"        ** Check on current value of YJCW1. **
IF Y1JCW = 0 THEN

   DSLINE Y1                    ** Remote logon. **
   REMOTE HELLO !WHO

   SETJCW Y1JCW = 1             ** Y1JCW is set to 1. **

ENDIF
**********
```

If you later tried to initiate a remote session on Y1 using the above UDCs, the value of the JCW Y1JCW would be 1. The IF block would not execute, and no new session would be initiated.

For more information on Job Control Words, refer to the :SETJCW and :SHOWJCW commands in Section II of this manual, as well as the MPE V Intrinsics Reference Manual (32002-90018).

TERMINALS SUPPORTED BY MPE V

APPENDIX
A

TERMINAL TYPES	DESCRIPTION
0	HP 2749B (ASR-33 EIA-compatible) Terminal (10 cps).
1	ASR-37 Teleprinter with Paper Tape Reader/Punch (10 cps).
2	ASR-35 EIA-compatible Terminal (10 cps).
3	Execuport 300 Data Communications Transceiver Terminal (10/15/30 cps).
4*	HP 2600A or Datapoint 3300 Keyboard Display Terminal (10/15/30/60/120/240 cps).
5	Memorex 1240 Communications Terminal (10/15/30/60 cps). NOTE: This terminal must be eqipped for ECHO PLEX.
6*	HP 2762A/B (General Electric Terminet 300 or 1200), or Data Communications terminal, Model B (10/15/30/120 cps) with Paper Tape Reader/Punch, Option 2. NOTE: This terminal must be equipped for ECHO PLEX.
9*	HP 2165A Terminal (Beehive MiniBee) (10/15/30/60/120/240 cps).
10*	HP 2382A, 39x, 262x, and 264x or HP Personal Computer running on Terminal Emulation Mode.
11*	HP 2640A/B, HP 2641A, HP 2644A or HP 2645A. Allows user to use block mode without program control of block mode transmission. Requires user to position cursor before pressing ENTER. Recommended for speeds exceeding 30 cps when you expect to switch between character mode and block/line mode. May not be used in block/page mode.
12*	HP 2645K Katakana/Roman Data Terminal.
13*	Message switching network or other computer.
14	Multipoint Terminal.
15*	HP 2635A Printing Terminal. 8-bit protocol (for second character set).
16*	HP 2635A Printing Terminal. 7-bit protocol (standard character set).
18*	Non-HP devices. Only provides XON/XOFF protocol data-handling. All application printers.

* = Terminal Types supported with ADCC (Series 30/33).

TERMINAL TYPES	DESCRIPTION
19*	HP 2631B or HP 2613B-compatible Remote Spooled Printers.
20*	8-bit Spooled Serial Printer (not supported with subtype 15). Not supported with ATC (Series II/III).
21*	Spooled Serial Printer with embedded escape sequences allowed. HP2631B or HP2631B-compatible printers. Not supported with ATC (Series II/III).
22	8-bit Spooled Serial Printer with embedded escape sequences allowed (not supported with subtype 15). HP 2631B or 2631B-compatible printers. Not supported with ATC (Series II/III).

* = Terminal Types supported with ADCC (Series 30/33).

For further information refer to the Fundamental Communications Handbook (5957-4634).

SUBSYSTEM FORMAL FILE DESIGNATORS

APPENDIX B

COMMAND	PARAMETER	FORMAL FILE DESIGNATOR
:BASIC	*commandfile* *inputfile* *listfile*	BASCOM BASIN BASLIST
:BASICOMP	*textfile* *uslfile* *listfile*	BSCTEXT BSCUSL BSCLIST
:BASICGO	*textfile* *listfile*	BSCTEXT BSCLIST
:BASICPREP	*textfile* *listfile*	BSCTEXT BSCLIST
:BBASIC	*commandfile* *inputfile* *listfile*	BASCOM BASIN BASOUT
:BBASICOMP	*infile* *uslfile* *listfile*	BBCIN BBCUSL BBCLIST
:BBASICGO	*infile* *listfile*	BBCIN BBCLIST
:BBASICPREP	*infile* *listfile*	BBCIN BBCLIST
:COBOL	*textfile* *uslfile* *listfile* *masterfile* *newfile*	COBTEXT COBUSL COBLIST COBMAST COBNEW
:COBOLGO	*textfile* *listfile* *masterfile* *newfile*	COBTEXT COBLIST COBMAST COBNEW
:COBOLPREP	*textfile* *listfile* *masterfile* *newfile*	COBTEXT COBLIST COBMAST COBNEW

COMMAND	PARAMETER	FORMAL FILE DESIGNATOR
:COBOLII	*textfile* *uslfile* *listfile* *masterfile* *newfile* *workspacename*	COBTEXT COBUSL COBLIST COBMAST COBNEW COBWKSP
:COBOLIIGO	*textfile* *listfile* *masterfile* *newfile* *workspacename*	COBTEXT COBLIST COBMAST COBNEW COBWKSP
:COBOLIIPREP	*textfile* *progfile* *listfile* *masterfile* *newfile* *workspacename*	COBTEXT COBPROG COBLIST COBMAST COBNEW COBWKSP
:EDITOR	*listfile*	EDTLIST
:FORTRAN	*textfile* *uslfile* *listfile* *masterfile* *newfile*	FTNTEXT FTNUSL FTNLIST FTNMAST FTNNEW
:FORTGO	*textfile* *listfile* *masterfile* *newfile*	FTNTEXT FTNLIST FTNMAST FTNNEW
:FORTPREP	*textfile* *listfile* *masterfile* *newfile*	FTNTEXT FTNLIST FTNMAST FTNNEW
:FTN77 (FORTRAN 77)	*textfile* *uslfile* *listfile*	FTNTEXT FTNUSL FTNLIST
:FTNGO	*textfile* *listfile*	FTNTEXT FTNLIST
:FTNPREP	*textfile* *progfile* *listfile*	FTNTEXT FTNPROG FTNLIST
:PASCAL	*textfile* *uslfile* *listfile*	PASTEXT PASUSL PASLIST

COMMAND	PARAMETER	FORMAL FILE DESIGNATOR
:PASCALGO	*textfile* *listfile*	PASTEXT PASLIST
:PASCALPREP	*textfile* *listfile*	PASTEXT PASLIST
:PREP :PREPRUN	PMAP	SEGLIST
:PREPRUN :RUN	LMAP	LOADLIST
:RESTORE	SHOW	SYSLIST
:RPG	*textfile* *uslfile* *listfile* *masterfile* *newfile*	RPGTEXT RPGLIST RPGMAST RPGNEW
:RPGGO	*textfile* *listfile* *masterfile* *newfile*	RPGTEXT RPGLIST RPGMAST RPGNEW
:RPGPREP	*textfile* *listfile* *masterfile* *newfile*	RPGTEXT RPGLIST RPGMAST RPGNEW
:SEGMENTER	*listfile*	SEGLIST
:SPL	*textfile* *uslfile* *listfile* *masterfile* *newfile*	SPLTEXT SPLUSL SPLLIST SPLMAST SPLNEW
:SPLGO	*textfile* *listfile* *masterfile* *newfile*	SPLTEXT SPLLIST SPLMAST SPLNEW
:SPLPREP	*textfile* *listfile* *masterfile* *newfile*	SPLTEXT SPLLIST SPLMAST SPLNEW

COMMAND	PARAMETER	FORMAL FILE DESIGNATOR
:STORE :RESTORE	SHOW	SYSLIST
:VINIT	*listfile*	VINLIST

INDEX

INDEX

C

INDEX

Q

R

S

T

U

V

W

READER COMMENT SHEET

HP 3000 Computer Systems

MPE V/U COMMANDS
Reference Manual

32033-90006 Feb 1986

We welcome your evaluation of this manual. Your comments and suggestions help us to improve our publications. Please explain your answers under Comments, below, and use additional pages if necessary.

Is this manual technically accurate? ☐ **Yes** ☐ **No**

Are the concepts and wording easy to understand? ☐ **Yes** ☐ **No**

Is the format of this manual convenient in size, arrangement, and readability? ☐ **Yes** ☐ **No**

Comments:

This form requires no postage stamp if mailed in the U.S. For locations outside the U.S., your local HP representative will ensure that your comments are forwarded.

FROM: **Date** _____

Name _____

Company _____

Address _____

_____

_____

FOLD FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 718 CUPERTINO, CA

POSTAGE WILL BE PAID BY ADDRESSEE

Publications Manager
Hewlett-Packard Company
Information Technology Group
19447 Pruneridge Ave.
Cupertino, Ca 95014

FOLD FOLD

Part No. 32033-90006
Printed in U.S.A.
E0286

HEWLETT PACKARD