MPE/iX Commands Reference Manual

HP 3000 MPE/iX Computer Systems

Edition 10

HEWLETT® PACKARD

32650-90864

E1098

Printed in: U.S.A. October 1998

Notice

The information contained in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose. Hewlett-Packard shall not be liable for errors contained herein or for direct, indirect, special, incidental or consequential damages in connection with the furnishing or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

This document contains proprietary information which is protected by copyright. All rights reserved. Reproduction, adaptation, or translation without prior written permission is prohibited, except as allowed under the copyright laws.

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013. Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19 (c) (1,2).

Hewlett-Packard Company
3000 Hanover Street
Palo Alto, CA 94304 U.S.A.

4. Command Definition F-K

5. Command Definitions L-O

6. Command Definitions P-R

7. Command Definitions S-SO

Preface

The ninth edition of the *MPE/iX Commands Reference Manual* is one volume with command descriptions alphabetically from A through X. This manual is written for all users of the HP 3000 MPE/iX Computers. MPE/iX, Multiprogramming Executive with Integrated POSIX, is the latest in a series of forward-compatible operating systems for the HP 3000 line of computers.

In HP documentation and in talking with HP 3000 users, you will encounter references to MPE XL, the direct predecessor of MPE/iX. MPE/iX is a superset of MPE XL. All programs written for MPE XL will run without change unde MPE/iX. You can continue to use MPE XL system documentation, although it may not refer to features added to the operating system to support POSIX (for example, hierarchical directories).

Finally, you may encounter references to MPE V, which is the operating system for HP 3000s, not based on the PA-RISC architecture. MPE V software can be run on the PA-RISC (Series 900) HP 3000s in what is known as compatibility mode.

The *MPE/iX Commands Reference Manual* is organized into eight chapters. A description of each chapter follows:

Chapters 2 thru 8 of this manual, explains the purpose, syntax, parameters, and operation (including examples) for each MPE/iX command. If you know which command to use for the task you wish to perform, turn directly to that command definition in chapter 2. For your convenience, commands are organized alphabetically. If you don't know which command to use, Chapter 1, which contains a listing of commands by task, is a good place to start. After reading the brief description, turn to the appropriate command definition.

For supplemental information on command syntax and use, refer to the *MPE/iX General User's Reference Manual*.

Chapter 1 **Commands by Task** provides an introduction to all MPE/iX commands and their functions, categorized by the task they perform.

Chapters 2-8 **Command Definitions** provides documentation on each MPE/iX command alphabetically for your ease of use.

Conventions

The following conventions are used throughout this manual.

italics In a syntax statement or an example, a word in intalics represents a parameter or argument that you must replace with the actual value. In the following example, you must replace filename with the name of the file:

COMMAND *filename*

{ } In a syntax statement, braces enclose required elements. When several elements are included within braces, you must select one. In the following example, you must select either `ON` or `OFF`

```
COMMAND {ON | OFF}
```

[ ] In a syntax statement, brackets enclose optional elements. In the following example, `OPTION` can be omitted:

COMMAND filename [OPTION]

When several elements are enclosed within brackets, you canm select one or none of the elements. In following example, you can select OPTION or parameter or neither. The elements cannot be repeated.

COMMAND filename [OPTION | parameter]

[...] In a syntax statement, horizontal ellipses enclosed in brackets indicate that you can repeatedly select the element(s) that appear within the immediately preceding pair of brackets or braces. In the example below, you can select parameter zero or more times. Each instance of parameter must be preceded by a comma:

[,parameter][...]

In the example below, you can only use the commas as a delimiter if papameter is repeated; no comma is used before the first occurrence of papameter:

[parameter][,...]

| ... | In a syntax statement, horizontal ellipses enclosed in vertical bars indicate that you can select more that one element within the immediately prededing pair of brackets or braces. However, each particular element can only be selected once. In the followig example you must select A, AB, BA, or B. The elements cannot be repeated.

[A | B] |...|

1 Commands by Task

Commands are used to communicate with the MPE/iX operating system. They request MPE/iX to perform a specific task or provide specific information.

Task-Related Commands

This chapter is an introduction to MPE/iX commands and their functions, categorized by the task they perform.

The categories of tasks identified for MPE/iX commands are:

- Accessing Subsystems and Utilities.
- Command Interpreter Programming Tools.
- Communicating with Other Users.
- Executing User Programs.
- Managing Accounts, Groups, and Users.
- Managing Devices.
- Managing Files.
- Managing Jobs and Sessions.
- Managing Spooler Operations.
- Managing System Resources.
- Managing User/System Logging.
- Managing Variables and Job Control Words.
- Managing Volumes (Disk Drives).
- Using Command Files and User-Defined Commands.
- Using Computer Language Programs.

To use this chapter, first determine what task you want to perform, for example, create a new account. Check the list above to find an appropriate category, which in this case would be "Managing Accounts, Groups, and Users". Turn to that category and you will find a list of MPE/iX commands that perform tasks related to managing accounts and a description of the particular function for each command.

Check that list to find an appropriate functional description, which in this case is "Creates a new account". Then check the lefthand column for the name of the command that performs that function, which in this case is NEWACCT.

When you have located the command that most closely performs the task you want to accomplish, turn to chapter 2 of this manual for complete information about the syntax, parameters, operation, use, and examples for that command. For your convenience, the commands in chapter 2 are listed in alphabetical order.

Command **Function**

Accessing Subsystems and Utilities

DEBUG	Instructs MPE/iX to enter the system debugger.
EDITOR	Starts the EDIT/3000 subsystem.
FCOPY	Runs the FCOPY subsystem
HELP	Accesses the help subsystem.
RESETDUMP	Disarms the debug facility call that is made during abnormal process termination.
SEGMENTER	Starts the MPE segmenter.
SETDUMP	Arms the system debug facility for a process abort.
SH	UDC that executes SH.HPBIN.SYS, the POSIX shell
SYSGEN	Starts configuration dialog and/or installation tape creation.
VOLUTIL	Managing user volume sets

Command Interpreter Programming

CALC or #	Evaluates an expression
COMMENT	Inserts a comment into a job stream or user command.
CONTINUE	Overrides a job error so that the job or user command continues executing.
ECHO	Displays a message on the terminal for a session or the printer for a job.
ELSE	Provides an alternate execution sequence within an IF statement.
ELSEIF	Provides an alternate execution sequence within an IF statement.
ENDIF	Terminates an IF block.
ENDWHILE	Terminates a WHILE block.
ESCAPE	Allows the CI programmer to simulate all aspects of CI error handling.
IF	Used to control the execution sequence of a job, UDC, or command file.
INPUT	Allows you to interactively assign a value to any variable that can be set with the SETVAR command.

RETURN	Causes execution to return from the current user command (UDC or command file) to the calling environment
SETVAR	Creates or modifies a CI variable.
WHILE	Used to control execution in a job, session, UDC, or command file.

Communicating with Other Users

TELL	Sends a message to another active session.
TELLOP	Sends a message to the system console.
WARN	Sends an urgent message to jobs/sessions.
WELCOME	Used to create the system welcome message.

Executing User Programs

LINK	Creates an executable program file.
OCTCOMP	Converts a compiled MPE V/E program into native mode code for the HP 3000 Series 900.
PREP	Prepares a compatibility mode program from a user subprogram library onto a program file.
PREPRUN	Prepares and executes a compiled compatibility mode program.
PROGRAM FILENAME	Executes a program. Note the filename may be qualified using the HPPATH CI variable
RUN	Executes a prepared or linked program.

Managing Accounts, Groups, and Users

ALTACCT	Changes the attributes of an existing account.
ALTGROUP	Changes the attributes of an existing group.
ALTUSER	Changes the attributes of an existing user.
LISTACCT	Displays information about a specified account(s).
LISTGROUP	Displays information about a specified group(s).
LISTUSER	Displays information about a specified user(s).
NEWACCT	Creates a new account.
NEWGROUP	Creates a new group.
NEWUSER	Creates a new user.
PURGEACCT	Removes an account from the system.
PURGEGROUP	Removes a group from the system.

PURGEUSER	Removes a user from an account.
REPORT	Displays accounting information about the logon account and group.

Managing Devices

ABORTIO =ABORTIO	Aborts a single pending I/O request for a device.
ASSOCIATE	Gives a user operator control of a device.
DEVCNTRL	Script that ejects a tape an/or sets a tape device online.
DISASSOCIATE	Removes control of a device from a user.
DOWN	Removes a device from normal system use.
DOWNLOAD	Downloads format information to a line printer.
HEADOFF	Stops header/trailer output to a device.
HEADON	Resumes header/trailer output to a device.
SET	Sets terminal and $STDLIST configuration.
SETMSG	Enables/disables receipt of user or operator messages on the terminal.
SHOWDEV	Reports the status of input/output devices.
SHOWIN	Reports the status of input device files.
SHOWOUT	Displays the status of output device files.
SPEED	Sets the input/output speed for a terminal.
STREAMS	Enables/disables the STREAMS device allowing users to submit job/data streams to a designated device.
UP	Returns a device (except disk drives) stopped with a DOWN command to normal function on the system.

Managing Files

ALTFILE	Changes a file’s owner or group ID
ALTSEC	Changes a file's security provisions.
BUILD	Creates and allocates a new empty file on disk.
CHDIR	Changes the current working directory.
COPY	Copies one file to another file.
DATA	Enters data into the system from a device file.
FILE	Declares file attributes for a file when it is opened.
LISTEQ	Displays all active file equations for a job or session.
LISTF	Displays information about permanent files.

LISTFILE Lists file information using native mode scanning/parsing that can be easily expanded.

LISTFTEMP Displays information about temporary files.

NEWDIR Creates a directory

NEWLINK Creates a symbolic link

PRINT Prints the contents of a file.

PURGE Deletes a file from the system.

PURGEDIR Deletes a directory.

PURGELINK Deletes a symbolic link, empty directory, or a regular file.

RELEASE Removes all security provisions for a file.

RENAME Changes the name of a file.

RESET Cancels file equations.

RESTORE Returns files stored on tape to the system.

SAVE Saves a file in the permanent system file domain.

SECURE Restores security provisions for a file.

STORE Copies disk files onto magnetic tape for storage.

VSTORE Verifies data on native mode backup media and reports errors incurred by STORE when writing the tape.

Managing Jobs and Sessions

ABORT Aborts the current program or operation.

ABORTJOB =ABORTJOB Aborts a job or session.

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

ALTJOB Alters the attributes of waiting or scheduled jobs.

BREAKJOB Suspends an executing job.

BYE Ends an interactive session.

CHDIR Changes the current working directory.

CHGROUP Switches user from current group to another group within the logon account.

DO Used to reexecute commands in the command line history stack.

:EOD Denotes end-of-data on the input stream from a job file or terminates data initialized by the DATA command.

EOJ Ends a batch job.

EXIT Terminates the command interpreter.

HELLO Initiates an interactive session.

JOB Defines a job to be activated in conjunction with the STREAM command to run in batch mode.

JOBFENCE Defines the minimum input priority a job or session must have in order to execute.

JOBPRI Sets or changes the default and/or maximum execution priority for batch jobs.

JOBSECURITY Designates what level of user may request resources and control the execution of jobs.

LIMIT Limits the number of concurrently running jobs/sessions for the entire system or for individual job queues.

LISTJOBQ Lists a job queue

LISTREDO Displays the contents of the command line history stack.

=LOGOFF Aborts all executing jobs/sessions and prevents any further logons.

=LOGON Enables job/session processing following a =LOGOFF command.

NEWJOBQ Creates a new job queue

PAUSE Sleep for a specified number of seconds or until a job(s) terminates.

PURGEJOBQ Deletes a job queue

REDO Used to edit/reexecute commands in the command line history stack.

REFUSE Disables jobs/sessions and/or data on a designated device.

RESUME Resumes execution of a suspended operation.

RESUMEJOB Resumes a suspended job.

SHOWJOB Displays status information about jobs/sessions.

SHOWME Reports job/session status.

SHOWTIME Displays the current time and date.

STARTSESS Creates a session on the specified device for a user with programmatic sessions (PS) capability.

STREAM Spools batch jobs or data from a session or job.

Managing Spooler Operations

ALTSPOOLFILE Alters the characteristics of an output spoolfile.

DELETESPOOLFILE Deletes a spoolfile from disk.

LISTSPF Produces a listing of spooled files, both input and output.

OPENQ Opens the spool queue for a specified logical device or device class.

OUTFENCE Defines the minimum priority an output spoolfile needs in order to be printed.

RESUMESPOOL Resumes suspended spooler output to a spooled device.

STARTSPOOL Initiates the spooler process for a device.

SHUTQ Closes the spool queue for a specified logical device or device class.

SPOOLER Controls spooler processes.

SPOOLF Allows a qualified user to alter, print, or delete output spoolfiles.

STOPSPOOL Terminates spooling to a specified device or device class.

SUSPENDSPOOL Suspends output to a spooled device.

Managing System Resources

ALLOCATE Loads a compatibility mode program or procedure into main memory.

ALLOW Grants a user access to a specific operator command.

ALTPROC Changes the priority for the specified processes.

CONSOLE Changes the system console from its current device to another job-accepting terminal.

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

DISALLOW Prohibits access to a specific operator command.

DISCRPS Enables/disables rotational position sensing on a specified logical device.

ERRDUMP Dumps a process or system error stack.

FREERIN Releases a global resource identification number (RIN).

GETRIN Acquires and assigns a password to a global resource identification number (RIN).

PAUSE Suspends current activity for a specified number of seconds.

RECALL =RECALL Displays all pending console REPLY messages.

REPLY =REPLY Replies to pending resource request messages that require a response.

RESETACCT Resets the system counters for CPU-time or connect-time, used by an account and its groups, to zero.

SETCLOCK Sets the system clock.

SETCOUNTER Sets the next value of a resource counter.

SHOWALLOW Displays allowed operator commands.

SHOWCLOCK Displays information about the system date and time.

SHOWPROC Displays information about one or more processes.

SHOWQ Displays process scheduling data and the contents of each subqueue.

=SHUTDOWN Initiates a shutdown of MPE/iX.

TUNE Alters the dispatcher subqueues which determine when processes must relinquish the CPU.

Managing User/System Logging

ALTLOG Alters the attributes of an existing user logging identifier.

CHANGELOG Changes the user logging file without stopping or interrupting the logging process.

GETLOG Establishes a logging identifier on the system.

LISTLOG Lists active logging identifiers and whether automatic log file changing has been enabled.

LOG Starts, restarts, or stops user logging.

RELLOG Removes a user logging identifier from the system.

RESUMELOG Resumes system logging following suspension caused by an error.

SHOWLOG Displays the number of the system's current log file and the percentage of disk space used.

SHOWLOGSTATUS Displays status information about opened user logging files assigned to a logging identifier.

SWITCHLOG Closes the current system log file, then creates and opens a new one.

Managing Variables and Job Control Words

DELETEVAR Deletes one or more MPE/iX variables.

ERRCLEAR Zeros out all HP predefined error-related variables.

INPUT Allows you to interactively assign a value to any variable that can be set with the SETVAR command.

SETJCW Creates or assigns a value to a job control word (JCW) variable.

SETVAR Assigns values to MPE/iX variables.

SHOWJCW Displays the current status of job control word variables.

SHOWVAR Displays current values for specific variables.

Managing Volumes (Disk Drives)

DSTAT Displays current status of system disk drives.

DISMOUNT Causes a volume set that was explicitly reserved by a user to be released.

LDISMOUNT	Causes a volume set that was reserved system-wide by the user to be released.
LMOUNT	Reserves a volume set system-wide.
MOUNT	Reserves an online volume set.
VMOUNT	Enables/disables the MPE/iX movable volume facility.
VOLTIL	Defragment diskspace, general user volume management
VSCLOSE	Closes a specified volume set and takes it offline.
VSOPEN	Reopens a volume set closed with VSCLOSE.
VSRELEASE	Releases a volume set that was explicitly reserved by the user with VSRESERVE.
VSRELEASESYS	Cancels a previously issued VSRESERVESYS command for a specified volume set.
VSRESERVE	Reserves a particular volume set online.
VSRESERVESYS	Reserves a volume set online system-wide.
VSUSER	Lists all users of a currently reserved, mountable volume set.

Using Command Files and User-Defined Commands

ANYPARM	Define a parameter that accepts all characters without the need for quotes.
ESCAPE	Allows the CI programmer to simulate all aspects of CI error handling.
OPTION	Modifies the environment of user-defined commands and command files.
PARM	Defines a parameter for a UDL or command file.
RETURN	Used in user command files to return execution to the calling environment.
SETCATALOG	Specifies a file containing user-defined commands.
SHOWCATALOG	Displays information about user-defined commands (UDCs).
XEQ	Executes a program or command file.

Using Computer Language Programs

BASIC	Interprets a compatibility mode BASIC/V program.
BASICGO	Compiles, prepares, and executes a compatibility mode BASIC/V program.
BASICOMP	Compiles a compatibility mode BASIC/V program.
BASICPREP	Compiles and prepares a compatibility mode BASIC/V program.
BBASIC	Starts execution of the HP Business BASIC/V interpreter in compatibility mode.
BBASICGO	Compiles, prepares, and executes an HP Business BASIC/V program in compatibility mode.

`BBASICOMP` Compiles an HP Business BASIC/V program in compatibility mode.

`BBASICPREP` Compiles and prepares an HP Business BASIC/V program in compatibility mode.

`BBXL` Initiates execution of the HP Business BASIC/XL interpreter.

`BBXLCOMP` Compiles an HP Business BASIC/XL program.

`BBXLGO` Compiles, links, and executes an HP Business BASIC/XL program.

`BBXLLK` Compiles and links an HP Business BASIC/XL program.

`CCXL` Compiles an HP C/iX program.

`CCXLGO` Compiles, links, and executes an HP C/iX program.

`CCXLLK` Compiles and links an HP C/iX program.

`COB74XL` Compiles an HP COBOL II/XL program using the 1974 ANSI standard entry point and creates an object file.

`COB74XLG` Compiles, links, and executes an HP COBOL II/XL program using the ANSI 1974 standard entry point.

`COB74XLK` Compiles and links an HP COBOL II/XL program using the 1974 ANSI standard entry point.

`COB85XL` Compiles an HP COBOL II/XL program using the 1985 ANSI standard entry point and creates an object file.

`COB85XLG` Compiles, links, and executes an HP COBOL II/XL program using the ANSI 1985 standard entry point.

`COB85XLK` Compiles and links an HP COBOL II/XL program using the 1985 ANSI standard entry point.

`COBOLII` Compiles a compatibility mode COBOLII program on the COBOL 74 compiler.

`COBOLIIGO` Compiles, prepares, and executes a compatibility mode COBOLII program on the COBOL 74 compiler.

`COBOLIIPREP` Compiles and prepares a compatibility mode COBOLII program on the COBOL 74 compiler.

`FORTGO` Compiles, prepares, and executes a compatibility mode FORTRAN 66/V program.

`FORTPREP` Compiles and prepares a compatibility mode FORTRAN 66/V program.

`FORTRAN` Compiles a compatibility mode FORTRAN 66/V program.

`FTN` Compiles a compatibility mode HP FORTRAN 77/V program.

`FTNGO` Compiles, prepares, and executes a compatibility mode HP FORTRAN 77/V program.

`FTNPREP` Compiles and prepares a compatibility mode HP FORTRAN 77/V program.

`FTNXL` Compiles an HP FORTRAN 77/iX program.

`FTNXLGO` Compiles, links, and executes an HP FORTRAN 77/iX program.

`FTNXLLK` Compiles and links an HP FORTRAN 77/iX program.

`PASCAL` Compiles a compatibility mode Pascal/V program.

`PASCALGO` Compiles, prepares, and executes a compatibility mode Pascal/V program.

`PASCALPREP` Compiles and prepares a compatibility mode Pascal/V program.

`PASXL` Compiles an HP Pascal/iX program.

`PASXLGO` Compiles, links, and executes an HP Pascal/iX program.

`PASXLLK` Compiles and links an HP Pascal/iX program.

`RPG` Compiles an RPG/V program in compatibility mode.

`RPGGO` Compiles, prepares, and executes an RPG/V program in compatibility mode.

`RPGPREP` Compiles and prepares an RPG/V program in compatibility mode.

`RPGXL` Compiles an RPG/XL program.

`RPGXLGO` Compiles, links, and executes an RPG/XL program.

`RPGXLLK` Compiles and links an RPG/XL program.

`SPL` Compiles a compatibility mode SPL/V program.

`SPLGO` Compiles, prepares, and executes a compatibility mode SPL/V program.

`SPLPREP` Compiles and prepares a compatibility mode SPL/V program.

2 Command Definitions A-B

This chapter provides information on MPE/iX commands. For your convenience, they are arranged in alphabetical order. Each command specification contains the following information:

Command Name Provides the command name at the top of each page followed by a brief definition of its function.

Syntax Provides information in diagram format defining how to enter the command and its parameters.

Parameters Provides an explanation of each parameter and its function, limitations, and defaults.

Operation Notes Provides an explanation of the operation of the command and notes on any special considerations.

Use Provides information on the conditions within which the command can be used such as a session, job, program, or in BREAK. This entry also indicates whether the command can be interrupted with the **Break** key and, if appropriate, lists any special capabilities required to use it. Refer to the NEWACCT command for a list of special capabilities.

Examples Provides examples of how to use the command.

Related Information Provides pointers to other commands or manuals that might contain additional information.

Commands and Parameters

MPE/iX commands tell the computer to perform a specific function. The parameters you enter for each command tell the computer to perform the function in a specific way. MPE/iX uses four classifications of parameters:

- Required parameters.
- Optional parameters.
- Keyword parameters.
- Positional parameters.

These four classifications of parameters are briefly defined below. To understand the command syntax diagrams, refer to the Conventions pages in the front of this manual.

Required Parameters

If a command has any required parameters, they must be entered or MPE/iX displays an error message. In the syntax diagrams for each command in this chapter, required parameters are either surrounded by no other marks or by braces `{ }`. In the following example, since *myfile* is not surrounded by any marks, it is a required parameter:

```
BUILD myfile
```

In some cases, you must select one parameter from a list of two or more parameters. In the following example, you must provide either a job number or a session number since these parameters are surrounded by braces:

```
ALTJOB [#Jnnn | #Snnn]
```

Optional Parameters

If a command has any optional parameters, you can either specify or ignore them, depending upon how you want the command to execute. In the syntax diagrams for each command in this chapter, optional parameters are surrounded by brackets `[ ]`. If you ignore optional parameters, MPE/iX uses the system-defined default values for each parameter. In the following example, `[;PASS]` is an optional parameter since it is surrounded by brackets:

```
NEWGROUP groupname [;PASS=[password]]
```

Positional Parameters

The meaning of a positional parameter depends upon its position (location) in the parameter list. In the syntax diagrams for each command in this chapter, positional parameters are separated from each other by a comma (`,`). If you omit a positional parameter from the list, you must provide the comma placeholder that would normally precede that parameter. In the following example, the subparameters of the `REC` parameter of the `BUILD` command can be treated as positional parameters:

```
BUILD filename;REC=128,1,F,ASCII
```

If you choose to use the system-defined default value `F`, you need not specify it, but you must hold the position with a comma:

```
BUILD filename;REC=128,1,,ASCII
```

Keyword Parameters

A keyword parameter denotes the meaning or value of a given parameter. In the syntax diagrams for each command in this chapter, keyword parameters appear in uppercase (`CAPITAL`) letters (although you may enter them in either uppercase or lowercase) and are preceded by a semicolon (`;`). In the following example, `REC` is a keyword parameter:

```
BUILD filename;REC=128,1,F,ASCII
```

Refer to the section "Combining Positional/Keyword Parameters," below, for additional information.

Native Mode Command Structure

Many commands in this chapter have the designation **Native Mode** at the end of their definition. This means that the command is parsed by the **Native Mode Command Parser**. If **Native Mode** is not specified, the command is parsed by the **Compatibility Mode Command Parser**. (A command *parser* separates command parameters.) There is no relationship between the parser a command uses and the function(s) the command performs. Also, just because a command is parsed by the **Compatibility Mode** parser does not mean it functions in the same way it did in the Classic HP3000 environment.

All new commands for MPE/iX use the **NM** parser. Some commands used on MPE V/E which have been changed for MPE/iX use the **NM** parser and some do not. MPE V/E commands which have not been changed for MPE/iX generally use the **CM** parser.

The important thing to remember is that the Native Mode parser accepts several different formats for commands that you enter at the colon prompt (`:`). You may enter these NM-parsed commands in one of the following ways:

- By using the formal command specification shown in the syntax diagram for each command in this chapter.
- By using positional parameter specifications to enter keyword parameter values.
- By combining positional and keyword specification

Another difference between the NM parser and the CM parser is that the CM parser restricts a single comand parameter value to be <=255 characters. On the NM side, the value is limited by the size of the CI's command buffer.

Formal Command Specification

You may enter an NM-parsed command as shown in the syntax diagram for each command, for example:

```
COMMAND KEYWORD1=A;KEYWORD2=B;KEYWORD3=C
```

Positional Parameter Specification

You may also enter an NM-parsed command by omitting the keyword parameter name and only entering the values as positional parameters, for example:

```
COMMAND A,B,C
```

If you omit the keyword specifications and enter the values as positional parameters, the values must be treated as such, and all rules for positional parameters must be followed. For example, if you only specify `A` and `C`, you must use the positional place holder (`,`) as shown in the following example:

```
COMMAND A,,C
```

Combining Positional/Keyword Parameters

Another option is to enter NM-parsed commands by using a combination of positional and keyword specifications, for example:

```
COMMAND A,B;KEYWORD3=C
```

There is one important rule to remember when you combine positional and keyword parameters: once you specify a keyword parameter, you may no use positional parameters. For example, entering the following command would produce an error:

```
COMMAND A;KEYWORD2=B,C
```

An exception to the rule is that you may specify positional parameters that are *subparameters* of a keyword parameter. For example, in the BUILD command shown below, REC is a keyword but the next four parameters (which define records as being 80 bytes long, blocked at 1 and in Fixed ASCII format) are *positional*. This syntax is acceptable because they are *subparameters* of the key word REC.

BUILD ***filename*****;REC=-80,1,F,ASCII**

The following example shows the correct way to combine positional and keyword parameters where the keyword has no subparameters:

```
COMMAND A;KEYWORD2=B;KEYWORD3=C
```

Entering Numbers in Commands

You may enter numbers as parameters to NM-parsed commands as follows:

- With or without leading zeros.
- As positive or negative numbers.
- Preceded by the $ sign indicating hexadecimal or base 16.
- Precdeded by the % sign indicating octal or base 8.
- Preceded by the # sign indicating decimal or base 10 (if neither $, % nor # is specified base 10 is used).
- In the decimal range -2,147,483,648 to 2,147,483,647.

For example, suppose you wanted to *suspend* spooling on LDEV 6, your system printer. You could enter :

```
SPOOLER DEV=#0006;SUSPEND;SHOW
```

Or, because *decimal* is the default you could omit the # sign and enter:

```
SPOOLER DEV=0006;SUSPEND;SHOW
```

Or, omitting the leading zeroes you could enter:

```
SPOOLER DEV=6;SUSPEND;SHOW
```

When entering numbers as command parameters, it is advisable to omit leading zeros for some commands parsed by the compatibility mode (CM) parser.

Using Quotes and Strings

The NM parser optionally accepts any *string* input in single or double quotes. For example, because the file name parameter of the `PRINT` command is a *string* parameter, you could enter it as follows:

```
PRINT FILENAME
```

or

```
PRINT "FILENAME"
```

or

```
PRINT 'FILENAME'
```

General Rules for Using Quotes

Quotes are *required* if the value of any string parameter contains any of the following *delimiters*:

,	comma
;	semicolon
	blank (one or more spaces)
=	equal sign
(	left parentheses
)	right parentheses

For example, suppose you want to set a variable called MYVAR to a value of `;(A)`. Because this string contains both a semi-colon and parentheses, you would enter `SETVAR` as follows:

```
SETVAR MYVAR ";(A)"
```

As another example, suppose you wanted to use the `INFO=` parameter of the `RUN` command to pass the following string (which contains both commas and spaces) `BLUE RIGHT 24, SPLIT LEFT, 2`. You would enter:

```
RUN PROG;INFO="BLUE RIGHT 24, SPLIT LEFT, 2"
```

String Processing

MPE/iX *string processing* finds the first double or single quote and pairs it with the last quote of the same *type* to form a string. In other words, single quotes pair only with other single quotes and double quotes only with other double quotes. For this reason you can use single quotes within double quotes, and double quotes within single quotes. For example, all three of the following `INFO` strings are correct:

```
...;INFO="THIS IS THE 'WRITE' WAY"
...;INFO='THIS IS THE "WRITE" WAY TOO'
...;INFO="YOU SIMPLY CAN'T GO WRONG"
```

In all of the above cases, the quotes around the word WRITE and in the word CAN'T are contained within the string and are treated just like any other character.

Quotes within Strings

A technique called *quote folding* enables you to embed single or double quotes in quoted strings. For example, the following `INFO=` string would pass the string shown below it:

```
... ;INFO="JUST SAY ""GATO""."
```

```
JUST SAY "GATO".
```

Quote folding works as follows: When the NM parser reads a quote (other than the very first quote in an entire line), it checks the character to the immediate right of the quote. If it is a quote of the same kind (single or double) it is disregarded and the previous quote is treated like any other non-quote character. For example, after being parsed, the following quoted string becomes the string listed below it:

```
...;"PASS ""A"" TO ""X"" AND ""B"" TO ""Y"""
```

```
PASS "A" TO "X" AND "B" TO "Y"
```

Here is another example:

```
 "HERE ARE FOUR QUOTES "" "" "" """
```

```
HERE ARE FOUR QUOTES " " " "
```

To delete spaces between the four quotes, you would enter the string like this:

```
 "HERE ARE FOUR QUOTES """""""""
```

After being parsed, the string would look like this:

```
HERE ARE FOUR QUOTES """"
```

The NM parser processes quoted strings in the same way regardless of the command or parameter with which they are used.

For *most* CM commands, the CM parser processes quoted strings in the same way as the NM parser. However, the CM parser limits the length of quoted strings to 255 characters.

Exceptions

There are four exceptions to the syntax governing MPE/iX commands:

- User command parameter lists (which may affect string quoting rules).

- The SETVAR command.
- The XEQ command.

These three exceptions allow the use of only specific delimiters when specifying parameters, as defined below.

Also the ECHO command accepts all delimiters and treats them as part of the value to be echoed

Invoking User Defined Commands

User defined commands may be structured to accept the KEYWORD=*parm* format, and you may mix keyword and positional parameters. User command parameter lists allow you to use the following to delimit parameters:

,	comma
;	semicolon
	blank (one or more spaces)
=	equal sign

For example, if the user defined command UDCA is defined as `UDCA parm1,parm2,parm3` you could invoke it as follows:

```
UDCA X;Y;Z
```

or

```
UDCA X PARM2=Y,PARM3=Z
```

If the *value* of any parameter contains any of the above *delimiters* you must use quotes to delimit the parameter string. For example, if I;J;K is a *single* string parameter value you must delimit it with quotes (because it contains semi-colons) as follows:

```
UDCA "I;J;K"
```

The = sign is used only to delimit a parameter *name* from a parameter *value*. If the value of a parameter contains an = sign, then you must delimit the value with quotes. For example:

```
UDCA PARM1="YES=OK"
```

Similarly, if a string value contains a quote, you must delimit it by a quote. As an example, suppose you have a UDC which runs a program with the INFO string. The RUN command within the UDC might look something like this:

```
RUN PROGNAME;INFO="!PARM"
```

If the *value* of the parameter were something like this: `THE "END" IS NEAR`, you would invoke the UDC like this:

```
UDCA PARM="THE ""END"" IS NEAR"
```

Or, you could enter this:

```
UDCA "THE ""END"" IS NEAR"
```

NOTE If a parameter value begins with a quote it *must* have a matching end quote. If it does not begin with a quote it may contain embedded quotes which will be treated as any other character.

For more information on the use of quotes, refer to the section "Using Quotes and Strings" earlier in this chapter.

The SETVAR Command

The SETVAR command allows you to use either spaces, semicolons, or commas to delimit parameters, as follows:

```
SETVAR NAME expression
SETVAR NAME,expression
SETVAR NAME;expression
```

The rules for using quotes within strings containing delimiters or quotes, previously discussed, apply to the SETVAR command.

For example, suppose you want to set a variable called BIGVAR to a value of X,"Y";Z. This expression contains two delimiters (comma and semicolon) as well as quotes. The correct SETVAR command would be:

```
SETVAR BIGVAR "X,""Y"";Z"
```

You could also delimit the expression from the variable name using either a comma or semicolon as follows:

```
SETVAR BIGVAR;"X,""Y"";Z"
```

```
SETVAR BIGVAR,"X,""Y"";Z"
```

For more information on the use of quotes, refer to the section "Using Quotes and Strings" earlier in this chapter.

The XEQ Command

The XEQ command allows you to use only spaces to delimit parameters, as follows:

```
XEQ filename [parameters]
XEQ cmdfile [parameters]
```

NOTE A leading semicolon is optional for the first keyword parameter supplied for most commands if it immediately follows the command name. For example, the two commands below are equally valid:

```
ALTJOB JOB=...
```

```
ALTJOB ;JOB=...
```

Remote Sessions and Command Intrinsics

When used to invoke commands on *remote* systems the `COMMAND` or `HPCICOMMAND` intrinsics do not return a meaningful status code. For more information on calling intrinsics refer to the MPE/iX Intrinsics Reference Manual.

Running the CI as a Program

The MPE/iX Command Interpreter (CI) is a Native Mode Program. You can run it the way you would any other program, either by explicitly using the `RUN` command (the first example below), or by using the the *implied* `RUN` (the second example):

```
RUN CI.PUB.SYS
CI
```

In the first case, the `RUN` command controls execution of the CI. For more information, refer to the `RUN` command in this chapter.

The second case, referred to as *implied* run, is limited to recognizing the `INFO=` and `PARM=` parameters. If you enter both `INFO=` and `PARM=`, `PARM=` goes into effect *after* the `INFO=` string is passed. The Table 2-1 shows the Parm= values.

Table 2-1 Parm= values for the CI

Parm	Action
0	UDC's are cataloged, the CI banner is displayed, and the WELCOME message is displayed. This is the default.
1	Same as 0, however the CI terminates after processing the *info* string; it terminates after the first command is executed if no *info* string is specified.
2	UDC's are cataloged, the CI banner is suppressed, and the WELCOME message is suppressed.
3	Same as 2, however the CI terminates after processing the *info* string; it terminates after the first command is executed if no *info* string is specified.
4	Logon UDC's are executed, UDC's are available, the CI banner is displayed, and the WELCOME message is displayed.
5	Same as 4, however the CI terminates after processing the *info* string; it terminates after the first command is executed if no *info* string is specified.
-1	UDC's are not cataloged, the CI banner is suppressed, and the WELCOME message is suppressed. This requires SM capability.
-2	Same as -1, however the CI terminates after processing the *info* string; it terminates after the first command is executed if no *info* string is specified. This requires SM capability.

NOTE Parm -1 and -2 can be defeated via a SYSGEN misc configuration setting.

ABORT

Aborts the current program or operation suspended by BREAK. (Native Mode)

Syntax

```
ABORT
```

Parameters

None.

Operation Notes

The ABORT command terminates a process that has been suspended by pressing the **Break** key. Programs do not terminate while critical system code is executing on their behalf, but terminate immediately following execution of that code.

The ABORT command is available only from a session and only during 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.

Use

This command may be issued from a session (in BREAK only). It is not available from a job or a program. Pressing **Break** has no effect on this command.

Example

To abort the current program or operation, press **Break**. When the colon prompt (:) appears, enter:

```
  ABORT
```

The system then displays the message PROGRAM ABORTED PER USER REQUEST and redisplays the colon prompt (:).

Related Information

Commands	RESUME, SETJCW
Manuals	None

ABORTIO/ =ABORTIO

Aborts a single pending I/O request for a device.

Syntax

ABORTIO ***ldev***

=ABORTIO ***ldev***

Parameters

ldev The logical device number of the device for which you intend to abort one pending I/O request.

Operation Notes

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

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

Devices that are job-accepting or data-accepting always have outstanding READ requests pending, due to the auto-recognition feature of MPE/iX. Use the ABORTIO command to clear these pending input requests.

Sometimes, you may need to clear all outstanding I/O requests to allow proper execution of other console commands including ABORTJOB, TAKE, DOWN, and REFUSE.

NOTE If the ABORTIO command is not effective from the system console, use the =ABORTIO command. (You can only issue the **CTRL A** =ABORTIO command from the physical console.) Use the =ABORTIO command only when you cannot execute the ABORTIO command.

Use

You may issue the ABORTIO command from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It is executable only from the console unless distributed to users with the ALLOW or ASSOCIATE command.

The =ABORTIO console command cannot be issued from a job.

Examples

To abort a pending I/O request for logical device 53, enter:

```
ABORTIO 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
```

Related Information

Commands	SHOWDEV
Manuals	*Performing System Operation Tasks*

ABORTJOB/ =ABORTJOB

Aborts a job or session.

Syntax

ABORTJOB{ #J*nnn* #S*nnn* [*jobname*,] *user*.*acct* }

=ABORTJOB{ #J*nnn* #S*nnn* [*jobname*,] *user*.*acct* }

Parameters

#J*nnn* A job number.

#S*nnn* A session number.

jobname The name of the job, as identified by the SHOWJOB command.

user A user name.

acct An account name.

Operation Notes

The ABORTJOB 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/iX selects which job/session to abort. Therefore, to exercise more precise control when aborting jobs or sessions, use the #J*nnn* or #S*nnn* form of the ABORTJOB command. Although the job/session is abnormally terminated, log records are issued, and CPU-times and connect-times are updated. Any I/O activity, such as printing or file storage, is 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 (in other words, the job is in the WAIT state), the entire file is deleted. If the ABORTJOB command is issued before the output spoolfile is complete, only that portion of the file already spooled is printed, along with an error message indicating that the job was aborted. If a request is pending at the system console, it is automatically terminated by the ABORTJOB/=ABORTJOB command and the following message appears on the system console:

```
time/#Snnn/pin/REQUEST REQUIRING OPERATOR REPLY FOR PIN #nn HAS BEEN ABORTED
```

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

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

The standard error message that appears when a request is manually terminated by entering Y in response to =REPLY (or REPLY) is displayed on the user's terminal:

```
SESSION ABORTED BY SYSTEM MANAGEMENT
```

The =ABORTJOB command may be used at the physical console if ABORTJOB is ineffective. Refer to the "Use" section of this command.

Use

You may issue this command from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It is executable only from the console unless it is distributed to users with the ALLOW command, or the JOBSECURITY command is set to LOW with AM or SM capability.

=ABORTJOB may be issued only from the console.

NOTE Users with AM capability may only abort jobs and sessions within their own account. Users with SM capability may abort jobs and sessions across accounts.

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
```

In both of the preceding examples, the LOGOFF ON LDEV # messages indicates that ABORTJOB command was successfully executed.

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
```

Related Information

Commands	ALTJOB, BREAKJOB, JOBFENCE, JOBSECURITY, RESUMEJOB, SHOWJOB, STREAM
Manuals	*Performing System Operation Tasks*

ACCEPT

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

Syntax

ACCEPT[JOBS | DATA] *,ldev*

Parameters

JOBS — The designated device recognizes the JOB and HELLO commands. The device must be interactive to support sessions.

DATA — The designated device recognizes the DATA command. Data-accepting devices are not supported.

NOTE If you omit both the JOBS and the DATA parameters, then both the JOB and HELLO commands, and the DATA command are allowed.

ldev — The logical device number of the device for which the JOB, HELLO, and/or DATA commands are being enabled.

Operation Notes

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/iX automatically scans 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 you explicitly specify the JOBS parameter, the ACCEPT command is not executed unless the device is configured as a default output device.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It is executable only from the console unless distributed to users with the ALLOW command.

Examples

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

```
ACCEPT 19
SHOWDEV 19

LDEV AVAIL  OWNERSHIP  VOLID ASSOCIATION
 19 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
```

Related Information

Commands	REFUSE
Manuals	*Introduction to MPE XL for MPE V System Administrators*
	System Startup, Configuration, and Shutdown Reference Manual

ALLOCATE

Loads a compatibility mode program or procedure into virtual memory.

Syntax

```
ALLOCATE [ PROCEDURE, | PROGRAM, ] name
```

Parameters

PROCEDURE The procedure in `SL.PUB.SYS` to be allocated. The default is `PROGRAM`.

PROGRAM The program file to be allocated. Default.

name The name of the program file or procedure to be allocated.

Operation Notes

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 does not increase execution speed but it does reduce the time it takes to load the program for execution.

CAUTION Use care in deciding which programs or procedures to load with the `ALLOCATE` command. The number of CST table entries is limited and, if the limit is exceeded, data may be lost.

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.

To issue the `ALLOCATE` command a user 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`.

NOTE Native mode (NM) and Compatibility Mode (CM) loader error messages are reported differently, allowing you to determine the system in which the error occurred.

NM Loader Error: *ErrMessage* (`LDRERR`*nnnn*)

CM Loader Error: *ErrMessage* (`LOAD ERR`*nnnn*)

Use

This command may be issued from a session or program. Pressing **Break** has no effect on this command. System supervisor (OP) capability is required to use this command.

In addition to comma (,) a semicolon (;) and equal sign (=) may be used as a delimiter.

Example

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

```
ALLOCATE PROCEDURE,PROC1
```

Program files residing in the nonsystem domain (a volume set) are not allocated. Attempts to do so result in a LOAD ERR 92 message.

Related Information

Commands	DEALLOCATE
Manuals	*Introduction to MPE XL for MPE V Programmers* (30367-60004)

ALLOW

Grants a user access to a specific operator command.

Syntax

ALLOW FILE=*formaldesignator*[;SHOW]

ALLOW[*@.@* | *user.@* | *@.acct* | *user.acct*

;COMMANDS=*command* [,*command*,...]

Parameters

formal- designator An ASCII file name, which may consist of one to eight alphanumeric characters, beginning with an alphabetic character. It may be fully or partially qualified and may be back-referenced in a file equation.

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 The names of those commands to which the user is granted access.

Operation Notes

The operator uses the ALLOW command to distribute specific operator commands to system users. ALLOW specifies which users may execute operator commands, and which commands they may execute.

You may specify an indirect file with the ALLOW command, or you may execute ALLOW in subsystem mode. Each of these is explained below.

Using an indirect file to allow commands

To allow commands via an indirect file, you create a file that contains records identifying the users and accounts to whom you are allowing operator commands, followed by the list of commands allowed.

Using an indirect file with the ALLOW command is particularly convenient for system administrators since, once you make the file, you can reuse it to disallow the set of commands (via the DISALLOW command) or to allow the same set of commands again.

Here is an example of an indirect file:

```
EDITOR
HP32201A.07.17 EDIT/3000 TUES, MAY 29, 1994, 5:08 PM
(C) HEWLETT-PACKARD CO. 1985
/ADD
1  SUSAN.PAYROLL;COMMANDS=ALTJOB,ALTSPOOLFILE
```

```
2  JOHN.ACCTNG;COMMANDS=ALTJOB,DELETESPOOLFILE
3  //
...
/KEEP ALLOWTMP
/E
```

Once you create an indirect file, you then issue the ALLOW command, using the ;SHOW parameter to display each command line as it is executed from the file. For example:

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

You may backreference the file with a file equation as follows:

```
FILE BACKF=ALLOWTMP
ALLOW FILE=*BACKF;SHOW
```

If the file has a lockword, enter it in the command line after the filename. For example, "ALLOWTMP/*password*".

Using ALLOW in subsystem mode

To use the ALLOW command in subsystem mode, following these steps:

1. Enter ALLOW, followed by **Return**.
2. At the prompt (>), enter all of the commands you want to allow.
3. When you finish, press **Return** and enter a colon : as the first character of the new line. (You may also type EXIT.)

You cannot use the FILE= parameter in subsystem mode. The ALLOW subsystem will terminate if it encounters an error.

You may allow commands only to users who are currently logged on unless you specify the @.@ option, which allows commands to *all* users. (Since this option has obvious disadvantages, you can remedy the situation by then issuing a DISALLOW command to disallow command use to selected users.)

Additional capabilities granted to a user are valid only for the duration of their current session. Once the user logs off, any special capabilities previously assigned are no longer applicable.

To determine which operator commands have been allowed globally (that is, using the @.@ construct), or to a specific user, use the SHOWALLOW command.

NOTE Do not confuse *console* commands which are NOT allowable with *operator* commands. Operator commands are used in the day-to-day operation of your system and are generally allowable. A console command must be executed on the actual system console and must be preceded by **cntl-A**. Some console commands have the same name as non-console commands, an example is RECALL, which may be executed on any device.

The following is a list of commands that may be allowed.

```
ABORTIO       HEADON      RESUMESPOOL
ABORTJOB      JOBFENCE    SHUTQ
ACCEPT        JOBSECURITY STARTSPOOL
ALLOW         LDISMOUNT   STOPSPOOL
ALTJOB        LIMIT       STREAMS
ALTSPOOLFILE  LOG         UP
```

```
CONSOLE           MRJECONTROL  VMOUNT
DELETESPOOLFILE   OPENQ        VSCLOSE
DISALLOW          OUTFENCE     VSOPEN
DISCRPS           REFUSE       VSRELEASESYS
DOWN              REPLY        VSRESERVESYS
DOWNLOAD          RESUMEJOB    WARN
HEADOFF            SPOOLER       WELCOME
```

Use

You may issue this command from a session, job, program, or in BREAK. Pressing **Break** will terminate subsystem mode and produce an error message but has no effect on commands already entered in subsystem mode. This command is executable only from the console unless distributed to users with the `ALLOW` command.

Examples

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

```
ALLOW USER.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 at the system console:

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

Related Information

Commands	`DISALLOW`, `SHOWALLOW`
Manuals	*Performing System Operation Tasks*

ALTACCT

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] ]
[ ;ONVS=volumesetname] [ ;USERPASS=[ {REQ | OPT } ] ] (1)
```

(1) The USERPASS parameter is only available if the HP Security Monitor has been installed.

Parameters

acctname The name of the account to be altered.

password The *password* to be assigned to the account. If you omit *password*, any existing *password* is removed. If you omit PASS=, any existing password is unchanged.

filespace Disk storage limit, in sectors, for the permanent files in the account. The *filespace* limit cannot be less than the number of sectors currently in use for the account.

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. You may set the counter to zero with the RESETACCT command.

connect The limit on total cumulative session connect-time, in minutes, allowed the account. This limit is checked at logon. Every time the process terminates the counter is updated. The maximum value allowed is 2,147,483,647 minutes. You may reset the counter to zero with the RESETACCT command.

capabilitylist Either 1) a list of capabilities, separated by commas, permitted the account, or 2) a list of additions and/or deletions to be applied to the account's existing set of capabilities. Additions and deletions are specified by a "+" or "-" immediately followed by the capability to add or delete, separated by commas.

If "+"/"-" is to be specified in the list, then the list must begin with "+" or "-". For example, CAP=+MR,-PH is legal, but CAP=MR,-PH is not. It is not necessary to prefix each capability to be added or deleted with "+" / "-", as the occurrence of "+" / "-" indicates an action that remains in effect until the indicator changes. For example, CAP=+MR,PH,-PM,DS is equivalent to CAP=+MR,+PH,-PM,-DS

If a capability is removed at the account level, users within the account are also denied that capability. No explicit change to the user's capabilities is necessary. Similarly, if a capability is returned to the account, any users with that capability regain it automatically.

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
Save Files       =  SF
Access to Nonshareable
I/O Devices      =  ND

Use Volumes       =  UV

Use Communication
Subsystem       =  CS
Programmatic Sessions  =  PS
User Logging      =  LG
Process Handling    =  PH
Extra Data Segments   =  DS
Multiple RINs      =  MR
Privileged Mode     =  PM
Interactive Access   =  IA
Batch Access      =  BA
```

Default is AM, AL, GL, SF, ND, IA, BA, except for the SYS account. The SYS 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.

If a capability is taken away from an account, it is unavailable to users in that account. However, users are not affected by this change until they log off and then log back on.

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

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

The 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
A = APPEND
W = WRITE
X = EXECUTE
```

LOCK allows exclusive access to the file. APPEND implicitly specifies LOCK. WRITE implicitly specifies APPEND and LOCK.

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. When you specify `;MAXPRI=` without a value, *subqueuename* defaults to CS.

CAUTION User processes executing in the AS or BS subqueues can deadlock the system. If you assign these subqueues to nonpriority 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/iX 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).

volume- setname The MPE/iX volume set in which the account is to be altered. This volume set must be already defined and recognized by the system. When `ONVS=`*volumesetname* is specified, the volume set directory is assumed. When `ONVS=` is specified without *volumesetname*, the system directory is assumed.

MPE/iX volume set names consist of from 1 to 32 characters, beginning with an alphabetic character. The remaining characters may be alphabetic, numeric, the underscore, or periods.

This parameter only works with the `FILES` parameter (all other parameters are ignored).

`REQ` USERPASS=REQ specifies that all users in the account must have a non-blank password. It is available only if the HP Security Monitor has been installed.

`OPT` USERPASS=OPT specifies that users in this account may or may not have passwords. If you do not use the USERPASS parameter, the old value remains. It is available only if the HP Security Monitor has been installed.

Operation Notes

The system manager uses the `ALTACCT` command to change the attributes of an existing account. You may enter multiple keywords on a single command line as shown in "Examples." When you change one capability in a *capabilitylist* that contains several nondefault values, you must specify the entire new *%capabilitylist*. When you omit an entire keyword parameter group from the `ALTACCT` command, that parameter remains unchanged for the account. When you include a keyword, but omit the corresponding parameter (for example, PASS= **Return**), the default value is assigned. Table 2-2 lists the default values for the `ALTACCT` command.

Table 2-2 shos the defalut parameters for the ALTACCT Command.

Table 2-2 Default Parameters for the ALTACCT Command

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 (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	0 (null)

Any value changed with the ALTACCT command takes effect the next time MPE/iX is requested to check the value. If an attribute is removed from an account while users are logged on, they are not affected until they log off their current job or session and log on again. MPE/iX 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 are denied the capability the next time that they log onto the account.

You cannot remove system manager (SM) capability from the SYS account or account manager (AM) capability from any account. From within any account, you can remove AM capability from all but one (the last) of the users assigned it. It is possible, however, to remove AM capability from all users in an account, but only if you do so from another account that has SM capability.

NOTE If you specify volume-related commands or parameters for a volume set that is not currently mounted, or for an account that does not exist, MPE/iX returns an error message.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. System manager (SM) capability is required to use this command.

Examples

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
```

To change the password and the file space of an account called MALCHIOR in the volume set TIME_LORD, you need to issue two commands:

```
ALTACCT MALCHIOR;PASS=OMSBOROS
ALTACCT MALCHIOR;ONVS=TIME_LORD;FILES=20000
```

You must specify the changes for the system volume set (the first command) and for the volume set itself (the second command). Specifying a *volumesetname* limits the user to changing only FILES in the second command.

Related Information

Commands	ALTGROUP, ALTUSER, LISTACCT, LISTGROUP, LISTUSER, NEWACCT, NEWGROUP, NEWUSER, RESETACCT
Manuals	*Performing System Management Tasks*

ALTFILE

Changes the attributes of an existing file or directory. (Native Mode)

Syntax

```
ALTFILE[ FILE=] filename [ ] ;OWNER=ownername] ] [ [ ;GROUPID=] POSIXgroupname]
```

Parameters

filename The filename of the object to be altered, specified in either MPE or HFS syntax. The *filename* may name a file, hierarchical directory, root, MPE group or account. Note that MPE groups or accounts can ONLY be named via HFS (Hierarchical File System) syntax. Temporary files are not recognized.

This is a required parameter. You may not use wildcards, back-reference a file equation, or name a system-defined file such as $NULL.

If the filename is in MPE syntax and it has a lockword, do NOT include the lockword on the command line, or you will get an error.

ownername The name of the user who will become the owner (UID) of *filename*. This *ownername* must already exist on the system. Default is for the UID of the file to remain unchanged. Note that no qualification is done on this name; it must be fully specified. To have the *ownername* upshifted, enclose it in quotes.

POSIXgroupname The name of the POSIX group (GID) that this file will belong to. This *POSIXgroupname* must already exist on the system. You cannot use this parameter to change the GID of an MPE group or account. Default is for the file to retain its previous GID. To have *POSIXgroupname* upshifted, enclose it in quotes.

Operation Notes

You use the `ALTFILE` command to alter a file's characteristics. Currently the attributes that you may modify are the owner (UID) and POSIX group (GID) for a file, hierarchical directory, MPE group or account, with the restriction that you may not alter the GID for MPE groups or accounts.

You must have the appropriate privilege to change the requested attribute(s). In order to change the UID of a file, you must be one of the following:

- The file's account manager (your logon account matches the GID of the file and you have MPE/iX account manager (AM) capability). In this case, *ownername* must specify a user belonging to the account manager's logon account.
- A system manager (a user who has the MPE/iX system manager (SM) user capability). In this case, *ownername* may specify any user existing in the user database.

In order to change the GID of a file, you must be one of the following:

- The file owner (your logon name matches the UID of the file). In this case, *POSIXgroupname* must specify your logon account.
- The file's account manager (your logon account matches the GID of the file and you have the MPE/iX account manager (AM) capability). In this case, *POSIXgroupname* must specify the account manager's logon account.
- A system manager (you have MPE/iX system manager (SM) capability). In this case, *POSIXgroupname* may specify any GID existing in the group database.

You may issue the command once to modify multiple attributes. If you specify multiple attributes, all modifications must succeed for any to take effect. If you enter no attributes, the command has no effect on the specified file.

Related Information

Commands	`ALTSEC`, `LISTFILE`, `RELEASE`, `SECURE`
Manuals	*Performing System Management Tasks*

ALTGROUP

Changes one or more attributes of a group.

Syntax

```
ALTGROUP groupname [ .acctname]
[ ;PASS=[ password]] [ ;CAP=[ capabilitylist] ]
[ ;FILES=[ filespace] ] [ ;CPU=[ cpu] ]
[ ;CONNECT=[ connect] ] [ ;ACCESS=[ (fileaccess)] ]
[ ;ONVS=volumesetname] [ ;HOMEVS=volumesetname]
```

Parameters

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

acctname The name of the account in which the group is to reside. System manager (SM) capability is required to use this parameter.

password The *password* to be assigned to the group, which is used to verify logon and access only. If the `PASS` parameter is omitted, no change is made. If `PASS` is used and *password* is omitted, the existing password is removed. If `PASS` is used and *password* is specified the existing password is changed; if there is no existing password for the group a password is created.

capabilitylist Either 1) a list of capabilities, separated by commas, permitted this group, or 2) a list of additions and/or deletions to be applied to the group's existing set of capabilities. Additions and deletions are specified by a "+" or "-" immediately followed by the capability to add or delete, separated by commas.

If "+"/"-" is to be specified in the list, then the list must begin with "+" or "-". For example, `CAP=+MR,-PH` is legal, but `CAP=MR,-PH` is not.

It is not necessary to prefix each capability to be added / deleted with "+" / "-", as the occurrence of "+" / "-" indicates an action that remains in effect until the indicator changes. For example, `CAP=+MR,PH,-PM,DS` is equivalent to `CAP=+MR,+PH,-PM,-DS`.

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
Batch Access     =    BA
```

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

filespace

Disk 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. Nor can a group's *filespace* be set to a value less than the actual number of sectors in use in that group. 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 allowed is 2,147,483,647 seconds. If the limit is exceeded, users with account manager capability are warned when logging on; other users are denied access.

The CPU limit for a group cannot be set to a value greater than the corresponding limit currently defined for the group's account. Default is unlimited CPU-time. The counter may be set to zero with the RESETACCT command.

connect

The limit on the total cumulative session connect-time, in minutes, that the group is allowed. This limit is checked at logon, and whenever the session initiates a new process. The maximum value allowed is 2,147,483,647 minutes. If the limit is exceeded, users with account manager capability are warned when logging on; other users are denied access.

A group's connect limit cannot be set to a value greater than the corresponding limit currently defined for the group's account. 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, where R, L, A, W, and X specify modes of access by types of users (ANY, AC, GU, AL, GL) as follows:

```
R =  READ
L =  LOCK
A =  APPEND
W =  WRITE
X =  EXECUTE
S =  SAVE
```

LOCK allows exclusive access. APPEND implicitly specifies LOCK. WRITE implicitly specifies APPEND and LOCK.

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
```

To specify two or more user or access types, separate them by commas.

`ONVS` A particular volume set for which the group attributes are to be changed. The volume set must be already defined and recognized by the system. If you specify `ONVS`, the only other parameter that works with it is the `FILES` parameter. If *volumesetname* is omitted from the `ONVS=` parameter, or you omit `ONVS`, the operation is performed on the system volume set.

`HOMEVS` Changes the home volume set from the current set to the set specified by *volumesetname*. You may do this only if the group on the current home system volume set is empty and not in use; no one is logged onto that group.

volume setname The full name of the MPE/iX volume set, consisting of from 1 to 32 characters, beginning with an alphabetic character. The remaining characters may be alphabetic, numeric, the underscore, or periods.

You cannot change the home volume set if the home volume set is the system volume set, and it contains files. If it contains no file, you can change the home volume set.

Consider the following when changing the home volume set:

- If the home volume set is the system volume set, no files may exist in the group and the group may not be in use (no users may be logged onto the group). Otherwise, the command fails.
- If the current home volume set is not the system volume set but the volume set is mounted, no files may exist in the group on that volume set, and the group may not be in use. Otherwise, the command fails.
- If the current home volume set is not the system volume set and it is not mounted, it may be changed.

It is permissible to reassign a group to a different volume set despite the presence of files belonging to *groupname*. This is possible provided that the old volume set is not the system volume set and the *groupname* is not currently bound to its home volume set. This binding occurs automatically when the volume set is mounted; it occurs explicitly when the `MOUNT` or `VSOPEN` commands are invoked; it occurs implicitly when the `FOPEN` intrinsic is invoked.

Operation Notes

This command changes one or more attributes of a group. Multiple parameters may be specified on a single command line as shown in "Examples." 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 = **Return**), the default value is assigned. Table 2-3 lists the default values for the ALTGROUP command. Table 2-3 shows the default values for the ALTGROUP Command.

Table 2-3 Default Values for the ALTGROUP Command

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

When a parameter is modified with 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 log off their current session and log on to that *username* and group again. For this reason, notify all group users of any planned changes in advance.

NOTE If you specify volume created commands or parameters for a volume set that is not currently mounted, or for an account that does not exist, MPE/iX returns an error message.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. Account manager (AM) or system manager (SM) capability is required to use this command.

Examples

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

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

To alter the group LEILA that resides on the volume set TIME_LORD:

```
ALTGROUP LEILA;ONVS=TIME_LORD;FILES=10000
```

If the group LEILA contains no files, and no one is logged onto the group, you may also alter the home volume set to DICHONDRITE, provided DICHONDRITE exists and is recognized by the system:

```
ALTGROUP LEILA;HOMEVS=DICHONDRITE
```

However, if `LEILA` does contain files, you cannot change the home volume set for this group without creating a new group and transferring those files to it.

Related Information

Commands	ALTACCT, ALTUSER
Manuals	*Volume Management Reference Manual*

ALTJOB

Alters the attributes of waiting or scheduled jobs. (Native Mode)

Syntax

```
ALTJOB[ JOB=] { #Jnnn  #Snnn }
[ ;INPRI=inputpriority] [ ;OUTDEV={ ldev devclass  } ]
[.HIPRI][;JOBQ=queuename]
```

Parameters

#J*nnn* A job number.

#S*nnn* A session number. (Although syntactically correct, this parameter is rarely used: sessions do not wait.)

inputpriority The new input priority (0 = lowest; 14 = highest).

ldev or *devclass* The logical device number or device class name of the destination device job's $STDLIST.

HIPRI Allows the OP or SM to bypass the joblimit, see the JOB command for more detail.

queuename The name of the job queue whose limit is being changed.

Operation Notes

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, WAIT, or SCHED state. Jobs with an input priority less than or equal to the current JOBFENCE, a numerical value from 0 to 14, are deferred.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **BREAK** has no effect on this command. If issued from the console or by a user with SM capability, or *allowed* via the ALLOW command, the ALTJOB command can be used to alter *any* job. A user who is *not* at the console, does not have SM or has *not* been allowed the command may issue ALTJOB only if *jobsecurity* is set to *low*. If *jobsecurity* is set to *low* then all users can issue ALTJOB against their own jobs and account managers (AM capability) can issue it against any job in that account.

Example

In the following example, three jobs are submitted by users, each with an INPRI value of 8. To change the 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
15:11/#J2/25/DEFERRED JOB INTRODUCED ON LDEV #53
15:13/#J3/26/DEFERRED JOB INTRODUCED ON LDEV #53


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,OP.SYS
#J2   WAIT  D 8  10S 12   THU 3:11P JOB3,SUE.PAYROLL
#J3   WAIT  D 8  10S 12   THU 3:13P JOB1,JIM.ACCTG

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
JOBFENCE 6
SHOWJOB

JOBNUM  STATE IPRI JIN  JLIST INTRODUCED JOB NAME

#S23   EXEC     20  20   THU 2:15P OPERATOR.SYS
#J1   EXEC   10  10S 12   THU 3:13P JOB2, OP.SYS
#J3   EXEC   9  10S LP   THU 3:11P JOB1,JIM.ACCTG
#J2   EXEC   8  10S 12   THU 3:11P JOB3,SUE.PAYROLL

4 JOBS:
 0 INTRO
 0 WAIT; INCL 0 DEFERRED
 4 EXEC; INCL 1 SESSIONS
 0 SUSP
JOBFENCE= 6; JLIMIT= 5; SLIMIT= 16
```

Related Information

Commands	JOBFENCE, JOBSECURITY, LISTJOBQ
Manuals	*Performing System Operation Tasks*

ALTLOG

Alters the attributes of an existing user logging identifier.

Syntax

```
ALTLOG logid [ ;LOG=logfile { ,DISC ,TAPE } ] [ ;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 log file resides, either `DISC` or `TAPE`.

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 log file becomes full. This option is ignored is `TAPE` is specified. Refer to the `CHANGELOG` command.

`NOAUTO` Prevents the initiation of an automatic `CHANGELOG`. A `CHANGELOG` is not performed if the current log file becomes full. Default.

Operation Notes

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. System supervisor (OP) or user logging (LG) capability is required to use this command. 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 file name with the numeric characters 001 (for example, *fname001*). This naming convention works in conjunction with the file set number to generate sequential file names automatically.

If a log file is restricted to a single volume or volume class when it is created with the `BUILD` command, then successive log files created by User Logging will have the same restriction.

If a new log file name is specified with the `ALTLOG` command, the links with any previous log file are broken.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. User logging (LG) capability is required to use this command.

Example

To change the destination log file of the logging identifier KIM to log file C and specify that C resides on disk, enter:

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

Since the keyword parameter, PASS=, was omitted, KIM retains any password previously specified.

Related Information

Commands	CHANGELOG, GETLOG, LISTLOG, SHOWLOGSTATUS
Manuals	*User Logging Programmer's Guide System Startup, Configuration, and Shutdown Reference Manual*

ALTPROC

Changes characteristics of the specified processes. Currently, a process' priority, queue attribute, and workgroup may be changed. (Native Mode)

Syntax

```
ALTPROC[ [ PIN=] { pinspec (pinspec [ ,pinspec ] ...) } [ ;JOB=] { jobspec (jobspec [ ,jobspec ]
...) } ]
[ [ ;PRI=] pri [ ;WG= ] { workgrp NATURAL_WG } ]
[ { ;TREE ;NOTREE }]
[{ ;USER ;ANYUSER } ]
[ ;SYSTEM]
```

Parameters

pinspec The process(es) you want to alter. This is a required parameter, unless you specify *jobspec*. If you omit both, you will get an error.

The *pinspec*, expressed [#p]pin, is a Process Identification Number (PIN). If *pinspec* is 0, then the caller's pin is used. To alter system processes, you must have SM capability and specify the SYSTEM option.

NOTREE is the default for all *pinspec* target processes, and can be overridden with the TREE option.

The USER and ANYUSER options do not apply to *pinspec*.

jobspec The name of the job or session whose processes are to be altered. A *jobspec* can be any of the following, jobnumber, username, @S, @J, or @.

- The jobnumber must be in the form of either *#Jnnn* or *#Snnn*.
- The username must be in the form *user*[.*account*]. If there is more than one job/session matching the same username, they will all be altered.
- Wildcards have the following meanings:

  @S - all sessions,

  @J - all jobs,

  @ - all sessions and jobs

The USER and ANYUSER options apply only to *jobspec* and only if *jobspec* is wildcarded. The USER option, which is the default, alters only processes matching the user's name, while the ANYUSER option alters all processes matching the wildcarded *jobspec*. For example, if the user's name is STEVE.UI and you enter the command shown below, then only job processes logged on as STEVE.UI are altered.

```
:ALTPROC job=@j;pri=cs
```

However, if you add anyuser to the same command as shown below, then all job processes are altered.

```
:ALTPROC job=@j;pri=cs;anyuser
```

TREE is the default for all *jobspec* target processes, and can be overridden with the NOTREE option.

The SYSTEM option is ignored for all *jobspec* target processes.

The *jobspec* is optional as long as a *pinspec* is supplied. If both are omitted, an error is reported.

pri The queue or absolute priority for the process. If omitted, the priority is unchanged.

CAUTION Exercise extreme caution when altering a process's priority, scheduling queue attribute, or workgroup membership. Such a change can significantly impact system performance.

NOTE **For Workload Manager Users**

Avoid using the ;PRI= option to explicitly change a process. If you have created user-defined workgroups that have ;MEMB_QUEUE as membership criteria, use of the ;PRI= option may change the workgroup. Instead, use either the ;WG=*workgrp* or ;WG=NATURAL_WG option, explained below, to move target processes into existing workgroups.

Using ;WG= to explicitly change a workgroup should be a temporary measure, and used rarely. Instead, adjust workgroup membership criteria to ensure that desired processes become natural members of the workgroup.

If you specify both the ;WG= and ;PRI= in the ALTPROC command line, you will get an error.

The *pri* value may be one of the following:

- A scheduling queue value {BS,CS,DS,ES} sets the queue attribute of the target process(es). If a user-defined workgroup does not capture the process, then the process will fall into to the corresponding system-defined default workgroup at the base priority (subject to decay as it consumes CPU). To assign a scheduling queue value, you must have OP capability.
- A queue manager value {BM,CM,DM,EM} sets the queue attribute of the target process(es). If a user-defined workgroup does not capture the process, then the process will fall into the corresponding system-defined default workgroup at the base priority (non-decayable). To assign a queue manager value, you must have SM capability.
- An absolute priority {*nnn*} sets the priority of the process to the specified value that will not decay. The workgroup of the process will not be changed (the process will have the same timeslice value). Note that the

priority specified need not fall between the base and limit priorities of the workgroup. To assign an absolute priority value, you must have SM capability.

If you do not have SM capability, then your `MAXPRI` value represents the highest priority that you can assign a process. A warning appears when the specified priority exceeds `MAXPRI`. `MAXPRI` is ignored for System Manager (SM) capability.

workgrp A workgroup value {*workgrp*} moves the target process(es) to the specified **workgroup**. A process moved in this manner is considered an **artificial member** of the workgroup (the process was placed in workgroup explicitly, rather than naturally by meeting the membership criteria specified for the workgroup).

A process remains an artificial member of its assigned workgroup until either the workgroup is purged or the process' explicit assignment is changed (via `ALTPROC` or an AIF call). An artificial member is not affected by a **system-wide scan** or by the changing of its process attributes used to determine workgroup membership. A workgroup specification requires SM capability and can only be used to modify the workgroup assignment of user processes.

You cannot specify both the `;WG=` and `;PRI=` in the `ALTPROC` command line. Workload Manager users should use `;WG=` instead of `;PRI=`.

`NATURAL_WG` The natural workgroup specification {`NATURAL_WG`} releases one or more process(es) from their explicit workgroup assignment, allowing them to migrate to their **natural workgroup**. A natural workgroup specification requires SM capability.

`TREE` This option alters each process specified as well as all of its descendants. `TREE` is the default for all *jobspec* target processes. If you specify both `TREE` and `SYSTEM,` you will see a warning that `TREE` will be ignored.

`NOTREE` This option alters only the processes specified. Descendant processes will not be altered. `NOTREE` is the default for all *pinspec* target processes.

`USER` The `USER` option applies only when *jobspec* is wildcarded. It alters only processes matching the user's name. `USER` is the default.

`ANYUSER` The `ANYUSER` option applies only when *jobspec* is wildcarded. It alters all *jobspec* target processes, regardless of their owners.

`SYSTEM` Use the `SYSTEM` option if the target process specified in *pinspec* is a system process. SM capability is required for the `SYSTEM` option. `SYSTEM` is ignored for all *jobspec* processes and when you specify a workgroup or natural workgroup. If you specify both `SYSTEM` and `TREE`, you see a warning that `TREE` will be ignored.

CAUTION Exercise extreme care when altering system processes since doing so can significantly degrade system efficiency.

Operation Notes

To execute the ALTPROC command, you must have System Supervisor (OP) or System Manager (SM) capability. SM capability is necessary to alter system processes, for the WG= option, for certain specifications to the PRI option, and to increase a process' priority above MAXPRI. You may issue the ALTPROC command from a session, job, program, or while in BREAK. Pressing **Break** aborts the execution of this command.

Example

To alter process 605, and its current descendants, so that their priorities execute within the DS_Default workgroup, enter:

```
:ALTPROC #p605; tree; wg=DS_Default
```

To alter process 605, and its current descendants, so that their scheduling queue attribute is DS, enter:

```
:ALTPROC #p605; tree; pri=DS
```

The outcome of this command is not necessarily identical to the outcome achieved with the previous command. If the system was configured with a user-defined workgroup that captured the processes (MEMB_QUEUE=DS and a match on other membership attributes, if specified), then the processes would be a member of the user-defined workgroup rather than the DS_Default workgroup.

To alter all job processes to the CS_Default workgroup, enter:

```
:ALTPROC job=@j; wg=CS_Default; anyuser
```

To return the processes modified by the previous example to their natural workgroup(s), enter:

```
:ALTPROC job=@j; wg=NATURAL_WG; anyuser
```

To alter all job processes matching the user's name to the CS_Default workgroup, enter:

```
:ALTPROC job=@j; wg=CS_Default; user
```

To alter the current process' priority so that it behaves like a CS queue manager (SM capability required), enter:

```
:ALTPROC 0;pri=CM
```

To alter all processes logged on as mgr.payroll to linear 155 (SM capability required), enter:

```
:ALTPROC job=mgr.payroll; pri=155
```

To alter the queue attribute of pins 150, 247, 211 to be ES, enter:

```
:ALTPROC (150,#p247,211); pri=ES
```

Related Information

Commands	SHOWPROC, TUNE, SHOWQ, NEWWG, ALTWG, PURGEWG, SHOWWG
Manuals	*MPE/iX Intrinsics Reference Manual*
	Using the HP 3000 Workload Manager

ALTSEC

Changes the access permissions of an object by altering the access control definition (ACD).

ACDs are the main method of controlling access to files, hierarchical directories, and devices. ACDs are automatically assigned to hierarchical directories and to files existing in hierarchical directories.

You can change access permissions for any of the following:

- files
- hierarchical directories
- devices
- device classes

You can also use ALTSEC to change the access masks of files. The file status change time stamp is updated by ALTSEC. You cannot use the ALTSEC command to change access permissions for MPE groups, accounts, or the root directory.

Syntax

```
ALTSEC objectname [ ,{FILENAME LDEV DEVCLASS } ]
[ ;[ ACCESS=] (fileaccess[ ;[ fileaccess] [ ;...] ] )]
[{ ;NEWACD= ;ADDPAIR= ;REPPAIR= } { (acdpair [ ;acdpair] [ ;...] ) ^filereference } ]
[ ;DELPAIR= { (userspec [ ;userspec] [ ;...] ) ^filereference } ]
[ { ;REPACD=} { (acdpair ;acdpair [ ;...] ) ^filereference objectname } ]
[ ;COPYACD= objectname { ,FILENAME ,LDEV } ] [ ;DELACD] [ ;MASK]
```

Parameters

objectname Specifies the actual file designator, directory name, logical device number, or device class whose security provisions you want to alter.

Either MPE or hierarchical file system (HFS) file name Syntax may be used for the actual file designator of the file or directory whose access permissions are to be altered.

You can only use wildcard characters with MPE Syntax files that reside in a group.

A logical device number must be a numeric value configured on the system, or an @ sign, that indicates all devices on the system. A device class name must be configured on the system.

File equations are ignored during resolution of the object name to avoid having accidental file equation references cause unintentional changes to an object's access permissions.

MPE Syntax

You can include MPE file name Syntax but not RFA information. If the object is an MPE Syntax file, its format is:

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

You may specify file lockwords for files protected by active lockwords unless the objects are also protected by a current ACD. In a batch job, if a lockword exists on a file, you must specify it. In a session, if a lockword exists and is omitted, MPE/iX will prompt you for it.

HFS Syntax

You must begin file designators using HFS file name Syntax with either a dot (.) or a slash (/). The maximum length is 255 characters (including the "./" or "/").

The *objectname* parameter is followed by one of the three *type identifiers* listed below.

`FILENAME` Indicates that *objectname* refers to either a file or directory. This is the default if a type identifier is not specified.

`LDEV` Indicates that *objectname* refers to a logical device number.

`DEVCLASS` Indicates that *objectname* refers to a device class.

`ACCESS` Optional keyword that indicates a *fileaccess* specification follows. This option affects security at the *file* level only. If the file is protected by an ACD, the ACD overrides the file access mask.

fileaccess File access mask specifications, entered as follows:

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

The 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
A  =  APPEND
W  =  WRITE
X  =  EXECUTE
```

`LOCK` allows opening the file with dynamic locking option. `APPEND` implicitly specifies `LOCK`. `WRITE` implicitly specifies `APPEND` and `LOCK`. You may specify two or more *modes* if you separate them 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 = Creator
```

You may specify two or more user types if you separate them by commas. The 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*.

`NEWACD` Creates a new ACD for the specified object. `NEWACD` is used when an ACD does not currently exist. It must be followed by valid ACD pair(s) as described below.

`REPACD` Indicates "replace ACD". Use `REPACD` to replace an entire existing ACD for the specified object, or to copy an ACD from an existing *objectname* to the specified *objectname* where *objectname* refers to a file. (You cannot use `REPACD` to copy ACDs between devices.) The `REPACD` parameter must be followed by valid ACD pair(s) as described below.

`ADDPAIR` Adds a new ACD pair to an existing ACD. It must be followed by valid ACD pair(s) as described below.

`REPPAIR` Replaces an existing ACD pair in an existing ACD. You must follow this with a valid ACD pair(s) as described below. A new ACD pair will replace an existing ACD pair if it has the same user and account name.

acdpair An access control definition pair. Like the *fileaccess* parameter this consists of a *modes* part and a *userspec* part. The *modes* part is separated from the *userspec* part by a colon (:). Acceptable *modes* for files are:

```
   R : read file access
   W : write file access
   L : lock file access
   A : append file access
   X : execute file access
NONE : no access
RACD : copy or read the ACD permission
```

Acceptable *modes* for directories are:

```
  CD : create directory entries access
  DD : delete directory entries access
  RD : read directory entries access
  TD : traverse directory entries access
NONE : no access
RACD : copy or read the ACD permission
```

File ACD pairs may contain R, W, L, A, X, NONE, and RACD. Directory ACD pairs may contain CD, DD, RD, TD, NONE, and RACD.

The *userspec* part consists of

- a fully qualified user name (*username.accountname*)
- the file owner represented as `$OWNER`
- the file group represented as `$GROUP`
- the file group mask represented as `$GROUP_MASK`
- *@.accountname*, which represents all users in the account `accountname`
- *@.@*, which represents all users in the system

You cannot use wildcards in any other manner within a user specification.

A typical ACD consisting of three ACD pairs might look like this:

```
(R,W:ENGR.MFG;R,W,RACD:@.MRKT;R:@.@)
```

This ACD would allow Read and Write access to the ENGR user of the MFG account; Read and Write access to any user of the MRKT account along with the ability to read or copy the ACD; and Read access to any user in any account.

^ *filereference*

A file containing one or more ACD pairs. ACD pairs must be separated by semi-colons and may be placed on separate lines. A single ACD pair may *not* span more than one line. The file name must be preceded by the ^ sign (caret symbol) to indicate that the designated file contains the ACD definition. This is known as an indirect file.

The ALTSEC command fails if the indirect file does not contain a syntactically correct ACD. ACD pairs may be on separate lines, but a pair may not span lines. Parentheses are optional when defining an *acdpair* within an indirect file.

The file reference may be specified using MPE or HFS file name Syntax. For example:

filename[/*lockword*] [.*group*[.*account*]]

If the file has an active lockword, you must be specify it. ACDs override lockwords. Lockwords can only be specified in file references using MPE name Syntax. Unqualified file names are relative to the current working directory.

DELPAIR

(Indicates "delete pair"). Use to delete one or more ACD pairs in an existing ACD). DELPAIR must be followed by a valid *userspec*.

userspec

Username and accountname, the same as the *userspec* described above in *acdpair*. A wildcard (@) may be used for the username or both the username and accountname together. A wildcard may *not* be specified for the accountname unless it is also specified for the username.

COPYACD

(Indicates "copy ACD"). Use COPYACD to copy an ACD from an existing *objectname* to the specified *objectname*. ACDs can be copied only between like objects. You must specify FILENAME or LDEV. FILENAME is the default. You cannot copy an ACD *from* a device class (DEVCLASS), although you may copy *to* all devices on the system by specifying the @ sign as the target device.

DELACD

(Indicates "delete ACD"). Use DELACD to delete all ACD pairs from the specified *objectname*. ACDs may be removed only from devices and files in MPE groups. The file access matrix controls access to a file when an ACD is deleted.

MASK

(Indicates "recalculate MASK"). Use MASK to recalculate the ACD file group class mask ($GROUP_MASK) access permissions.

Operation Notes

You use the ALTSEC command to alter security provisions for files, hierarchical directories, devices, and device classes by manipulating an object's access control definition (ACD) or its access mask. All of these objects may have ACDs, but only files have access masks which can be changed using this command. An object's ACD may be altered using this command with the ACD keywords NEWACD, REPACD, COPYACD, ADDPAIR, REPPAIR, DELPAIR, DELACD, and MASK.

A file's access mask may be altered using either the ACCESS keyword or an access specification without a keyword. Using the ACCESS keyword is a recommended practice to help distinguish between file access mask and ACD operations. Only the owner of a file can use the ALTSEC command to change a file's access mask. Object owners and users with appropriate privilege can use this command to manipulate an object's ACD. Files and hierarchical directories have their owner's identity and a file group ID (GID) stored in their file labels. System managers have the appropriate privilege to manipulate the ACDs for all objects. Account managers for the account matching an object's GID have appropriate privilege. Devices are owned by system managers. The ability to manipulate an ACD or file mask is not affected by the object access currently granted to a user.

File ACDs override file lockwords and the file access matrix. ACDs permit more precise access control than the file access matrix by allowing access permissions to specific users. MPE/iX allows you to specify a maximum of 40 ACD pairs for a particular object. Since a large number of ACD pair specifications overflows the command line buffer, you must enter large numbers of ACD specifications may be entered through an indirect file.

The ALTSEC command fails if you attempt to alter the access permissions for a permanent disk file whose group's home volume set is not mounted.

Release 5.0 requires ACDs on the following files:

- All hierarchical directories
- All files under hierarchical directories
- All files directly under MPE/iX groups where the file GID does not match the GID of the accound and group in which the file is located. One way this occurs would be if you rename a file from an MPE group outside the account to another MPE group.

Required ACDs cannot be removed with the ALTSEC command even by users with SM or AM capability.

File Access Matrix Examples

To view the file access matrix, use LISTFILE,4.

You have created a file named FDATA, and want to change its file access matrix access permissions to grant write access to only yourself. Enter:

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

To change file access permissions for the FPROG program file to allow all group users to execute programs, but only account and group librarian users to read or write to the file, enter:

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

ACD Examples

To view ACD information, use the LISTFILE,-2 command. This form of the LISTFILE command displays *only* ACD information.

You have created a file named FDATA, and want to assign a new ACD to FDATA, granting write access to a user named FRIEND.ACCT. Enter:

```
ALTSEC FDATA;NEWACD=(W:FRIEND.ACCT)
```

As the creator of a file, you can access the file by default, so you don't need to grant yourself access through an ACD. Users with appropriate privileges are always permitted to access files protected by ACDs.

To extend the ACD for the FDATA file so that all users on the system can read it, and all users within your account ACCT can also write to it, enter:

```
ALTSEC FDATA;ADDPAIR=(R:@.@;W,R:@.ACCT)
```

If you decide that users outside your account ACCT should not have read access to the file FDATA any longer, enter:

```
ALTSEC FDATA;DELPAIR=(@.@)
```

This does not delete all ACD pairs, only the ACD pair matching @.@. To delete the entire ACD, enter:

```
ALTSEC FDATA;DELACD
```

To replace the entire ACD, enter:

```
ALTSEC FDATA;REPACD=(W:FRIEND.ACCT)
```

You want to copy the ACD associated with LDEV 5 to all devices in device class TERM:

```
ALTSEC TERM,DEVCLASS;COPYACD=5,LDEV
```

ACDs may be copied only between objects of the same type.

You want to grant users in account ACCT all access to directory Mydir1:

```
ALTSEC ./Mydir1;ADDPAIR=(CD,DD,RD,TD,RACD:@.ACCT)
```

You want to grant read and write access to yourself and read access for other members of your group to an HFS Syntax file named a_file_of_Mine:

```
ALTSEC ./a_file_of_Mine;REPPAIR=(RACD,R,W:$OWNER;
RACD,R:$GROUP,$GROUP_MASK;NONE:@.@)
```

To add a new ACD to file PROGNAME allowing all users on the system to execute it, but only users in account ACCT to write to it enter:

```
ALTSEC PROGNAME;NEWACD=(X:@.@;W,X:@.ACCT)
```

To add a new ACD pair to an ACD which already exists for file PROGNAME which will allow the user ENGR of the LAB account to read, write, lock, append, execute and read the ACD information enter:

```
ALTSEC PROGNAME;ADDPAIR=(R,W,X,RACD:ENGR.LAB)
```

Note that L and A (lock and append) need not be specified because they are implied with W (write).

To add an ACD that prevents any user except OPERATOR.SYS (and any user with SM capability) from accessing LDEV 7 (a tape drive), enter:

```
ALTSEC 7,LDEV;NEWACD=(R,W:OPERATOR.SYS)
```

Note in the last example that X is not used because it makes no sense to *execute* a tape drive. It also makes no sense to *lock* or *append* a tape drive but W tacitly provides L and A anyway.

To eliminate any ACD that may be in effect for device class LP, and to prevent any user except *MGR.FINANCE* from writing to a printer in device class LP, enter:

```
ALTSEC LP,DEVCLASS;DELACD
ALTSEC LP,DEVCLASS;NEWACD=(W:MGR.FINANCE)
```

Related Information

Commands	LISTF, LISTFILE, RELEASE, SECURE, SHOWDEV, and the *fileaccess* parameter for the ALTACCT, ALTGROUP, NEWACCT and NEWGROUP commands.
Manuals	None

ALTSPOOLFILE

Alters the characteristics of an output spoolfile.

Syntax

ALTSPOOLFILE{ *#Onnn ldev1* } { ;PRI=*outputpriority* ;COPIES=*numcopies* ;DEV={ *ldev2 devclass* }
;DEFER } [;...]

Parameters

#O*nnn* The output device file identification of a spoolfile.

ldev1 The logical device number of the device where an ACTIVE spoolfile currently resides.

outputpriority The output priority of the designated device file (0 = lowest; 14 = highest).

numcopies The number of copies to be produced from the designated device file. Range is 1 through 127; default is 1.

ldev2 or *devclass* 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.

Operation Notes

The operator uses the ALTSPOOLFILE command 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/iX returns the colon prompt (:). No change to the spoolfile is made, however, until the output device is returned online.

NOTE If you are altering the PRI or COPIES parameter for an ACTIVE spoolfile there is no need to take the output device offline. These two parameters can be altered while the device is 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 stops immediately. In both cases, the entire file is 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.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It is executable from the console by users with system supervisor (OP) or system manager (SM) capability. It may be distributed to other users with the ALLOW or ASSOCIATE command.

Examples

To defer the ACTIVE spoolfile (#O86) on LDEV 6 take device 6 offline, then enter:

```
ALTSPOOLFILE #O86;DEFER
```

or

```
ALTSPOOLFILE 6;DEFER
```

To change the priority of deferred spoolfile #O123 from 0 to 3 enter:

```
ALTSPOOLFILE #O123;PRI=3
```

Related Information

Commands	OUTFENCE
Manuals	*Native Mode Spooler Reference Manual*

ALTUSER

Changes the attributes currently defined for a user.

Syntax

ALTUSER *username*[*.acctname*]

[;PASS=[*password*]] [;CAP=[capabilitylist]]

[; MAXPRI=[*subsueuename*]] [;LOCATTR=[*localattribut*]]

[;HOME=[*homegroupname*]] [;UID=[*uid*]

[;USERPASS=[*req opt*] [*Expired*]]

The USERPASS parameter is only available if the HP Security Monitor has been installed.

Parameters

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

acctname The account in which the user is to reside. System manager (SM) capability is required to use this parameter.

password The *password* to be assigned to the user. If *password* is omitted, any existing *password* is removed. If PASS= is omitted, any existing password is unchanged.

capabilitylist Either 1) a list of capabilities, separated by commas, permitted to this user, or 2) a list of additions and/or deletions to be applied to the user's existing set of capabilities. Additions and deletions are specified by a "+" or "-" immediately followed by the capability to add or delete, separated by commas.

If "+"/"-" is to be specified in the list, then the list must begin with "+" or "-". For example, CAP=+MR,-PH is legal, but CAP=MR,-PH is not. It is not necessary to prefix each capability to be added or deleted with "+" / "-", as the occurrence of "+" / "-" indicates an action that remains in effect until the indicator changes. For example, CAP=+MR,PH,-PM,DS is equivalent to CAP=+MR,+PH,-PM,-DS.

The capabilities allowed to users are restricted by the capabilities assigned to the user's account. If a capability is absent at the account level, users within the account are also denied that capability, whether or not it is explicitly assigned to them.

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
```

```
Save Files      =  SF
Access to Nonshareable
 I/O Devices     =  ND
Use Volumes     =  UV
Create Volumes    =  CV
Use Communication
 Subsystem        CS
Programmatic Sessions =  PS
User Logging     =  LG
Process Handling   =  PH
Extra Data Segments  =  DS
Multiple RINs     =  MR
Privileged Mode    =  PM
Interactive Access  =  IA
Batch Access     =  BA
Programmatic Sessions =  PS
```

Default is SF, ND, IA, and BA. Note that CV automatically gives the user UV capability, and removal of UV results in automatic removal of CV.

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 at logon is checked against the *subqueuename* defined for the user, and the user 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 nonpriority 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 arbitrary double word bit map is used to further classify users. While it is not part of standard MPE/iX 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 assigned as the home group for this user. The first user established when an account is created, by default, has PUB assigned as the home group. Subsequent new users, by default, have no home group assigned. If no home group is assigned, the user must always specify an existing group when logging on.

uid User ID to be altered for the account manager in the user database. The *uid* parameter must be a unique positive (non-zero) 32-bit integer.

Req USERPASS=REQ specifies that all users in the account must have a non-blank password. It is available only if the HP Security Monitor has been installed.

Opt USERPASS=OPT specifies that users in this account may or may not have passwords. If you do not use the USERPASS parameter, the old value remains. It is available only if the HP Security Monitor has been installed.

Expired The password expires immediately. The user cannot logon without selecting a new password. It is only available if the HP Security Monitor has been installed.

Operation Notes

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 semicolon (;) delimiter.

To change an attribute, enter the keyword and its new value. 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=**Return**), a default value is assigned as shown in

Table 2-4 'Default Values for the ALTUSER Command' on page 81 .

Table 2-4 Default Values for the ALTUSER Command

Parameter	Default Values
password	NULL password
capabilitylist	SF, ND, IA, and BA (provided these capabilities have been specified for the account)
subqueuename	CS
localattribute	0 (null)
homegroupname	The first user established when the account is created has PUB assigned as home group. 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 does not affect users who are currently logged on to the system. They are affected the next time they log on to the same user name and account. For this reason, warn users in advance of any intended changes.

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

ALTUSER will not allow a user with AM capability to remove AM from their own capability list. However, a user with AM can remove AM from the capability list of another AM user inside the same account.

Use

This command may be issued from a session, a job, a program, or in break mode. Pressing **Break** has no effect on this command. Account manager (AM) capability is required to use this command. System manager (SM) capability is required to specify a user in an account other than your own.

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 (MR) capability, 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
```

Related Information

Commands	ALTACCT, ALTGROUP, LISTUSER, NEWACCT, NEWUSER
Manuals	*Performing System Management Tasks*

ASSOCIATE

Gives a user operator control of a device class.

Syntax

ASSOCIATE *devclass*

Parameters

devclass The name of a logical device class configured with SYSGEN.

Operation Notes

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 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 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 ASOCTBL.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.

The operator commands, which may be made available to users through the ASSOCIATE command, are:

```
  ABORTIO           OUTFENCE
  ACCEPT           REFUSE
  ALTSPOOLFILE        REPLY
  DELETESPOOLFILE       RESUMESPOOL
  DISCRPS           SHUTQ
  DOWN            SPOOLER
  DOWNLOAD          STARTSPOOL
  FORMSALIGN         STOPSPOOL
  HEADOFF           SUSPENDSPOOL
  HEADON           UP
  OPENQ
```

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.

Use

This command may be issued from a session, program, or in BREAK. It may not be used from a job. Pressing **Break** has no effect on this command.

Example

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

```
  ASSOCIATE TAPE
```

Related Information

Commands	`DISASSOCIATE`
Manuals	*Performing System Operation Tasks*

BASIC

Interprets a compatibility mode BASIC/V program. BASIC/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately.

Syntax

BASIC[*commandfile*] [,[*inputfile*] [,*listfile*]]

Parameters

commandfile Actual file designator of the source file or device from which BASIC/V 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/V 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.

Operation Notes

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

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

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/V interpreter through input files that you have stored on disk. Files created using the editor must be kept with the UNN (unnumbered) option of the editor KEEP command. In this example, BASIC/V 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
```

Related Information

Commands	BASICGO, BASICOMP, BASICPREP
Manuals	*BASIC/V Compiler Manual*
	MPE Segmenter Reference Manual

BASICGO

Compiles, prepares, and executes a compatibility mode BASIC/V program. BASIC/V is not part of the HP 3000 Series 900 Computer System 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/V 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. For further information, refer to the "Implicit FILE Commands for Subsystems" discussion of the FILE command.

Operation Notes

This command compiles, prepares, and executes a compatibility mode program from a "fastsave" file created by the BASIC/V 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/V 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.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Example

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

```
BASICGO
$CONTROL USLINIT
$COMPILE MYPROG
$EXIT
```

The above example begins execution of the BASIC/V compiler, initializes the USL, compiles the program MYPROG, and then exits from the compiler.

Related Information

Commands	BASIC, BASICOMP, BASICPREP
Manuals	*BASIC/V Compiler Reference Manual*
	MPE Segmenter Reference Manual

BASICOMP

Compiles a compatibility mode BASIC/V program. BASIC/V is not part of the HP 3000 Series 900 Computer System 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 BASIC/V 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 a 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 MPE segmenter command BUILDUSL. Refer to the *MPE Segmenter Reference Manual* (30000-90011).
- By creating a new USL file with the MPE/iX 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 FILE command.

Operation Notes

The BASICOMP command compiles a program from a "fastsave" file generated by the BASIC/V interpreter. If a USL file is not specified, the BASIC/V compiler stores the object code in the default systemcdefined 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/V compiler to store the object code in this file by naming the USL file in the BASICOMP command line. Refer to "Examples."

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

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

```
BUILD OBJECT;CODE=USL
BASICOMP, OBJECT
$CONTROL USLINIT
$COMPILE MYPROG
$EXIT
```

The above example builds the USL file, begins execution of the BASIC/V compiler and specifies the USL named OBJECT, initializes the USL, compiles the fastsave program named MYPROG, and then exits from the compiler.

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

```
BASICOMP
$COMPILE MYRUN
$EXIT
```

The above example begins execution of the BASIC/V compiler, accepts commands from $STDINX, and specifies $OLDPASS the USL output and $STDLIST for listing output. It compiles from the fastsave file named MYRUN into a USL named $OLDPASS, and then exits from the BASIC/V compiler.

To run your program, enter:

```
PREPRUN $OLDPASS
```

Related Information

Commands	BASIC, BASICGO, BASICPREP
Manuals	*BASIC/V Compiler Reference Manual*
	MPE Segmenter Reference Manual

BASICPREP

Compiles and prepares a compatibility mode BASIC/V program. BASIC/V is not part of the HP 3000 Series 900 Computer System 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 BASIC/V 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. When *progfile* is omitted, the MPE segmenter creates the program file, which resides in the temporary file domain as `$OLDPASS`. To create your own program file, do so in one of two ways:

- By using the `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 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 `FILE` command.

Operation Notes

The `BASICPREP` command compiles and prepares a program for execution from a " fastsave" file generated by the BASIC/V interpreter. If the *progfile* parameter is omitted, the prepared program segments are stored in the systemcdefined temporary file `$OLDPASS`. To save the prepared program in a file other than `$OLDPASS`, either create a file and specify its file name on the `BASICPREP` command line, or specify a nonexistent *progfile*.

A program compiled and prepared with the `BASICPREP` command may be executed with the MPE/iX `RUN` command.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

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

```
BASICPREP,MYCOMDS
```

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

```
$CONTROL USLINIT SOURCE
$COMPILE MYPROG
$EXIT
```

The above example initializes the USL and lists the program, compiles the fastsave program MYPROG, and then exits from the compiler.

Related Information

Commands	BASIC, BASICGO, BASICOMP
Manuals	*BASIC/V Compiler Reference Manual*

BBASIC

Starts execution of the HP Business BASIC/V interpreter in compatibility mode. HP Business BASIC/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately.

Syntax

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

Parameters

commandfile Actual file designator of the source file or device from which HP Business BASIC/V 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 HP Business BASIC/V 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 `FILE` command.

Operation Notes

The `BBASIC` command is generally used for online programming in HP Business BASIC/V, but it can also be used to interpret HP Business BASIC/V programs submitted in batch mode. In batch mode, the HP Business BASIC/V `>EXIT` or `>::` command is required as the last statement in the command file. HP Business BASIC/V has its own online help facility.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the `RESUME` command continues the execution.

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 HP Business BASIC/V interpreter through input files that you have stored on disk. Files created using the editor must be kept with the UNN (unnumbered) option of the editor's KEEP command. In this example, HP Business BASIC/V 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
```

Related Information

Commands	BBASICGO, BBASICOMP, BBASICPREP
Manuals	*HP Business BASIC/XL Reference Manual*
	MPE Segmenter Reference Manual

BBASICGO

Compiles, prepares, and executes an HP Business BASIC/V program in compatibility mode. HP Business BASIC/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately.

Syntax

BBASICGO *infile* [*,listfile*]

Parameters

infile Actual file designator of the BSAVE file containing the HP Business BASIC/V 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 FILE command.

Operation Notes

This command compiles, prepares, and executes a program from a BSAVE file created by the HP Business BASIC/V 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 HP Business BASIC/V interpreter after it is saved by using the HP Business BASIC/V interpreter >SAVE *filename* command. The program then can be compiled, prepared, and executed with the BBASICGO command.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Example

To compile, prepare, and execute the HP Business BASIC/V program MYPROG and send the listing to the disk file LISTFL, enter:

```
BBASICGO MYPROG,LISTFL
```

Related Information

Commands	`BBASIC`, `BBASICOMP`, `BBASICPREP`
Manuals	*HP Business BASIC/XL Reference Manual*
	MPE Segmenter Reference Manual

BBASICOMP

Compiles an HP Business BASIC/V program in compatibility mode. HP Business BASIC/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately.

Syntax

BBASICOMP *infile* [,[*uslfile*] [,*listfile*]]

Parameters

infile Actual file designator of the BSAVE file containing the HP Business BASIC/V program to be compiled. 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 BBCUSL. 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 MPE segmenter command BUILDUSL. Refer to the *MPE Segmenter Reference Manual* (30000-90011).
- By creating a new USL file with the 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 FILE command.

Operation Notes

The BBASICOMP command compiles a source program stored in a BASIC SAVE file generated by the HP Business BASIC/V interpreter. The compiled program executes significantly faster than the corresponding interpreted version.

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

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile the HP Business BASIC/V program MYPROG into the USL named OBJECT, enter:

```
BBASICOMP MYPROG,OBJECT
```

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

```
BBASICOMP MYPROG
```

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

```
PREPRUN $OLDPASS
```

Related Information

Commands	BBASIC, BBASICGO, BBASICPREP
Manuals	*HP Business BASIC/XL Reference Manual*
	MPE Segmenter Reference Manual

BBASICPREP

Compiles and prepares an HP Business BASIC/V program in compatibility mode. HP Business BASIC/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately.

Syntax

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

Parameters

infile Actual file designator of the BSAVE file containing the HP Business BASIC/V program to be compiled. Formal file designator is BBCIN.

progfile Actual file designator of the program file to which the prepared program segments are written. When *progfile* is omitted, the MPE segmenter creates the program file, which resides 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 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 temporary job file of the correct size and type is created.

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 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 FILE command.

Operation Notes

The BBASICPREP command compiles and prepares a program from a BSAVE file generated by the HP Business BASIC/V interpreter. If you omit the *progfile* parameter, the prepared program segments are stored in the systemcdefined 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 file name on the BBASICPREP command line, or specify a nonexistent *progfile*.

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

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Example

To compile and prepare a program named MYPROG from the HP Business BASIC/V BSAVE file named MYCOMDS, and send the listing to the standard list device, enter:

```
  BBASICPREP MYCOMDS,MYPROG
```

Related Information

Commands	BBASIC, BBASICGO, BBASICOMP
Manuals	*HP Business BASIC/XL Reference Manual*
	MPE Segmenter Reference Manual

BBXL

Initiates execution of the HP Business BASIC/XL interpreter. HP Business BASIC/XL is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. (Native Mode)

Syntax

`BBXL[` *commandfile*`] [ ,[` *inputfile*`] [ ,[` *listfile*`] ] ] [ ;XL=`*xllist*`]`

NOTE This command follows the optional MPE/iX command line Syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

commandfile The name of an ASCII file that contains a set of HP Business BASIC/XL commands and/or statements. The formal file designator is `BASCOM`. Default is `$STDINX`.

inputfile Actual file designator of the file containing data input for a HP Business BASIC/XL program. 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 `BASOUT`. Default is `$STDLIST`.

xllist A quoted list of the executable libraries which is searched when resolving external procedure references during execution of a user's program.

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 `FILE` command.

Operation Notes

The `BBXL` command is generally used for online programming in HP Business BASIC/XL, but it can also be used to interpret HP Business BASIC/XL programs in batch mode. In batch mode, the HP Business BASIC/XL `>EXIT` or `>::` command is required as the last statement in the command file. HP Business BASIC/XL has its own online help facility.

NOTE This command is implemented as a command file. If you set the `HPPATH` variable to null (`SETVAR ""`), the command file is not executed and the command fails.

Use

This command may be issued from a session, job, or program. It is not available in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To enter commands and data from your standard input device, with the program listing and output transmitted to the standard output device (both of these are usually the terminal in interactive mode), use:

```
BBXL
```

You may also enter commands and statements to the HP Business BASIC/XL interpreter by using input files that you have stored on disk. Files created using the editor must be kept with the UNN (unnumbered) option of the editor's KEEP command. In this example, HP Business BASIC/XL interpreter commands and statements are entered 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.

```
BBXL MYCOMDS,MYDATA,MYLIST
```

If you have compiled a number of library procedures into an executable library named MYXL.MYGRP.MYACCT and wish to reference these in a program in the interpreter, use:

```
BBXL XL='MYXL.MYGRP.MYACCT'
```

Appropriate EXTERNAL and/or INTRINSIC statements in your program are used to define the formal parameters, and an alias, if required, for the external procedure in the executable library.

Related Information

Commands	BBXLCOMP, BBXLGO, BBXLLK
Manuals	*HP Business BASIC/XL Migration Guide HP Business BASIC/XL Reference Manual*

BBXLCOMP

Compiles an HP Business BASIC/XL program. HP Business BASIC/XL is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. (Native Mode)

Syntax

`BBXLCOMP` *textfile* [,[*objectfile*] [,*listfile*]]

NOTE This command follows the optional MPE/iX command line Syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile Actual file designator of the `BASIC SAVE` file (file code `1247` or `BSVXL`) containing the HP Business BASIC/XL program to be compiled. Formal file designator is `BBCIN`.

objectfile Actual file designator of the object file to which the object code is written. This file is stored in binary form and has a file code of `1461` or `NMOBJ`. If your program uses `GLOBAL COPTION RLFILE` then this file is a binary file with a file code of `1033` or `NMRL`. Its formal file designator is `BBCOBJ`. If the *objectfile* parameter is omitted, the object code is saved to the temporary file `$OLDPASS`.

If you specify *objectfile*, the compiler stores the object file in a permanent file of the correct size and type, and with the name you specified.

For an `NMOBJ` file, if a file of the same name already exists, the object code overwrites that file.

For an `NMRL` file, if `GLOBAL COPTION RLINIT` is used, then the relocatable library file is overwritten. If `GLOBAL COPTION RLINIT` is not used, then the new object code is added but previously written information remains.

If the compiler issues an error message telling you that a new or existing object file is too small, build the object file with a larger size and recompile to it.

You may use the MPE/iX `SAVE` command to store `$OLDPASS` as a permanent file under another name.

listfile The name of the file to which the compiler writes the program listing. This can be any ASCII file. The formal file designator is `BBCLIST`. If you do not specify *listfile*, the default is `$STDLIST`. `$STDLIST` is usually the terminal in a session or the printer in a batch job.

NOTE The formal file designators used in this command (BBCIN, BBCOBJ, 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 FILE command.

Operation Notes

The BBXLCOMP command compiles a source program stored in a BASIC SAVE file generated by the HP Business BASIC/XL interpreter. The compiled program executes significantly faster than the corresponding interpreted version.

Create a BASIC SAVE program source file from within the HP Business BASIC/XL interpreter by entering the program and using the HP Business BASIC/XL interpreter >SAVE *filename* command. Compile the source program in *filename* with the BBXLCOMP command, then link with the MPE/iX LINK command, and execute the program with the MPE/iX RUN command.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile the HP Business BASIC/XL source program in the file MYPROG into the NMOBJ file named OBJECT, enter:

```
BBXLCOMP MYPROG,OBJECT
```

If you do not specify an NMOBJ file, the BBXLCOMP command compiles your program, storing the object code in the default file $OLDPASS.

```
BBXLCOMP MYPROG
```

The above example runs the HP Business BASIC/XL compiler using the contents of MYPROG as the BASIC SAVE formatted source file. $OLDPASS is the default object file (NMOBJ) and $STDLIST is the default output listing.

If you now want to run your program, enter the LINK and RUN commands:

```
LINK
RUN $OLDPASS
```

This links the NMOBJ file and runs the program.

Related Information

Commands	BBXL, BBXLGO, BBXLLK

Manuals *HP Business BASIC/XL Migration Guide HP Business BASIC/XL Reference Manual*

BBXLGO

Compiles, links, and executes an HP Business BASIC/XL program. HP Business BASIC/XL is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. (Native Mode)

Syntax

BBXLGO *textfile* [,[*listfile*]] [;XL=*xllist*]

NOTE This command follows the optional MPE/iX command line Syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile Actual file designator of the BASIC SAVE file (file code = 1247 or BSVXL) containing the HP Business BASIC/XL 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.

xllist A quoted list of the executable libraries which is searched when resolving external procedure references when the program is loaded.

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 FILE command.

Operation Notes

This command compiles a BASIC SAVE file created by the HP Business BASIC/XL interpreter. The compiled program executes significantly faster than the corresponding interpreted version.

A BASIC SAVE program file is created from within the HP Business BASIC/XL interpreter by using the HP Business BASIC/XL >SAVE *filename* command. The program then can be compiled, linked, and executed with the BBXLGO command.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the `RESUME` command continues the execution.

Example

To compile, link, and execute the HP Business BASIC/XL program `MYPROG` and direct the listing to the disk file `LISTFL`, enter:

```
  BBXLGO MYPROG,LISTFL
```

Related Information

Commands	`BBXL`, `BBXLCOMP`, `BBXLLK`
Manuals	*HP Business BASIC/XL Migration Guide HP Business BASIC/XL Reference Manual*

BBXLLK

Compiles and links an HP Business BASIC/XL program. HP Business BASIC/XL is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. (Native Mode)

Syntax

BBXLLK *textfile* [,[*progfile*] [,*listfile*]]

NOTE This command follows the optional MPE/iX command line Syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile Actual file designator of the BASIC SAVE file (filecode 1247 or BSVXL) containing the HP Business BASIC/XL program to be compiled. Formal file designator is BBCIN.

progfile Actual file designator of the object file to which the Link Editor writes the linked program. If you do not specify *progfile*, the default is $NEWPASS, which is closed as $OLDPASS.

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 BBCLIST. If you do not specify *listfile*, the 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 FILE command.

Operation Notes

The BBXLLK command compiles and links a source program stored in a BASIC SAVE file generated by the HP Business BASIC/XL interpreter. If the *progfile* parameter is omitted, the linked program is written to the systemcdefined temporary file $OLDPASS. To save the linked program in a file other than $OLDPASS, specify the file name on the BBXLLK command line.

Create a BASIC SAVE program file from within the HP Business BASIC/XL interpreter, by using the HP Business BASIC/XL >SAVE *filename* command. The program may be compiled and linked with the BBXLLK command and executed with the MPE/iX RUN command.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Example

To compile and link a source program stored in the HP Business BASIC/XL BASIC SAVE file named MYSCR to the program file named MYPROG, and send the listing to the standard list device, enter:

```
BBXLLK MYSCR,MYPROG
```

Related Information

Commands	BBXL, BBXLCOMP, BBXLGO
Manuals	*HP Business BASIC/XL Migration Guide HP Business BASIC/XL Reference Manual*

BREAKJOB

Suspends an executing job. (Native Mode)

Syntax

```
BREAKJOB #Jnnn
```

Parameters

#J*nnn*	A job number.

Operation Notes

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

When you issue the BREAKJOB command for a job that controls a nonshareable 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, operate on suspended jobs. The SHOWJOB command, which lists all jobs, displays SUSP for those in the suspended state. To list suspended jobs only, enter SHOWJOB SUSP.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It is executable only from the console unless distributed to users with the ALLOW command, or if JOBSECURITY is set to LOW.

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.ACCT
```

Related Information

Commands	ALTJOB, ABORTJOB, RESUMEJOB, SHOWJOB, STREAM
Manuals	*Performing System Operation Tasks*

BUILD

Creates and immediately allocates a new empty file on disk.

Syntax

```
BUILD filereferencer
[;REC=[ [ recsize] [ ,[ blockfactor] [ ,[ F U V B ] [ ,BINARY ,ASCII ] ] ] ] ]
[ ;CCTL [ ;NOCCTL] ]
[ ;TEMP] [ ;DEV= [ dsdevice# dsdevice#device [ device] ] ]
[ ;CODE=filecode] BUILD [ ;DISC=[ [ numrec] [ ,[ numextents] [ ,initialloc] ] ] ]
[ ;RIO ;NORIO ] [ ;MSG ;CIR ;STD ;KSAMXL ;SPOOL ]
[ ;ULABEL=numlabels] [ ;KEY={ ^filereference keyinfo } ]
[ ;FIRSTREC=recnum] [ ;REUSE ;NOREUSE ]
[ ;langid={ langid langname } ]
[ { ;DEFBLK ;OPTMBLK }]
```

Parameters

filereference Actual name of the file to be created. The *filereference* can be either in MPE of HFS Syntax.

MPE Syntax

If the *filereference* does not begin with a dot or a slash, it is parsed according to the MPE Syntax and has the following format:

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

MPE names must contain from one to eight alphanumeric characters, beginning with an alphabetic character. If *acctname* is specified, you must have create directory (CD) access to the target group in the account. The default *groupname* and *acctname* are the logon group and account.

HFS Syntax

If the *filereference* begins with a dot (.) or a slash (/), it is parsed according to the HFS Syntax. In this case the *filereference* can consist of 1 to 253 characters for relative pathnames (for example, `./253chars`), and 254 characters for absolute names (for example, `/254chars`).

The following Syntax rules apply:

- File names are not upshifted.
- File names can be up to 254 characters in length for absolute pathnames, and 253 characters for relative pathnames.
- File names can begin with, and contain, any of the following characters:

  a-z, A-Z, 0-9, _, .

- File names can contain (but not begin with) a dash (-).

File names are of the form

path/filename

where the *path/filename* combination may have a maximum of 255 characters.

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. For byte-stream files, *recsize* is 1 byte.

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 length 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 is `DISC` with 1023 records.

For example, a fixed length ASCII file with a record size specified as 11 bytes has only 11 bytes available for data in each logical record. However, to determine actual block size, 12 bytes is used for the record size (block size = 12 bytes multiplied by the *blockfactor*). If the file is specified as a binary file, the 11 bytes are rounded up to 12 bytes (6 words), all of which are available for each logical record.

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.

For byte-stream files, *blockfactor* is set to 1.

`F, U, V` or `B` Defines the length of the records of the file. A file may contain fixed length records (`F`), undefined length records (`U`), variable length records (`V`) or byte-stream format (`B`). For disk 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 is created as a temporary file and is saved in the job/session temporary file domain when closed. 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 file name of the remote computer and the remote device file name.

device Either the *devclass* or *ldev* on which the file is to reside. A device class name (*devclass*), such as `DISC` consists of up to eight alphanumeric characters beginning with an alphabetic character. The `DEV=` parameter does not accept device names, volume classes, or volume names. When you specify *devclass*, the file is allocated to any available device in that class. If you are opening a file destined for a mountable volume, you must specify a device class that 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 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 32,767 or a mnemonic name for this parameter, certain reserved integers and mnemonics have particular system defined meanings.

Default is the unreserved file code of 0.

Using `1090` (`LOG`) as your designated file code 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 maximum for fixed length and undefined length records is 2,147,483,647. The default is 1023.

numextents Maximum number of disk extents. You may specify a value of -1, or any number from 1 to 32. Default is 8.

initialloc Number of extents to be initially allocated to the file at the time that it is opened. If you specify -1 for this parameter, the default value is used.

`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 length record. The default, `NORIO`, creates a nonrelative 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 is not accessed in RIO mode; otherwise, RIO access is used with RIO files. Special operations on RIO files, such as replicating an RIO file, set NOBUF access. Refer to the *Accessing Files Programmer's Guide* for a discussion of relative I/O.

STD, MSG, CIR, KSAMXL, SPOOL Defines the type of file.

The default is STD (standard MPE/iX disk file). You do not need to specify STD; in fact, if you do specify it, you will see the error message `The STD keyword is not appropriate in the context of a BUILD command. (CIERR 216)`.

A MSG (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 are logically deleted and/or are appended to the end of the file.

CIR (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.

KSAMXL specifies a native mode KSAM file (KSAM XL file).

SPOOL specifies an unlinked output spool file. The default *outpri* on the spool file is 8; the default number of copies is 1. The unlinked output spool file must be created on a disk device. Specify the target printer device at SPOOLF...;PRINT time; if you do not, an error results.

The characteristics of a file created with the SPOOL keyword are:

- variable length records of 1008 bytes each
- a blocking factor of 1
- ASCII format
- permanent file
- record limit of 1023
- undefined maximum number of extents with 0 extents initially allocated

These characteristics override any other characteristics, such as binary format, which may be specified.

numlabels The number of user label records to be created for the new file. Up to 255 labels can be specified. This parameter applies to any type of file.

^filereference or keyinfo filereference is a file containing key information. This parameter only applies to new KSAM files; it is required for new KSAM files. The caret (^) indicates that the contents of the file will be used.

keyinfo has the following format:

```
;KEY=
 (keytype,keylocation,keysize
  [,DUP|RDUP];
      .
      .
  keytype,keylocation,keysize
  [,DUP|RDUP])
```

One key specification (*keytype, keylocation, keysize* [,DUP|RDUP] must be included for each key in the KSAM file. The first occurrence of the key specification describes the primary key; each subsequent key specification describes an alternate key. There may be up to 15 alternate key specifications in addition to the primary key description.

keytype KSAM key type, specified as BYTE, INTEGER, REAL, IEEEREAL, NUMERIC, PACKED, OR *PACKED. Specify the whole word or only the first letter; valid abbreviations are B, I, R, E, N, P, and *. If more than one letter is specified, the word must be spelled correctly.

keylocation Location of the first byte of the key within the data record counting from the first byte in the record. The first byte in the data record is always numbered 1. Only one key can start at the same location. This parameter applies only to KSAM files.

keysize Length of the KSAM key in bytes. The length depends on *keytype* as follows:

```
BYTE       1 to 255 bytes
INTEGER     1 to 255 bytes
REAL       1 to 255 bytes
IEEEREAL     4, 8, or 16 bytes
NUMERIC     1 to 28 bytes
PACKED      1 to 14 bytes (odd number of digits)
*PACKED     2 to 14 bytes (even number of digits)
```

This parameter is required for all key types.

DUP OR RDUP These two options apply only to KSAM files. The DUP option allows you to specify that duplicate key values are permitted. If DUP is not specified, records with duplicate key values are rejected and an error message is issued when such records are written to the file. When the DUP option is used, each new duplicate key is inserted at the end of the duplicate key chain. This maintains the chronological order of the duplicate keys.

The RDUP option specifies that duplicate keys are allowed and to be inserted randomly in the duplicate key chain. This method makes insertion of such keys faster, but does not maintain the chronological order of the duplicate key chain. The default is that duplicate keys are not allowed.

recnum Determines whether record numbers in the new KSAM file are to start with zero or one. If the integer 1 is specified, records are numbered beginning with 1; otherwise, they start with 0. The only acceptable values for *recnum* are 1 and 0. This option can only be used for new KSAM files.

REUSE or NOREUSE The REUSE option forces KSAM files to reuse deleted record space. The REUSE option forces RDUP to be set to TRUE for all keys.

If the NOREUSE option is used, deleted record space is not reused. If the DUP option is specified for a key, duplicate records are placed chronologically at the tail end of the file. The default is NOREUSE.

langid An integer number indicating the native language of the KSAM file to be built. The default is 0, or NATIVE-3000. The language must be currently configured on the system. See the Native Language documentation for more information.

langname The name indicating the native language for the KSAM file to be built. The default language is NATIVE-3000. The language must be currently configured on the system. See the Native Language documentation for more information.

DEFBLK or OPTMBLK These two options apply only to KSAM files. DEFBLK specifies that the data block size will be the default data block size of 4096 bytes. OPTMBLK specifies that KSAMXL will select the optional data block size based on the record size. The default is DEFBLK.

NOTE The file system uses the values specified on the BUILD command line to compute other characteristics of the file. Therefore, the values (or default values) may be valid within their respective fields, but may cause overflow errors in the computation of internally needed file specifications.

Operation Notes

This command builds a new file on disk. If it is an ASCII file, the initially allocated 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 save file (SF) capability and SAVE access in the group to which the new file belongs. 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 belongs. If the group's home volume set is not mounted, BUILD implicitly generates a volume set reservation request. If the volume is not recognized by the system, the command fails. Refer to *Volume Management Reference Manual*.

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 0 extent initially allocated. This is equivalent to entering:

```
BUILDfilename;REC=128,1,F,BINARY;DEV=DISC;DISC=1023,8,
```

Use

This command may be issued from a session, a job, a program, or in break mode. Pressing **Break** has no effect on this command.

Examples

The following example creates a permanent disk file named WORKFILE, which can reside on any disk. 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 initially allocated.

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

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

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

Related Information

Commands	COPY, LISTFILE, LISTF, LISTFTEMP, PURGE, RENAME
Manuals	*MPE/iX Intrinsics Reference Manual*
	Native Mode Spooler Reference Manual

BYE

Ends an interactive session. (Native Mode)

Syntax

```
BYE
```

Parameters

None.

Operation Notes

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

```
  CPU=48. CONNECT=35. FRI, MAY 4, 1987, 10:56 PM
```

If you enter the HELLO command without logging off your current session, MPE/iX terminates your current session and immediately initiates a new one. If you are logged on to the computer with a telephone connection, and you hang up before terminating your session, MPE/iX issues a BYE command automatically.

If you enter the BYE command before initiating a session on the system, no system message is displayed.

Use

This command may be issued from a session. It may not be used from a job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

To terminate a session, enter:

```
  BYE
```

Related Information

Commands	HELLO
Manuals	None

3 Command Definitions C-E

CALC

Evaluates an expression. (Native Mode)

Syntax

CALC *expression*

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

expression The expression to be evaluated.

Operation Notes

The CALC command evaluates *expression* and displays the result to $STDLIST. Expressions can yield integer, string, or Boolean results. Integer results are displayed in decimal, hexadecimal ($ prefix), and octal (% prefix) notations. Boolean expressions are displayed as TRUE or FALSE. The variable HPRESULT is set to the result of the last *expression* evaluated by CALC. The type of HPRESULT changes depending on the type of result generated by CALC.

Table 3-1 lists some of the logical operators for the CALC command. Enter :HELP expressions for more information

Table 3-1 Logical Operators - The CALC Command

Logical operators:	AND, OR, XOR, NOT
Boolean functions and values:	BOUND, TRUE, FALSE, ALPHA, ALPHANUM, NUMERIC, ODD
Comparison operators:	=, <>, <, >, <=, >=
Bit manipulation operators:	LSL, LSR, CSR, CSL, BAND, BOR, BXOR, BNOT
Arithmetic operators:	MOD, ABS, * , / , + , -, ^ (exponentiation)
Functions returning strings:	CHR, DWNS, UPS, HEX, OCTAL, INPUT, LFT, RHT, RPT, LTRIM, RTRIM, STR
Functions returning integers:	ABS, LEN, MAX, MIN, ORD, POS, TYPEOF
Other functions:	FINFO, SETVAR

The operands you may use are any variable, integer, string, Boolean constant, or the system-reserved words WARN, FATAL, SYSTEM, and OK. You may form compound logical expressions using the AND, NOT, XOR, and OR logical operators, optionally nested within parentheses.

Do not use the FINFO function with the CALC command for remote files. It ignores their existence and returns incorrect information.

Use

This command is available in a session, job, program, or in BREAK. Pressing **Break** terminates the INPUT( ) function.

Example

The result of CALC sample depends on the value entered for sample and on the type of the value, as shown in Table 3-2

Table 3-2 Results of CALC

sample	Displayed (HPRESULT)	Type
5*10-7	43, $2B, %53	Integer
LEN("abc")	3, $3, %3	Integer
UPS("Abc")	ABC	String
1=1	TRUE	Boolean
MAX(1,0,abs(-12),10)	12, $c, %14	Integer

Related Information

Commands	DELETEVAR, ELSEIF, IF, SETJCW, SETVAR, SHOWJCW, SHOWVAR, WHILE
Manuals	Appendix B, "Expression Evaluator Functions"
	Command Interpreter Access and Variables Programmer's Guide

CCXL

Compiles an HP C/iX program. HP C/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP C/iX is installed on your system. (Native Mode)

Syntax

CCXL[*textfile*] [,[*objectfile*] [,[*listfile*]]] [;INFO=*quotedstring*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile The name of the text file that contains the source code to be compiled. This is an ASCII file that you prepare with an editor such as EDIT/3000. The formal file designator is CCTEXT.

If you are running HP C/iX from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile*, then the default file is $STDIN. $STDIN is the current input device, usually your terminal.

When *textfile* is your terminal, you can enter source code interactively. When you have entered all the source code, type a colon (:) to end the interactive input.

objectfile Actual file designator of the object file to which the object code is stored. This file is in binary form and has a file code of 1461 or NMOBJ. Its formal file designator is CCOBJ. If the *objectfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS.

If you specify *objectfile*, the compiler stores the object file in a permanent file of the correct size, type, and name you specified. If a file of the same name already exists, the object code overwrites that file.

If the compiler issues an error message telling you that a new or existing object file to which you are trying to compile is too small, build a larger object file and recompile to it.

You may use the MPE/iX SAVE command to store $OLDPASS as a permanent file under another name.

listfile The name of the file on which the compiler writes the program listing. It can be any ASCII file. The default is `$STDLIST.$STDLIST` is usually the terminal from a session or the printer from a batch job. The formal file designator is `CCLIST`.

If *listfile* is `$NULL` or a file other than `$STDLIST`, the compiler displays on `$STDLIST` those lines that contain errors.

quotedstring A string of no more than 1024 characters (including the single or double quotation marks that enclose it).

The *quotedstring* is used to pass initial compiler options to the compiler program. Options must be delimited by blank spaces.

NOTE The formal file designators used in this command (`CCTEXT`, `CCOBJ`, and `CCLIST`) cannot be backreferenced as actual file designators in the command parameter list.

Operation Notes

The `CCXL` command compiles an HP C/iX program and stores the object code in a permanent file (*objectfile*) or in `$OLDPASS` if you do not specify an object file. If *textfile* is not specified, the compiler expects the source program to be entered from your standard input device. If you do not specify *listfile*, the compiler sends the program listing to your standard device and identifies it by the formal file designator `CCLIST`.

NOTE This command is implemented as a command file. If you set the `HPPATH` variable to null (`SETVAR HP PATH ""`), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the `RESUME` command continues the execution.

Examples

The following example compiles an HP C/iX program entered from your standard input device and stores the object program in the object file $OLDPASS. The listing is then sent to your standard list device.

```
CCXL
```

The next example compiles an HP C/iX program contained in the disk file SOURCE and stores the object program in the object file OBJECT. The program listing is stored in the disk file LISTFILE.

```
CCXL SOURCE,OBJECT,LISTFILE
```

Program development in native mode uses the MPE/iX LINK command, not the MPE V/E PREP command. This produces a significant change in the method of linking code. In MPE/iX, you must compile the source files into separate object files and then use the Link Editor to link the two object files into the program file, as in this example:

```
CCXL MAIN, OBJMAIN
CCXL SUB, OBJSUB
LINK FROM=OBJMAIN,OBJSUB;TO=SOMEPROG;RL=LIBCINIT.LIB.SYS
RUN SOMEPROG
```

Related Information

Commands	CCXLGO, CCXLLK, RUN, LINK, XEQ, LINKEDIT Utility
Manuals	*HP C Programmer's Guide*

CCXLGO

Compiles, links, and executes an HP C/iX program. HP C/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP C/iX is installed on your system. (Native Mode)

Syntax

CCXLGO[*textfile*] [,[*listfile*]] [;INFO=*quotedstring*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile The name of the text file that contains the source code to be compiled. This is an ASCII file that you prepare with an editor such as EDIT/3000. The formal file designator is `CCTEXT`.

If you are running HP C/iX from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile*, then the default file is `$STDIN`. `$STDIN` is the current input device, usually your terminal.

When *textfile* is your terminal, you can enter source code interactively. When you have entered all the source code, type a colon (`:`) to end interactive input.

listfile The name of the file on which the compiler writes the program listing. It can be any ASCII file. The default is `$STDLIST`. `$STDLIST` is usually the terminal from a session or the printer from a batch job. The formal file designator is `CCLIST`.

If *listfile* is `$NULL` or a file other than `$STDLIST`, the compiler displays on `$STDLIST` those lines that contain errors.

quotedstring A quoted string of no more than 1024 characters (including the single or double quotation marks that enclose it).

The *quotedstring* is used to pass initial compiler options to the compiler. Options must be delimited by blank spaces.

NOTE The formal file designators used in this command (`CCTEXT` and `CCLIST`) cannot be backreferenced as actual file designators in the command parameter list.

Operation Notes

The CCXLGO command compiles, links, and executes an HP C/iX program. If *textfile* is omitted, the compiler expects input from your standard input device. If you do not specify *listfile*, the compiler sends the program listing to the formal file designator CCLIST (default is $STDLIST).

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. The Link Editor purges the object file and writes the linked program to $OLDPASS, which is then executed and may be executed repeatedly.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile, link, and execute an HP C/iX program entered from your standard input device, with the program listing sent to your standard list device, enter:

```
CCXLGO
```

To compile, link, and execute an HP C/iX program from the disk file SOURCE and send the program listing to the file LISTFILE, enter:

```
CCXLGO SOURCE,LISTFILE
```

Related Information

Commands	CCXL, CCXLLK, RUN, LINK, XEQ, LINKEDIT Utility
Manuals	*HP C Programmer's Guide*

CCXLLK

Compiles and links an HP C/iX program. HP C/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP C/iX is installed on your system. (Native Mode)

Syntax

CCXLLK[*textfile*] [,[[*progfile*]] [,[*listfile*]]] [;INFO=*quotedstring*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile The name of the text file that contains the source code to be compiled. This is an ASCII file that you prepare with an editor such as EDIT/3000. The formal file designator is `CCTEXT`.

If you are running HP C/iX from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile*, then the default file is `$STDIN`. `$STDIN` is the current input device, usually your terminal. When *textfile* is your terminal, you can enter source code interactively. When you have entered all the source code, type a colon (`:`) to end the interactive input.

progfile The name of the program file on which the MPE/iX linker writes the linked program. If you omit the *progfile* parameter, the program is saved to the temporary file `$OLDPASS`.

listfile The name of the file on which the compiler writes the program listing. It can be any ASCII file. The default is `$STDLIST`. `$STDLIST` is usually the terminal from a session or the printer from a batch job. The formal file designator is `CCLIST`.

If *listfile* is `$NULL` or a file other than `$STDLIST`, the compiler displays on `$STDLIST` those lines that contain errors.

quotedstring A string of no more than 1024 characters (including the single or double quotation marks that enclose it).

The *quotedstring* is used to pass initial compiler options to the compiler. Options must be delimited by blank spaces. For a list of options, refer to the *HP C/iX Reference Manual* (31506-90005).

NOTE The formal file designators used in this command (`CCTEXT` and `CCLIST`) cannot be backreferenced as actual file designators in the command parameter list.

Operation Notes

The CCXLLK command compiles and links an HP C/iX program into a file on disk. If you do not specify *textfile*, the compiler expects input from the current input device. If you do not specify *listfile*, the compiler sends the listing output to the formal file designator CCLIST (default $STDLIST).

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. Link Editor overwrites *progfile* and writes the linked program to $OLDPASS, if *progfile* is omitted, which can then be executed.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles and links an HP C/iX program entered through your standard input device and stores the linked program in the file $OLDPASS. The listing is printed on your standard list device:

```
CCXLLK
```

To compile and link an HP C/iX source program from the source file SOURCE, store it in PROG, and send the listing to your standard list device, enter:

```
CCXLLK SOURCE,PROG
```

Related Information

Commands	CCXL, CCXLGO, RUN, LINK, XEQ, LINKEDIT Utility
Manuals	*HP C Programmer's Guide*
	HP C/iX Reference Manual

CHANGELOG

Changes the user logging file without stopping or interrupting the logging process.

Syntax

CHANGELOG ***logid***[;DEV=*device*]

Parameters

logid Name of the currently active user logging process. This name may contain from one to eight alphanumeric 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 either DISC or TAPE. Default is DISC.

Operation Notes

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 enable automatic logging with the ALTLOG or GETLOG command, however, the only device available for logging is the default, DISC.

If a log file is restricted to a single volume or volume class when it is created with the BUILD command, then successive log files created by User Logging will have the same restriction.

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 tries to open a new file half the size of the old file. It repeats this process until a new file is opened successfully, or until the size is less than 256 records. In the second case, user logging terminates.

If the system opens a new log file, it immediately writes 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 record written to the new file contains the fully qualified identifier of the old logging file. Changelog records also contain the device type of the logging file to which the changelog refers.

The following message is 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 advising the operator to mount the new log file appears on the console:

```
 Mount new tape volume for changelog of logid AAA
 (ulogmsg 40).
```

Normally 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 enable an automatic CHANGELOG, thereby eliminating the need to issue the CHANGELOG command manually. Refer to the ALTLOG and GETLOG commands in this chapter.

To use CHANGELOG (manually or automatically), end the first user logging file name with the numeric characters 001 (for example, *fname001*). This establishes a naming convention that works in conjunction with the file set number to generate sequential file names independently. New file names consist of the file name 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 resets to 000 and begins incrementing all over again.

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

NOTE The logging process specified by *logid* must be in an ACTIVE state. If the logging process is in any other state, such as RECOVERING, STOP, INITIALIZING, or if the logging process has another CHANGELOG pending, the command terminates in an error state. The ALTLOG command permits changing the log file for an inactive logging process. ALTLOG, however, does not provide a way to link log files into a set.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

You must be the *logid* creator or have system manager (SM) or system supervisor (OP) capability to use the CHANGELOG command. User logging (LG) capability is also required.

Example

If you are running a logging process with a *logid* of KATHY, logging to logfile KLOG001, and you want to close the current logfile and log to a new logfile, KLOG002, without interrupting the logging process, enter:

```
 CHANGELOG KATHY
```

Related Information

Commands `ALTLOG`, `GETLOG`, `LISTLOG`, `LOG`, `OPENLOG`, `RELLOG`, `SHOWLOG`, `SHOWLOGSTATUS`

Manuals *User Logging Programmer's Guide*

CHDIR

Changes the process' current working directory (CWD). (Native Mode)

Syntax

CHDIR[[DIR=] *dir_name*] [;SHOW | NOSHOW]

Parameters

dir_name The name of the directory you want to change to, which is assumed to be an MPE name unless you specify otherwise. To change to an HFS-named directory, begin *dir_name* with a dot (.) or a slash (/). The *dir_name* may not end in a slash, and using wildcards is not allowed.

This parameter is optional. If you omit *dir_name*, CHDIR switches you to your logon directory, which is your logon group in the form /LOGON_ACCOUNT/LOGON_GROUP in all uppercase letters.

SHOW Displays the absolute pathname of the new directory on $STDLIST. SHOW is the default.

NOSHOW Does not display the absolute pathname.

Operation

The CHDIR command changes the process' current working directory to *dir_name* or to the logon group, if you omit *dir_name*. You can change the CWD to any HFS directory if you precede *dir_name* with a dot (.) or a slash (/) or to an MPE account or group to which you have the appropriate permission.

Issuing the CHDIR command does not give users access to files in a directory (or group and account) that they would not otherwise have. That is, it has no affect on file access permissions.

The CWD is a process-local attribute, which means that CHDIR changes the CI's CWD for the life of that CI process or until another CHDIR command is issued. When CHDIR is executed programmatically from a child process of the CI (e.g., HPEDIT), only that process' CWD is changed; the CWD of the parent process (in this example, the CI) remains the same.

CHDIR does not post any accounting information: Connect and CPU time are still accounted to the user's logon account and group.

HPCWD is a read-only, CI string variable that contains the name of the current working directory in HFS syntax. At logon, HPCWD contains */account_name/logon_group_name*. The CHGROUP command causes the HPCWD variable to be set to */account/group_changed_to*.

The table on the next page summarizes the differences and similarities between the CHDIR and CHGROUP commands.

Affects	CHGROUP	CHDIR
Accumulation of CPU and Connect times	yes	no
Set of accessible files	yes	no
CWD of process	yes	yes
HPCWD variable	yes	yes
Disk space accumulation	yes	no

Use

The CHDIR command may be invoked from a job, a session, a program, or in Break. Pressing **Break** has no effect on this command. You must have traverse directory entries (TD) permission to each directory component in *dir_name* (refer to the ALTSEC command in this chapter for more information on directory permissions.) The CWD is not changed if the CHDIR command fails.

Examples

The following example shows the command entry to change to the directory dir1 in the MYGRP group in the MYACCT account.

```
CHDIR /MYACCT/MYGRP/dir1
```

The following example shows the command entry to change to the MPE group level (AGROUP) in the MYACCT account.

```
CHDIR /MYACCT/AGROUP
```

The following example shows the command entry to change to a directory named My_dir. In this example, My_dir is a relative pathname and it is subdirectory in the current working directory (CWD).

```
CHDIR ./My_dir
```

The following example shows the command entry to change to a directory named john, in the group JONES, in the account MYACCT, by specifying the full pathname.

```
CHDIR /MYACCT/JONES/john
```

In the following example, a change is made to a directory named final by specifying the relative pathname. The variable HPCWD displays the current working directory after the change is made.

```
CHDIR ./es/final
SHOWVAR HPCWD
```

```
HPCWD = /MYACCT/JONES/john/es/final
```

Related Information

Commands	`CHGROUP`, `FINDDIR` (UDC), `LISTFILE`, `LISTDIR` (UDC), `NEWDIR`, `PURGEDIR`
Manuals	*Performing System Management Tasks*

CHGROUP

Switches you from the current group to another group within the logon account to which you are allowed access. (Native Mode)

Syntax

CHGROUP[[*groupname*] [/*grouppass*]]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

groupname The name of the group to which the user is switched. If the parameter is omitted, the user is switched to the home group.

grouppass The password of the group you are switching to, if it is assigned a password. In a session, if the target group has a password and you fail to supply one on the command line, MPE/iX will prompt you to enter one. You have three tries to enter the correct password before the command fails.

In a batch job, if the target group has a password and you fail to supply one, MPE/iX issues an error message "INCORRECT PASSWORD (CIERR 1441)" and the job fails.

In either case, when you switch to your home group, you may omit the password.

Operation Notes

This command changes the user's current group to *groupname*. The entire command interpreter environment is preserved (temporary files, file equations, cataloged UDCs, and variables). The user must know the password, if any, for *groupname*. In a session, if a password is associated with *groupname*, and the user fails to supply a *grouppass*, the system prompts the user to enter one. In a job, if a password is associated with *groupname*, and the user fails to supply a *grouppass*, the error message `INCORRECT PASSWORD (CIERR 1441)` is issued and the job fails.

The `CHGROUP` and `CHDIR` commands both change their process' CWD. However, `CHDIR` does not post any accounting information, and `CHGROUP` affects the CWD of every process in the job/session structure. Connect and CPU times are still accounted to the user's logon account and logon group.

Use

This command is available in a session or a job, but not in BREAK or from a program. Pressing **Break** has no effect on this command.

Examples

To switch the user from the current group to the group called GORODA, enter:

```
CHGROUP GORODA
```

To switch the user from the current group to the group called GORODA, with the assigned password MUSASHI, enter:

```
CHGROUP GORODA/MUSASHI
```

To switch the user from the current group to the user's home group, enter:

```
CHGROUP
```

Related Information

Commands	CHDIR, HELLO
Manuals	None

COB74XL

Compiles an HP COBOL II/iX program using the 1974 ANSI standard entry point and creates an object file. HP COBOL II/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP COBOL II/iX is installed on your system. (Native Mode)

Syntax

COB74XL [*textfule*]

[,[*objectfile*][,[*listfile*][,[*masterfile*][,*newfile*]]]]

[;INFO=*quotedstring*][;WKSP=*workspacename*]

[;XDB=*xdbfilename*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile The name of the file that contains the source code that is to be compiled. This can be any ASCII or toolset access method (TSAM) file that you prepare with an editor such as EDIT/V. The formal file designator is COBTEXT.

If you are running HP COBOL II/iX from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile*, the default file is $STDIN. $STDIN is the current input device, usually your terminal.

objectfile Actual file designator of the object file, which is the output of the compiler. This file is stored in binary form and has a file code of either NMOBJ (1461) or NMRL (1033). Its formal file designator is COBOBJ. If the *objectfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS, if it exists, or to $NEWPASS, which then becomes $OLDPASS.

If you specify *objectfile*, the compiler stores the object file in a permanent file of the correct size, type, and name you specified.

If either a file of the same name or the default file $OLDPASS already exists, the new object code overwrites the old if the file code is NMOBJ or is appended to the old if the file code is NMRL. If the file code is NMRL, any existing version of the code module is first purged.

The functionality of NMRLS closely maps to the MPE/V USLS. Refer to the *HP COBOL/XL Programmer's Guide* (31500-90002) for information on the RLINIT and RLFILE commands that cause creation of an NMRL by default or initialization.

The compiler may issue an error message telling you that a new or existing object file is too small to contain the compiler's output or number of modules. In that case you must build a larger file or use the Link Editor to clean the NMRL. You may then recompile to the new file.

You may use the MPE/iX SAVE command to store $OLDPASS as a permanent file under another name.

listfile The name of the file to which the compiler writes the program listing. This can be any ASCII file. The formal file designator is COBLIST. If you do not specify *listfile*, the default is $STDLIST. $STDLIST is usually the terminal in a session or the printer in a batch job.

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 the *masterfile*. This can be any ASCII output file. Formal file designator is COBNEW. Default is that no file is written.

quotedstring A quoted string of no more than 255 characters, including the single or double quotation marks that enclose it, that specifies compile time options.

The *quotedstring* string may be used to pass dollar sign ($) commands to the compiler: "$command1$command2$command3...". The $ must be the first character in the string, and it must be used to separate multiple commands. To extend the *quotedstring* string over more than one physical line make an ampersand (&) the last character of one line and continue the *quotedstring* string onto the next physical line. Each $ command is limited in length to the same size as in the source file:

```
COB74XL SALARIES,SALPRG;INFO="$CONTROL &
BOUNDS,MAP,VERBS$SET&$X9=ON"&
COB74XL ACCOUNTS;INFO="$DEFINE %A=5#"
```

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

xdbfilename Actual file designator for the file to be used by the symbolic debugger (XDB). This is a permanent file created by the compiler that contains the listing of the source files. The formal file designator is COBXDB.

If this file exists, then it must be in a special format created by a previous compile using this option. In this case, it is first purged. If the file is of the wrong type, the compile is not attempted. The user must either use a different name or purge the file.

Once the file is created, XDB expects the fully qualified name of the file to be unchanged. A FILE equation could be used if the file is renamed.

Operation Notes

The COB74XL command compiles an HP COBOL II/iX program into an object file on disk. If you do not specify *textfile*, HP COBOL II/iX expects your input from your standard input device. If you do not specify *listfile*, HP COBOL II/iX sends the program listing to the current list device.

You cannot backreference the formal file designators used in this command (COBTEXT, COBOBJ, COBLIST, COBMAST, COBNEW, COBWKSP, and COBXDB) 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.

NOTE This command is recognized only if HP COBOL II/iX is installed on your system. This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program but not in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile an HP COBOL II/iX program stored in the file SOURCE into an object file called OBJECT, and send the listing to the disk file LISTFL, enter:

```
COB74XL SOURCE,OBJECT,LISTFL
```

Program development in native mode uses the MPE/iX LINK command, not the MPE V/E PREP command. This produces a significant change in the method of compiling code. For example, if you have created a program called MAIN and a subprogram called SUB, each contained in a separate file, you might choose to append the code from SUB to SOMEUSL in MPE V/E, like this:

```
COBOLII MAIN, SOMEUSL
COBOLII SUB, SOMEUSL
PREP SOMEUSL, SOMEPROG
RUN SOMEPROG
```

However, the LINK command (in MPE/iX native mode) does not append SUB. On MPE/iX, you must compile the source files into separate object files and then use the Link Editor to link the two object files into the program file, as in this example:

```
COB74XL MAIN, OBJMAIN
COB74XL SUB, OBJSUB
LINK FROM=OBJMAIN,OBJSUB;TO=SOMEPROG
RUN SOMEPROG
```

On the other hand, if an NMRL is used instead of an NMOBJ, the above can be simplified to the following:

```
BUILD RLFILE;DISC=10000;CODE=NMRL
COB74XL MAIN, RLFILE
COB74XL SUB, RLFILE
LINK RLFILE,SOMEPROG
RUN SOMEPROG
```

Related Information

Commands	COB74XLG, COB74XLK, LINK, RUN, XEQ, LINKEDIT Utility
Manuals	*HP COBOL II/XL Reference Manual*
	HP COBOL II/XL Programmer's Guide
	HP Link Editor/iX Reference Manual

COB74XLG

Compiles, links, and executes an HP COBOL II/iX program using the ANSI 1974 standard entry point. HP COBOL II/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP COBOL II/iX is installed on your system. (Native Mode)

Syntax

```
COB85XLG[textfile]
[ ,[ listfile] [ ,[ masterfile] [ ,newfile] ] ] ]
[ ;INFO=quotedstring] [ ;WKSP=workspacename]
[ ;XDB=xdbfilename]
```

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile The name of the file that contains the source file that is to be compiled. This can be any ASCII or toolset access method (TSAM) file. The formal file designator is COBTEXT.

If you are running HP COBOL II/iX from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile*, the default file is $STDIN. $STDIN is the current input device, usually your terminal.

listfile The name of the file to which the compiler writes the program listing. This can be any ASCII file. The formal file designator is COBLIST. If you do not specify *listfile*, the default is $STDLIST. $STDLIST is usually the terminal in a session or the printer in a batch job.

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 *masterfile*. This can be any ASCII output file. Formal file designator is COBNEW. Default is that no file is written.

quotedstring A quoted string of no more than 255 characters, including the single or double quotation marks that enclose it, that specifies compile time options.

The *quotedstring* string may be used to pass dollar sign ($) commands to the compiler: `"$command1$command2$command3..."`. The $ must be the first character in the string, and it must be used to separate multiple commands. To extend the *quotedstring* string over more than one physical line make an ampersand (&) the last character of one line and continue the *quotedstring* string onto the next physical line. Each $ command is limited in length to the same size as in the source file:

```
COB74XLG SALARIES;INFO="$CONTROL &
BOUNDS,MAP,VERBS$SET&$X9=ON" &
COB74XLG ACCOUNTS;INFO="$DEFINE %A=5#"
```

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

xdbfilename Actual file designator for the file to be used by the symbolic debugger (XDB). This is a permanent file created by the compiler that contains the listing of the source files. The formal file designator is COBXDB.

If this file exists, then it must be in a special format created by a previous compile using this option. In this case, it is first purged. If the file is of the wrong type, the compile is not attempted. The user must either use a different name or purge the file.

Once the file is created, XDB expects the fully qualified name of the file to be unchanged. A FILE equation could be used if the file is renamed.

Operation Notes

The COB74XLG command compiles, links, and executes a program using the ANSI 1974 standard entry point. If you do not specify *textfile*, HP COBOL II/iX expects the source program to be entered from your standard input device. If you do not specify *listfile*, HP COBOL II/iX sends the output to your standard list device.

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. The Link Editor purges the object file and writes the linked program to $OLDPASS, which is then executed and may be executed repeatedly.

You cannot backreference the formal file designators used in this command (COBTEXT, COBOBJ, COBLIST, COBMAST, COBNEW, COBWKSP, and COBXDB) 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.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program but not in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile, link, and execute an HP COBOL II/iX program entered from your standard input device and send the program listing to your standard list device, enter:

```
COB74XLG
```

To compile, link, and execute an HP COBOL II/iX program from the disk file TEXTFL and send the program listing to the disk file LISTFL, enter:

```
COB74XLG TEXTFL,LISTFL
```

Related Information

Commands	COB74XL, COB74XLK, LINK, RUN, XEQ, LINKEDIT Utility
Manuals	*HP COBOL II/XL Reference Manual*
	HP COBOL II/XL Programmer's Guide
	HP Link Editor/iX Reference Manual

COB74XLK

Compiles and links an HP COBOL II/iX program using the 1974 ANSI standard entry point. HP COBOL II/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP COBOL II/iX is installed on your system. (Native Mode)

Syntax

```
COB74XLk[textfile]
[ ,[ progfile] [ ,[ listfile] [ ,[ masterfile] [ ,newfile] ] ] ]
[ ;INFO=quotedstring] [ ;WKSP=workspacename]
[ ;XDB=xdbfilename]
```

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile The name of the file that contains the source code that is to be compiled. This can be any ASCII or toolset access method (TSAM) file. The formal file designator is `COBTEXT`.

If you are running HP COBOL II/iX from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile*, the default file is `$STDIN`. `$STDIN` is the current input device, usually your terminal.

progfile The name of the object file to which the Link Editor writes the linked program. If you do not specify *progfile*, the default is `$NEWPASS`.

listfile The name of the file to which the compiler writes the program listing. This can be any ASCII file. The formal file designator is `COBLIST`. If you do not specify *listfile*, the default is `$STDLIST`. `$STDLIST` is usually the terminal in a session or the printer in a batch job.

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 `$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 file designator is `COBNEW`. Default is that no file is written.

quotedstring A string of no more than 255 characters, including the single or double quotation marks that enclose it, that specifies compile time options.

The *quotedstring* string may be used to pass dollar sign ($) commands to the compiler: `"$command1$command2$command3..."`. The `$` must be the first character in the string, and it must be used to separate multiple commands. To extend the *quotedstring* string over more than one physical line, make an ampersand (&) the last character of one line and continue the *quotedstring* string onto the next physical line.

Each $ command is limited in length to the same size as in the source file:

```
COB74XLK SALARIES,SALPRG;INFO="$CONTROL &
BOUNDS,MAP,VERBS$SET&$X9=ON" &
COB74XLK ACCOUNTS;INFO="$DEFINE %A=5#"
```

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

xdbfilename Actual file designator for the file to be used by XDB. This is a permanent file created by the compiler that contains the listing of the source files. The formal file designator is COBXDB.

If this file exists, then it must be in a special format created by a previous compile using this option. In this case it is first purged. If the file is of the wrong type, the compile is not attempted. The user must either use a different name or purge the file.

Once the file is created, XDB expects the fully qualified name of the file to be unchanged. A FILE equation could be used if the file is renamed.

Operation Notes

The COB74XLK command compiles and links an HP COBOL II/iX program into a disk file. If you do not specify *textfile*, HP COBOL II/iX expects your input from your standard input device. If you do not specify *listfile*, HP COBOL II/iX sends the listing output to your current list device.

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. The Link Editor overwrites *progfile* which can then be executed.

You cannot backreference the formal file designators used in this command (COBTEXT, COBLIST, COBMAST, COBNEW, COBWKSP, and COBXDB) 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.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program but not in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile and link an HP COBOL II/iX program entered from your standard input device with the listing printed on the standard list device, enter:

```
COB74XLK
```

To compile and link an HP COBOL II/iX source program input from the text file `SFILE` into a program file named `MYPROG`, with the resulting listing sent to the current list device, enter:

```
COB74XLK SFILE,MYPROG
```

Related Information

Commands	`COB74XL`, `COB74XLG`, `LINK`, `RUN`, `XEQ`, LINKEDIT Utility
Manuals	*HP COBOL II/XL Reference Manual*
	HP COBOL II/XL Programmer's Guide
	HP Link Editor/iX Reference Manual

COB85XL

Compiles an HP COBOL II/iX program using the 1985 ANSI standard entry point and creates an object file. HP COBOL II/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP COBOL II/iX is installed on your system. (Native Mode)

Syntax

`COB85XL`[*textfile*]

[,[*progfile*] [,[*listfile*] [,[*masterfile*] [,*newfile*]]]]

[;`INFO=`*quotedstring*] [;`WKSP=`*workspacename*]

[;`XDB=`*xdbfilename*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile The name of the file that contains the source code that is to be compiled. This can be any ASCII or toolset access method (TSAM) file. The formal file designator is `COBTEXT`.

If you are running HP COBOL II/iX from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile*, the default file is `$STDIN`. `$STDIN` is the current input device, usually your terminal.

objectfile Actual file designator of the object file, which is the output of the compiler. This file is stored in binary form and has a file code of either `NMOBJ` (`1461`) or `NMRL` (`1033`). Its formal file designator is `COBOBJ`. If the *objectfile* parameter is omitted, the object code is saved to the temporary file `$OLDPASS` if it exists, or to `$NEWPASS` which then becomes `$OLDPASS`.

If you specify *objectfile*, the compiler stores the object file in a permanent file of the correct size, type, and name you specified.

If either a file of the same name or the default file `$OLDPASS` already exists, the new object code overwrites the old if the file code is `NMOBJ` or is appended to the old if the file code is `NMRL`. If the file code is `NMRL`, any existing version of the code module is first purged.

Refer to the *HP COBOL/XL Programmer's Guide* (31500-90002) for information on the RLINIT and RLFILE commands that cause creation of an NMRL by default or initialization.

The compiler may issue an error message telling you that a new or existing object file is too small to contain the compiler's output or number of modules. In that case you must build a larger file or use the Link Editor to clean the NMRL. You may then recompile to the new file.

You may use the MPE/iX SAVE command to store $OLDPASS as a permanent file under another name.

listfile The name of the file to which the compiler writes the program listing. This can be any ASCII file. The formal file designator is COBLIST. If you do not specify *listfile*, the default is $STDLIST. $STDLIST is usually the terminal in a session or the printer in a batch job.

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 quoted string of no more than 255 characters, including the single or double quotation marks that enclose it, that specifies compile time options.

The *quotedstring* string may be used to pass dollar sign ($) commands to the compiler: "$command1$command2$command3...". The $ must be the first character in the string, and it must be used to separate multiple commands. To extend the *quotedstring* string over more than one physical line make an ampersand (&) the last character of one line and continue the *quotedstring* string onto the next physical line. Each $ command is limited in length to the same size as in the source file:

```
COB85XL SALARIES,SALOBJ;INFO="$CONTROL &
BOUNDS,MAP,VERBS$SET &$X9=ON" &
COB85XL ACCOUNTS;INFO="$DEFINE %A=5#"
```

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

xdbfilename Actual file designator for the file to be used by the symbolic debugger (XDB). This is a permanent file created by the compiler that contains the listing of the source files. The formal file designator is COBXDB.

If this file exists, then it must be in a special format created by a previous compile using this option. In this case it is first purged. If the file is of the wrong type, the compile is not attempted. The user must either use a different name or purge the file.

Once the file is created, XDB expects the fully qualified name of the file to be unchanged. A FILE equation could be used if the file is renamed.

Operation Notes

The COB85XL command compiles an HP COBOL II/iX program into an object file on disk. If you do not specify *textfile*, HP COBOL II/iX expects the source text to be entered from your standard input device. If you do not specify *listfile*, HP COBOL II/iX sends the program listing to the current list device.

You cannot backreference the formal file designators used in this command (COBTEXT, COBOBJ, COBLIST, COBMAST, COBNEW, COBWKSP, and COBXDB) 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.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program but not in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile an HP COBOL II/iX program stored in the file SOURCE into an object file called OBJECT, and send the listing to the disk file LISTFL, enter:

```
COB85XL SOURCE,OBJECT,LISTFL
```

Program development in native mode uses the MPE/iX LINK command not the MPE V/E PREP command. This produces a significant change in the method of compiling code. For example, if you have created a program called MAIN and a subprogram called SUB, each contained in a separate file, you might append the code from SUB to SOMEUSL in MPE V/E, like this:

```
COBOLII MAIN, SOMEUSL
COBOLII SUB, SOMEUSL
PREP SOMEUSL, SOMEPROG
RUN SOMEPROG
```

When using NMOBJ, however, the COB85XL command (in MPE/iX native mode) does not append SUB. MPE/iX compiles the source files into separate object files and then uses the Link Editor to link the two object files into the program file, as in this example:

```
COB85XL MAIN, OBJMAIN
COB85XL SUB, OBJSUB
LINK FROM=OBJMAIN,OBJSUB;TP=SOMEPROG
RUN SOMEPROG
```

On the other hand, if an NMRL is used instead of an NMOBJ, the above can be simplified to the following:

```
BUILD RLFILE;DISC=10000;CODE=NMRL
COB85XL MAIN, RLFILE
COB85XL SUB, RLFILE
LINK RLFILE, SOMEPROG
RUN SOMEPROG
```

Related Information

Commands	COB85XLG, COB85XLK, LINK, RUN, XEQ, LINKEDIT Utility
Manuals	*HP COBOL II/XL Reference Manual*
	HP COBOL II/XL Programmer's Guide
	HP Link Editor/iX Reference Manual

COB85XLG

Compiles, links, and executes an HP COBOL II/iX program using the ANSI 1985 standard entry point. HP COBOL II/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP COBOL II/iX is installed on your system. (Native Mode)

Syntax

```
COB85XLG[textfile]
[ ,[ progfile] [ ,[ listfile] [ ,[ masterfile] [ ,newfile] ] ] ]
[ ;INFO=quotedstring] [ ;WKSP=workspacename]
[ ;XDB=xdbfilename]
```

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile The name of the file that contains the source file that is to be compiled. This can be any ASCII or toolset access method (TSAM) file. The formal file designator is `COBTEXT`.

If you are running HP COBOL II/iX from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile*, the default file is `$STDIN`. `$STDIN` is the current input device, usually your terminal.

listfile The name of the file to which the compiler writes the program listing. This can be any ASCII file. The formal file designator is `COBLIST`. If you do not specify *listfile*, the default is `$STDLIST`. `$STDLIST` is usually the terminal in a session or the printer in a batch job.

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 *masterfile*. This can be any ASCII output file. Formal file designator is `COBNEW`. Default is that no file is written.

quotedstring A string of no more than 255 characters (including the single or double quotation marks that enclose it).

The *quotedstring* string may be used to pass dollar sign ($) commands to the compiler: `"$command1$command2$command3..."`. The `$` must be the first character in the string, and it must be used to separate multiple commands. To extend the *quotedstring* string over more than one physical line, make an ampersand (&) the last character of one line and continue the *quotedstring* string onto the next physical line. Each $ command is limited in length to the same size as in the source file:

```
COB85XLG SALARIES;INFO="$CONTROL &
BOUNDS,MAP,VERBS$SET&$X9=ON"
COB85XLG ACCOUNTS;INFO="$DEFINE %A=5#"
```

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

xdbfilename Actual file designator for the file to be used by the symbolic debugger (XDB). This is a permanent file created by the compiler that contains the listing of the source files. The formal file designator is COBXDB.

If this file exists, then it must be in a special format created by a previous compile using this option. In this case, it is first purged. If the file is of the wrong type, the compile is not attempted. The user must either use a different name or purge the file.

Once the file is created, XDB expects the fully qualified name of the file to be unchanged. A FILE equation could be used if the file is renamed.

Operation Notes

The COB85XLG command compiles, links, and executes a program using the ANSI 1985 standard entry point. If you do not specify *textfile*, HP COBOL II/iX expects the source program to be entered from your standard input device. If you do not specify *listfile*, HP COBOL II/iX sends the output to your standard list device.

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. The Link Editor purges the object file and writes the linked program to $OLDPASS, which is then executed and may be executed repeatedly.

You cannot backreference the formal file designators used in this command (COBTEXT, COBOBJ, COBLIST, COBMAST, COBNEW, COBWKSP, and COBXDB) 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.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program but not in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile, link, and execute an HP COBOL II/iX program entered from your standard input device and send the program listing to your standard list device, enter:

```
COB85XLG
```

To compile, link, and execute an HP COBOL II/iX program from the disk file `TEXTFL` and send the program listing to the disk file `LISTFL`, enter:

```
COB85XLG TEXTFL,LISTFL
```

Related Information

Commands	`COB85XL`, `COB85XLK`, `LINK`, `RUN`, `XEQ`, LINKEDIT Utility
Manuals	*HP COBOL II/XL Reference Manual*
	HP COBOL II/XL Programmer's Guide

COB85XLK

Compiles and links an HP COBOL II/iX program using the 1985 ANSI standard entry point. HP COBOL II/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP COBOL II/iX is installed on your system. (Native Mode)

Syntax

`COB85XLK[` *textfile*`]`

`[ ,[` *progfile*`] [ ,[` *listfile*`] [ ,[` *masterfile*`] [ ,`*newfile*`] ] ] ]`

`[ ;INFO=`*quotedstring*`] [ ;WKSP=`*workspacename*`] [ ;XDB=`*xdbfilename*`]`

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

textfile The name of the file that contains the source code that is to be compiled. This can be any ASCII or toolset access method (TSAM) file. The formal file designator is `COBTEXT`.

If you are running HP COBOL II/iX from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile*, the default file is `$STDIN`. `$STDIN` is the current input device, usually your terminal.

progfile The name of the object file to which the Link Editor writes the linked program. If you do not specify *progfile*, the default is `$NEWPASS`.

listfile The name of the file to which the compiler writes the program listing. This can be any ASCII file. The formal file designator is `COBLIST`. If you do not specify *listfile*, the default is `$STDLIST`. `$STDLIST` is usually the terminal in a session or the printer in a batch job.

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 `$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 file designator is `COBNEW`. Default is that no file is written.

quotedstring A quoted string of no more than 255 characters (including the single or double quotation marks that enclose it).

The *quotedstring* string may be used to pass dollar sign ($) commands to the compiler: `"$command1$command2$command3..."`. The `$` must be the first character in the string, and it must be used to separate multiple commands. To extend the *quotedstring* string over more than one physical line, make an ampersand (&) the last character of one line and continue the *quotedstring* string onto the next physical line.

Each $ command is limited in length to the same size as in the source file:

```
COB85XLK SALARIES,SALPRG;INFO="$CONTROL &
BOUNDS,MAP,VERBS$SET&$X9=ON"
COB85XLK ACCOUNTS;INFO="$DEFINE %A=5#"
```

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

xdbfilename Actual file designator for the file to be used by the symbolic debugger (XDB). This is a permanent file created by the compiler that contains the listing of the source files. The formal file designator is COBXDB.

If this file exists, then it must be in a special format created by a previous compile using this option. In this case, it is first purged. If the file is of the wrong type, the compile is not attempted. The user must either use a different name or purge the file.

Once the file is created, XDB expects the fully qualified name of the file to be unchanged. A FILE equation could be used if the file is renamed.

Operation Notes

The COB85XLK command compiles and links an HP COBOL II/iX program into a disk file. If you do not specify *textfile*, HP COBOL II/iX expects your input from your standard input device. If you do not specify *listfile*, HP COBOL II/iX sends the listing output to your current list device.

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. The Link Editor overwrites *progfile* which can then be executed.

You cannot backreference the formal file designators used in this command (COBTEXT, COBOBJ, COBLIST, COBMAST, COBNEW, COBWKSP, and COBXDB) 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.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program but not in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile and link an HP COBOL II/iX program entered from your standard input device, with the listing printed on the standard list device, enter:

```
COB85XLK
```

To compile and link an HP COBOL II/iX source program input from the text file SFILE into a program file named MYPROG, with the listing sent to the current list device, enter:

```
COB85XLK SFILE,MYPROG
```

Related Information

Commands	COB85XL, COB85XLG, LINK, RUN, XEQ, LINKEDIT Utility
Manuals	*HP COBOL II/XL Reference Manual*
	HP COBOL II/XL Programmer's Guide

COBOLII

Compiles a compatibility mode COBOLII program on the COBOL 74 compiler. COBOLII is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. The native mode equivalent of this command is `COB74XL`.

For information on the 85 entry point, refer to the *HP COBOL II/XL Reference Manual* (31500-90001)

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 this 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/iX `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 occurrence 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 appear before the first line of source code in the text file.

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

Operation Notes

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

You cannot backreference the formal file designators used in this command (COBTEXT, COBLIST, COBMAST, COBNEW, COBWKSP, and COBXDB) 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, job, or program but not in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

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 disk file LISTFL, enter:

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

Related Information

Commands	`COBOLIIGO`, `COBOLIIPREP`, `LINK`, `RUN`, `XEQ`, LINKEDIT Utility
Manuals	*HP COBOL II/XL Reference Manual*

COBOLIIGO

Compiles, prepares, and executes a compatibility mode COBOLII program on the COBOL 74 compiler. COBOLII is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. The native mode equivalent of this command is `COB74XLG`.

For information on the 85 entry point, refer to the *HP COBOL II/XL Reference Manual*

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 occurrence of two single or two double quotation marks 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 appear before the first line of source code in the text file.

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

Operation Notes

The `COBOLIIGO` command compiles, prepares, and executes a compatibility mode 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, `$OLDPASS`, which is passed directly to the MPE segmenter. The segmenter purges the USL file and writes the prepared program to `$OLDPASS`, which is then executed and may be executed repeatedly.

You cannot backreference the formal file designators used in this command (`COBTEXT`, `COBLIST`, `COBMAST`, `COBNEW`, `COBWKSP`, and `COBXDB`) 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, job, or program. It is not available in BREAK. Pressing **Break** suspends the execution of this command. Entering the `RESUME` command continues the execution.

Examples

To compile, prepare, and execute a compatibility mode 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 disk file `TEXTFL` and send the program listing to the disk file `LISTFL`, enter:

```
COBOLIIGO TEXTFL,LISTFL
```

Related Information

Commands	`COBOLII`, `COBOLIIPREP`, `LINK`, `RUN`, `XEQ`, LINKEDIT Utility
Manuals	*HP COBOL II/XL Reference Manual*

COBOLIIPREP

Compiles and prepares a compatibility mode COBOLII program on the COBOL 74 compiler. COBOLII is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. The native mode equivalent of this command is COB74XLK.

For information on the 85 entry point, refer to the *COBOL/II 3000 Reference Manual*

Syntax

```
COBOLIIPREP[ textfile]
[ ,] progfile] ,[ listfile] [ ,[ masterfile] [ ,newfile] ] ]
[ ;INFOR=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.

progfile Actual file designator of the program file to which the prepared program segments are written. If *progfile* is omitted, the MPE segmenter creates the program file, which resides in the temporary file domain as $OLDPASS. If entered, *progfile* indicates that the file was created in one of two ways:

- By 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 $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 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 occurrence of two single or double quotation marks 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 appear before the first line of source code in the text file.

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.

Operation Notes

The COBOLIIPREP command compiles and prepares a compatibility mode COBOLII program into a program file on disk. 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 MPE segmenter. The segmenter overwrites the USL file and writes the prepared program to $OLDPASS, if *progfile* is omitted, which can then be executed.

You cannot backreference the formal file designators used in this command (COBTEXT, COBLIST, COBMAST, COBNEW, and COBWKSP) 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, job, or program. It is not available in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

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

```
COBOLIIPREP
```

To compile and prepare a COBOLII source program input from the text file SFILE into a program file named MYPROG, with the listing sent to the current list device, enter:

```
COBOLIIPREP SFILE,MYPROG
```

Related Information

Commands	COBOLII, COBOLIIGO
Manuals	*HP COBOL II Reference Manual* (31500-90001)

COMMENT

Inserts a comment into a command stream or user command. (Native Mode)

Syntax

COMMENT [*text*] or

# [*text*]

Parameters

text	Information composed of 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.

Operation Notes

The COMMENT command allows you to include an explanation about the purpose of commands or the logic used in creating the job. It also is used to create job headings. After the COMMENT command is entered, it can be followed by a message made up of any ASCII characters. If # format of a comment is used the # must be the 1st non-blank character in the command line

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

The following is an example of a job heading using a comment:

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

Related Information

Commands	JOB, UDCs, command files
Manuals	None

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.

Operation Notes

The CONSOLE command is used to display the logical device number of the terminal currently being used as the system console, or to move the console to another logical device. Listing the current location of the console requires no special capabilities. Moving the console requires system manager (SM) capability.

The console cannot be moved to a terminal using a multipoint terminal software (MTS) line, or a packet assembly and disassembly (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 new console displaying 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.

When you enter the CONSOLE command without parameters, it reports the current logical device number (LDEV) of the console. You may also find out the LDEV of the current console by interrogating the HPCONSOLE variable. To do so, enter the command SHOWVAR HPCONSOLE at the colon prompt. Note, however, that Control and maintenance processor (CMP) and diagnostic control unit (DCU) prompts and messages remain with the configured terminal, for example, Channel 1, Device 0. This feature cannot be moved to another terminal.

NOTE Before transferring the system console to another terminal, be sure that you can take the console back when you need it by allowing yourself the =CONSOLE command. (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.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be used by any user to determine the location of the console. To change the location of the console, this command must be issued from the console itself, unless distributed to users with the `ALLOW` command, or the user must have system manager (SM) capability.

Examples

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/iX as logical device 31, enter:

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

CONTINUE

Overrides a job error so that the job or user command (command file or UDC) continues executing. **(Native Mode)**

Syntax

CONTINUE

Parameters

None.

Operation Notes

The CONTINUE command permits a job or session to continue even though the command immediately following the CONTINUE command results in an error (with an 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 or user command to abort. If an error occurs, the job or user command continues to run, and the error message is reported. The variable CIERROR contains the error number.

The CONTINUE command protects only the next command. However, if the next command is a user command (command file or UDC) and an error occurs anywhere within it, execution resumes at the command following the user command. In effect, the CONTINUE command treats a user command as a simple, indivisible command.

You may use the HPAUTOCONT variable to produce a global "continue." Refer to appendix A, "Predefined Variables in MPE/iX."

Use

This command may be issued in a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

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 USER.PUBS
!CONTINUE
!RUN MYPROG
!IF JCW <= WARN THEN
!  RUN MYPROG2
!ENDIF
!EOJ
```

Related Information

Commands	`JOB`
Manuals	Appendix A, "Predefined Variables in MPE/iX"

COPY

Copies one file to another by creating a new file or by overwriting an existing file.

The `COPY` command can be used to copy files to and from HFS directories. You cannot use `COPY` to copy directories to or from other directories. Users with SM capabilities are able to copy files to MPE accounts outside of their current logon account.

Syntax

COPY[FROM=] *sourcefile* [{ ;TO= | , | blank } *targetfile*] [ASK | YES | NO]

Parameters

sourcefile The name of the file that is to be copied.

A file with HFS syntax must begin with a dot (.), or a slash (/).

You may not specify system-wide ($ prefix), CM KSAM, or privileged files as *sourcefile* or *targetfile*.

targetfile The name of the file to which *sourcefile* is to be copied. If *targetfile* is omitted, the source file is copied to *sourcefile* in the user's current working directory (CWD). You may qualify *targetfile* with both file and group name, or specify only the destination *group* or specify only the destination directory. To specifiy a group name as the target use *.groupname*. If only *group* is specified, `COPY` copies the source file to a file named *sourcefile targetfile*. Likewise if only a directory is specified, COPY copies the source file to a file named *targetfile/sourcefile*.

NOTE Since *.groupname* can be specified as the *targetfile*, and HFS file names can also start with a dot (.), this could lead to confusion as to whether an MPE group or HFS file name is desired for the *targetfile*. If the *targetfile* is an HFS filename starting with a dot (.), then the *targetfile* must be preceded with a dot and slash (./). For example, to represent a *targetfile* `.FOO` in an HFS current working directory, the file must be represented as `./.FOO`.

NOTE If the target file is a directory name it may end in a slash (/) to improve readability of copy in scripts.

NOTE The *max extent* value for *targetfile* value may not be the same as for *sourcefile*.

ASK If *targetfile* already exists, `COPY` prompts the user to choose an action with the following prompt:

`PURGE OLD` *targetfile*?

Valid replies to this prompt are:

Y or YES Instructs `COPY` to purge the original *targetfile* and create a new *targetfile*.

N or NO Instructs COPY to terminate.

ASK is the default, except in a job or in other cases when the user is not using interactive mode. In such cases, ASK has no meaning, and YES becomes the default.

YES Instructs COPY to purge *targetfile* if it already exists. No message is displayed for the user, as would be the case with ASK. YES is the default in jobs, or at other times when the user is not using an interactive mode.

NO Instructs COPY to terminate if *targetfile* already exists.

Operation Notes

This command performs a fast copy of *sourcefile* to *targetfile* and leaves *sourcefile* unchanged. Both files must be nonspooled disk files residing on the host system. You may specify files that are backreferenced with a file equation (*). However, this command only supports three file equation options: the file name, the final disposition (;TEMP or ;SAVE), and the disk volume or volume class (;DEV= DISC or ;DEV=<DISC LDEV NUMBER>). All other file equation options are ignored.

The file disposition of *targetfile* defaults to that of *sourcefile*. For example, if *sourcefile* is TEMP, *targetfile* is created TEMP. If *sourcefile* is PERM, *targetfile* is created as PERM. This file disposition can be overridden by using a file equation since this is one of the three options supported for file equations.

All file access attributes of the source file, including ACDs (access control definitions) are duplicated for the target file.

If a source file has an ACD, the ACD is copied to the target file. If a file does not have an ACD, and it is copied outside an MPE group, it is automatically assigned an ACD.

Use

This command may be invoked from a session, a job, a program, or in BREAK. Pressing **Break** aborts the execution of this command and purges the *targetfile*.

The COPY command can be invoked in BREAK and does not suffer from process creation overhead.

Examples

To copy ABCD.*logongroup* to EFG.*logongroup*, enter:

```
COPY ABCD, EFG
```

To copy ABCD.*logongroup* to ABCD.*newgroup*, enter:

```
COPY ABCD, .newgroup
```

To copy ABCD.*grp* to ABCD.*logongroup*, enter:

```
COPY ABCD.grp
```

In the next example the file MYFILE.PUB.SYS is copied to MyFile under the current working directory (CWD). Note that the target file name has to have the dot and slash (./) prefix.

```
COPY myfile.pub.sys, ./MyFile
```

In the next example, the file File1 under the CWD is copied to MYFILE.PUB in the current account.

```
COPY ./File1, myfile.pub
```

In this next example, file1 in directory dir0 is copied to file2 in directory dir1.

```
COPY ./dir0/file1, ./dir1/file2
```

In the following example, the file TEST has a lockword which is the word LOCK. The file is copied into file1 in the dir0 directory.

```
COPY TEST/LOCK, ./dir0/file1
```

The next example copies the file FILE1 to the directory dir1

```
COPY FILE1 ./dir1/
```

The next example copies the file file1 to the directory DIR1.

```
COPY ./file1 DIR1
```

Related Information

Commands	FCOPY
Manuals	*Using the HP 3000 Series 900: Fundamental Skills*

DATA

Enters data into the system from a device file. (Cannot be used to enter data from `$STDIN`.) (Native Mode)

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. It may contain up to eight alphanumeric characters, beginning with a letter.

username User name that allows you to access MPE/iX in this account, as established by the account manager. It may contain up to eight alphanumeric characters, beginning with a letter.

userpass User password, optionally assigned by the account manager. It may contain up to eight alphanumeric characters, beginning with a letter. If a password exists, but is not supplied in the command syntax, the `STREAM` command will prompt you for it if:

- The `STREAM` command is invoked from a session.
- Neither $STDIN nor $STDLIST is redirected.
- The `DATA` command is a first level data command (it is not nested within a second level `STREAM` command).

acctname Account name under which job/session is running, as established by the system manager. It may contain up to eight alphanumeric characters, beginning with a letter.

acctpass Account password, optionally assigned by system manager. It may contain up to eight alphanumeric characters, beginning with a letter. If a password exists, but is not supplied in the command syntax, the `STREAM` command will prompt you for it if:

- The `STREAM` command is invoked from a session.
- Neither $STDIN nor $STDLIST is redirected.
- The `DATA` command is a first level data command (it is not nested within a second level `STREAM` command).

filename Optional name for the data, used to distinguish between two separate data files that are to be read by the same program. It may contain up to eight alphanumeric characters, beginning with a letter. Default is that no distinguishing name is assigned.

Operation Notes

This command identifies data to be read from a device file other than your standard job/session input device. It can be used, for example, to input a data file from a spooled input device for later use by an interactive session or a batch job. The DATA command is the only command that can be entered before a job or session is initiated. Files identified by DATA may be input only from magnetic tape on spooled tape drives or with the STREAM command.

To designate a set of data as an auxiliary file for your job or session, enter the DATA command followed by the set of data and the EOD command. To access the data, begin your job or session using the same identity ([*jsname*,]*username.acctname*) used in the DATA command. If the *filename* parameter is omitted, several data files can be read from any job or session with the same identity.

When entered from magnetic tape, such data must reside in a file on a single tape volume, and the blocking factor must be 1. When the media containing the data file is placed on the tape drive and that device is placed online, MPE/iX reads the entire file. At that point, the job can access the data, which remains available until it is actually read. To submit data from a disk file, you must use the STREAM command.

The time-related parameters of the STREAM command may *not* be used when STREAM is used with the DATA command.

The STREAM command will prompt you for both user and account passwords if they exist and are not supplied in the DATA command if the following conditions are met:

- The STREAM command is invoked from a session.
- Neither $STDIN nor $STDLIST is redirected.
- The DATA command is a first level data command (it is not nested within a second level STREAM command).

Use

This command may be issued from a session or job. Use the STREAM command to input a data file. This command cannot be used directly from $STDIN or from a program.

Examples

A data file is created on disk, and the STREAM command is used to make the file available to your program.

To create the file DATAFL on disk, invoke a text editor (like EDITOR) and enter the data beginning with the DATA command and ending with the EOD command. For example:

```
EDITOR
/ADD
DATA SESSB,BROWN.ACCT1
 .
 .
 .
EOD
//
/KEEP DATAFL,UNN
/EXIT
```

To stream the data file using the STREAM command, enter:

```
STREAM DATAFL
```

To log on to a session, using precisely the same identity that was used in the DATA command, enter:

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

To enter a FILE command equating the formal file designator (used by the program) with the stream device (identified by the device class name JOBTAPE), enter:

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

To run the program that requires the data, enter:

```
RUN PROGY
```

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 with the STREAM command.

Related Information

Commands	EOD, STREAM
Manuals	None

DEALLOCATE

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

Syntax

DEALLOCATE[PROGRAM | PROCEDURE] *,name*

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.
name	The name of the program file or procedure to be deallocated.

Operation Notes

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

You may use a comma (,), a semicolon (;), and an equal sign (=) as delimiters.

NOTE NM and CM loader error messages are reported differently, allowing you to determine the system in which the error occurred.

NM Loader Error: *ErrMessage* (LDERR *nnnn*)

CM Loader Error: *ErrMessage* (LOAD ERR *nnnn*)

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. Any program for which a user has EXECUTE access can be deallocated. A user with system supervisor (OP) capability can deallocate any program.

Example

To deallocate a program file named PROGEX, enter:

```
DEALLOCATE PROGEX
```

DEALLOCATE does not give back memory; it gives back table entries.

Related Information

Commands	ALLOCATE

Manuals *Introduction to MPE XL for MPE V Programmers*

DEBUG

Instructs MPE/iX to enter the system debugger. (Native Mode)

Syntax

DEBUG[*commands*]

Parameters

commands A series of system debugger commands to be executed before the debugger prompt is displayed. The string may be as many as 255 characters long. There are no delimiters or keywords needed to pass these commands to the debugger. If the CONTINUE command is not part of the *commands* string, you are left in debug after the execution of those commands.

Operation Notes

The DEBUG command enters the system debugger. An optional parameter, *commands*, defines a string of system debugger commands that are executed when the debugger is invoked, but before the debugger prompt is displayed.

If the string contains commands that return the user to the command interpreter, those commands are executed. Any remaining commands are pushed onto a command stack. Another invocation of the DEBUG command executes the commands saved on the stack. If you invoke DEBUG X;Y;Z and the command X returns control to the CI, then DEBUG A;B;C executes the commands A;B;C;Y;Z.

Use

This command may be issued from a session, program, or in BREAK. It may not be issued from a job. Pressing **Break** has no effect on this command. Privileged mode (PM) capability is required to use this command.

Example

To produce a stack trace and return to the command interpreter:

```
DEBUG TRACE;C

DEBUG XL A.00.00

HPDEBUG Intrinsic at: a.006b4104 hxdebug+$130
   PC=a.006b4104 hxdebug+$130
* 0) SP=40221c58 RP=a.006b8e7c exec_cmd+$73c
  1) SP=40221ac8 RP=a.006ba41c try_exec_cmd+$ac
  2) SP=40221a78 RP=a.006b8638 command_interpret+$274
  3) SP=40221620 RP=a.006bae5c xeqcommand+$1d0
  4) SP=40221210 RP=a.006b7604 ?xeqcommand+$8
    export stub: 7d.000068dc main_ci+$94
  5) SP=40221178 RP=7d.00007420 PROGRAM+$250
```

```
   6) SP=40221130 RP=7d.00000000
    (end of NM stack)
 :
```

Related Information

Commands	RESETDUMP, RUN, SETDUMP
Manuals	*System Debug Reference Manual*

DELETESPOOLFILE

Deletes a spoolfile from disk.

Syntax

DELETESPOOLFILE{ #O*nnn* #I*nnn* *ldev* }

Parameters

#O*nnn* The identification of a READY or ACTIVE output spoolfile.

#I*nnn* The identification of a READY, input spooled data file.

ldev The logical device number on which the spoolfile is ACTIVE.

Operation Notes

Before deleting an ACTIVE spoolfile, 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/iX returns the colon prompt (`:`), you know that the `DELETESPOOLFILE` command instruction has been sent to the spooler process. It is not executed, however, until the output device is put back online.

You may not use the `DELETESPOOLFILE` command on the following type of files:

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

The `DELETESPOOLFILE` command deletes ACTIVE data input files that are submitted on a spooled device. It cannot delete such files while they are being streamed.

Use

This command may be issued from a session, program, or in BREAK. It may not be issued from a job. Pressing **Break** has no effect on this command. This command may be used only from the console unless distributed to users with the `ALLOW` or `ASSOCIATE` command.

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 is 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 skips to the top of the next physical page. If the device is a page printer, the default environment is reloaded.

Related Information

Commands	`ALTSPOOLFILE`
Manuals	*Native Mode Spooler Reference Manual*

DELETEVAR

Deletes one or more MPE/iX variables. (Native Mode)

Syntax

DELETEVAR *varname* [,*varname*] ... [,*varname*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

varname The name of the variable to be deleted.

Operation Notes

Deletes a specific MPE/iX variable, or all variables specified by a pattern. If you specify more than one *varname*, you must separate them with commas.

You may use the wildcard characters, @, #, ?, and [ ] to specify a set or range of values.

@ Specifies zero or more alphanumeric characters, or the underbar character (_). Used by itself, it specifies all possible combinations of such characters. Used with other characters it indicates all the possible names that include the specified characters. @ABC@ specifies all names that include ABC anywhere in the name.

# Specifies one numeric character. A###@ specifies all names that begin with A followed by any three digits, followed by any combination of 0 to 251 alphanumeric (or underbar) characters.

? Specifies one alphanumeric character. A?# specifies all three-character names that begin with A, followed by an alphanumeric, followed by a digit.

[ ] Specifies a set or range of characters. The set may appear anywhere in the name. This range specification is not case sensitive and, therefore, [A-K] is the same as [a-k]. If you specify a null set such as [k-a], MPE/iX reports an error.

@[abc]@# = All names containing A, B, or C and ending in a single digit.

[a-k]@ = All names that begin with any one of the letters A through K.

[n-a] = Is not valid and is flagged as an error.

Use

This command is available in a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Examples

To delete two specific variables, enter:

```
DELETEVAR firstvariable, secondvariable
```

To delete all variables beginning with a single alphabetic character and ending with the characters axval, enter:

```
DELETEVAR ?axval
```

To delete all variables created by the user, enter:

```
DELETEVAR @
```

To delete a range of variables, for example, those that begin with the letters P, Q, R, S, or T followed by zero or more characters that end with the string module. In the following example variables such as PMODULE, QMODULE, RMODULE, SMODULE, TMODULE, and TIME_MODULE are all deleted by entering:

```
DELETEVAR [P-T]@MODULE
```

MPE/iX predefined variables, which are listed in appendix A, cannot be deleted.

To delete all variables beginning in T and ending in two digits such as TMP11, T25, TMP_237 but not T2, enter:

```
DELETEVAR T@##
```

Related Information

Commands	SETJCW, SETVAR, SHOWJCW, SHOWVAR
Manuals	*Using the HP 3000 Series 900: Advanced Skills*
	Appendix A, "Predefined Variables in MPE/iX"

DISALLOW

Prohibits access to a specific operator command.

Syntax

```
DISALLOW FILE=formaldesignator[ ;SHOW]
DISALLOW] [ @.@ user.@ @.user user.acct ] ;COMMANDS=command [ ,command,...]
```

Parameters

formal- designator An ASCII file name, which may consist of one to eight alphanumeric characters, beginning with an alphabetic character. It may be fully or partially qualified and may be back-referenced in a file equation.

`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.

command The names of those commands to which the user is prohibited access.

Operation Notes

The operator uses the `DISALLOW` command to prohibit a user from executing specific operator commands previously allowed with the `ALLOW` command. You can use the command in any of three ways:

- Direct mode, in which you enter specific user names and account and the list of prohibited commands directly at the console.
- Indirect mode, in which you use a text editor such as EDIT/3000 to create a file that contains the user name and account of those users who will be prohibited from executing certain operator commands, and a list of disallowed commands.
- Subsystem mode, in which you enter the `DISALLOW` command, press **Return**, and, at the ">" prompt, enter the user and account names and the list of prohibited commands.

See the "Examples" section for more information.

You may enter as many prohibited commands as you want, in any of the three modes. However, in direct mode and subsystem mode, `DISALLOW` acts to prohibit the first nineteen commands and ignores any additional commands you may have specified. To disallow more than nineteen commands, create a file that contains the necessary information and specify it on the command line (i.e. "Indirect mode").

NOTE Do NOT confuse *operator* commands with *console* commands. For a description of the difference between console and operator commands refer to the ALLOW command. The commands which may be disallowed are the same as the commands which may be allowed. Refer to the ALLOW command for a list of commands which may be allowed.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** will terminate subsystem mode and produce an error message but has no effect on commands already entered in subsystem mode. This command may be used only from the console unless distributed to users with the ALLOW command.

Examples

To prohibit the user USER.TECH from executing the REPLY and ABORTIO commands, enter the following at the system console:

```
  DISALLOW USER.TECH;COMMANDS=REPLY,ABORTIO
```

To use subsystem mode to prohibit the user MGR.MANUALS from executing the BREAKJOB command, enter the following at the system console:

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

To use indirect mode, you create a file with all of the necessary information, and then invoke the changes by specifying the file using the FILE= parameter of the DISALLOW command.

```
 EDITOR
 HP32201A.07.17 EDIT 3000 TUE, MAY 29, 1987, 5:08 PM
 (C) HEWLETT-PACKARD CO. 1985
 /ADD
   1  SUSAN.PAYROLL;COMMANDS=ALTJOB,ALTSPOOLFILE
   2  JOHN.ACCTNG;COMMANDS=ALTSPOOLFILE,DELETESPOOLFILE
   3  //
   ...
 /KEEP COMNDTMP
 /E

 DISALLOW FILE=COMNDTMP;SHOW
```

If you want MPE/iX to display each command line as it is executed from the file, inclue the SHOW parameter.

You may backreference the file with a file equation as follows:

```
 FILE BACKF=COMNDTMP
 DISALLOW FILE=*BACKF;SHOW
```

If the file has a lockword it may be inserted as follows:

```
DISALLOW FILE=COMNDTMP/LOCKWORD;SHOW
```

Related Information

Commands	`ALLOW`, `SHOWALLOW`
Manuals	*Performing System Operation Tasks*

DISASSOCIATE

Removes control of a device class from the user.

Syntax

DISASSOCIATE ***devclass***

Parameters

devclass	The name of a device class configured during SYSGEN.

Operation Notes

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.

Use

This command may be issued from a session, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

To terminate control of the device class TAPE, enter:

```
DISASSOCIATE TAPE
```

Related Information

Commands	ASSOCIATE

DISCRPS

Enables or disables the rotational position sensing (RPS) feature on a specified logical device. It requires a special firmware upgrade CS-80 disk drives.

Syntax

DISCRPS *ldev* {   ,ENABLE [{ ,*value*,*value*}]   ,DISABLE }

Parameters

ldev The logical device number of the specified CS-80 disk drive.

ENABLE Enables rotational position sensing on the device.

DISABLE Disables rotational position sensing on the device.

value Allows the time-to-target and window size to be tuned, in hundreds of micro seconds. If you specify one value you must specify both values. The first is interpreted as the time-to-target value; the second is interpreted as the window size value. This parameter only works in conjunction with ENABLE.

```
                  (micro seconds)
 Default time-to-target 90 (9000     )
         window size   30 (3000     )
```

ONLY use this parameter if you have a clear understanding of its meaning and implications.

Operation Notes

The DISCRPS command allows you to enable or disable the rotational position sensing feature for CS-80 disk drives. With RPS enabled, the disk 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.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be executed only from the console unless distributed to users with the ALLOW or ASSOCIATE commands.

Example

To enable the RPS feature on logical device 1 and display the status of the disk drive, enter:

```
DISCRPS 1,ENABLE
SHOWDEV 1
LDEV   AVAIL    OWNERSHIP  VOID  DEN ASSOCIATION
1    DISC (RPS) 50 FILES
```

To use the *value* parameter with ENABLE to set time-to-target and window size to the default values, enter:

```
DISCRPS ldev,ENABLE,90,30
```

Related Information

Commands	SHOWDEV
Manuals	*CS/80 Instruction Set Programmers Manual*

DISCUSE (UDC)

The DISCUSE UDC executes the DISKUSE command to display disk space usage, in sectors, for one or more directories or a directory tree. This UDC is provided for those who are used to spelling disk with a "c".

System-defined UDCs are not automatically available. Your System Manager must use the SETCATALOG command to make these UDCs available for your use. For example:

```
SETCATALOG HPPXUDC.PUB.SYS;SYSTEM
```

Syntax

DISCUSE[[DIR=] *dir_name*] [;USENAME | ;TREE | ;NOTREE]

Parameters

Refer to the DISKUSE command for a complete explanation of the parameters used with the DISCUSE UDC. The following parameters are supported with the DISCUSE UDC.

dir_name Directory name for which information is being listed (optional).

TREE Causes all directories below and including *dir_name* to be reported.

NOTREE Causes *dir_name* only to be reported.

USENAME Causes DISKUSE to use *dir_name* name to decide whether or not to display multiple levels of directories.

Operation Notes

The DISCUSE UDC runs the DISKUSE command and reports disk space, in sectors, for a directory. Refer to the DISKUSE command for a complete explanation of the operation.

Use

This UDC may be issued from a session, a job, a program, or in break mode. Pressing **Break** aborts execution.

Examples

The following example illustrates the use of the DISCUSE UDC. Note that a message is printed to remind you to use the DISKUSE command.

```
DISCUSE
Please use the DISKUSE command.
                  ^

      SECTORS
    TREE   LEVEL DIRECTORY (CWD= /ACCT/GROUP/d0)
           BELOW

    2100    330 .
```

Refer to the DISKUSE command later in this chapter for additional examples.

Related Information

Commands DISKUSE, LISTFILE, REPORT

DISKUSE

Displays disk space usage, in sectors, for one or more directories or a directory tree.

Syntax

DISKUSE[[DIR=] *dir_name*] [; TREE | NOTREE | USENAME]

Parameters

dir_name Directory name for which information is being listed (optional). The *dir_name* is assumed to be an MPE syntax name. HFS-named directories may be shown if *dir_name* starts with a dot (.) or a slash (/). If *dir_name* is an HFS name and ends in a slash, then all objects at all levels under and including *dir_name* are reported, unless the NOTREE option is specified. The use of wildcards is permitted. If *dir_name* is omitted, the process' current working directory (CWD) is assumed.

TREE Causes all directories below and including *dir_name* to be reported. The *dir_name* may or may not end in a slash (/), with no error or warning detected. Since the MPE naming convention does not support a trailing slash (/), the TREE option is the only way to report multi-level disk space usage for an MPE-named directory in a single command.

NOTREE Causes *dir_name* only to be reported. If *dir_name* is an HFS name and ends in a slash (/), a warning tells you that NOTREE overrides the trailing slash (/).

USENAME Causes DISKUSE to use *dir_name* name to decide whether or not to display multiple levels of directories. If *dir_name* is an HFS name and ends in a slash (/), then it and all directories under it are shown. If *dir_name* does not end in a slash (/), then only *dir_name* is reported. The USENAME parameter only applies to HFS-named directories and is ignored for MPE-named directories. The USENAME parameter is the default.

Operation

The DISKUSE command reports disk space, in sectors, for a directory. Disk space allocated to directories themselves (including accounts and MPE groups) is counted as part of the total number of sectors. The process' CWD is shown for all relative pathnames.

The number of components in the pathname controls the level of directories being reported. If a pathname has four components, for example, /a/b/c/d, then only directories with four or more components contribute to the output. This also applies to the use of wildcard component names. For example, /@/@/@/@ only counts directories with at least four components in their pathname (absolute or relative, depending on how it was specified). MPE names follow the same formula: @.@.@ reports only MPE-named directories one level below MPE groups. (@.@ is the same since it is qualified with the logon account name.)

Use

You must have traverse directory entries (TD) and read directory entries (RD) permissions to each directory contributing to the reported totals. TD access is needed to each directory component named in *dir_name*. (Refer to the `ALTSEC` command in this chapter for additional information on directory permissions.)

Note that the MPE syntax cannot specify a *group.account*. MPE syntax only permits *dir.group.acct* if *dir* is a valid MPE name; that is, all uppercase alphanumeric. (If *group.account* were specified, it would be interpreted as a file called *group.account.logon_account*.)

Directory errors can occur while `DISKUSE` is collecting file space information. For example, if you lack traverse directory entries (TD) access to one or more of the lower level directories, an error occurs.

If `;TREE` is specified, you will only be able to see directories that you have TD and RD access to. `DISKUSE` stops on the first error encountered. This may result in no data (other than a header) displayed, or in the case of wildcard names, some directories are seen (up to the directory where the error occurred). Even in the wildcard directory name case, once an error is encountered, `DISKUSE` terminates.

There are several ways to see all disk space used on the system:

- To show the disk space for every directory on the system, enter:

```
DISKUSE /
```

- To show only the total system disk space in one line, enter:

```
DISKUSE /;NOTREE
NOTREE option overrides directory name ending in "/". (CIWARN 9041)
```

- To display disk space used by all directories directly under the root, enter:

```
DISKUSE /@
```

Examples

The illustration below shows a hierarchical directory structure, upon which all of the succeeding examples are based. Directory names are shown as the character d plus a number (for example, d0), and file names are shown as the character f plus a number (for example f1). For illustrative purposes, the HPPROMPT variable has been set to show the current working directory (HPCWD). For example:

```
:setvar hpprompt "hpcwd:"
/ACCT/GROUP/d0:
```

Hierarchical Directory Structure

```
                 /ACCT/GROUP/d0 = CWD
                    |
   -----------------|------------------------------
   |        |                  |            |  |  |
   d1       d2                 d3           f1 f2 f3
            |                  |
   ---------|--------------    |---------------------
    |   |  |   |      |   |       |   |  |  |  |
    d4  f4 f5  d5    d6  f6      d7  f7 f8 f9 f10
    |          |      |           |
  --|----     -|-  ---|------   --|--------------------
   |    |      |   |  |     |   |    |   |   |    |   |
  f11 f12    d8  f13 f14 f15  d9  f16 f17 f18 f19 f20
```

The example shown below illustrates the format of the DISKUSE output. In this example, the TREE option is implied by the trailing slash (/). The current working directory (CWD) relative display is shown as part of the header line. If the CWD name is long, it truncates with a dollar sign ($).

```
/ACCT/GROUP/d0:diskuse ./
     SECTORS
   TREE    LEVEL  DIRECTORY (CWD= /ACCT/GROUP/d0)
           BELOW
    64  +     0  ./d1/
    96       32  ./d2/d4/
    64        0  ./d2/d5/d8/
    128      64  ./d2/d5/
    112      48  ./d2/d6/
    448 +   240  ./d2/
    64        0  ./d3/d7/d9/
    208     144  ./d3/d7/
    336 +   128  ./d3/
    48  +     0   (files directly below specified directory)
    960     240  ./ (64 +)
/ACCT/GROUP/d0:
```

Each of the columns contains information about the directory.

DIRECTORY (left-justified) Displays the selected directory name, in HFS-format. The directory pathname wraps around to the next line if it is longer than the field.

LEVEL BELOW (right-justified) Shows the number of sectors allocated directly to all objects immediately under the named directory. The space used by the listed directory file (container) does not contribute to this number, nor does the space used by the objects under directories under the displayed directory. The sum of the number of sectors reported by the following command equals the number shown under the LEVEL BELOW column. The number in the LEVEL BELOW column is zero if the reported directory is empty.

LISTFILE *dir_name*/@,2;NOTREE

TREE (right-justified) Displays the total number of sectors used by the directory listed. This includes space used by the directory itself, all files immediately under the directory, and space used by all subdirectory entries. The sum of the number of sectors seen in the following command equals the total number in the TREE column.

LISTFILE *dir_name*,2;TREE

The plus signs (+) shown in the TREE column refer to the directories that are one level below the target directory. When added, the sectors shown in this example equal 896. The last entry shows the total number of sectors (960) used by all subdirectories under the target directory (896) plus the sectors used by the target directory itself (64).

The next example illustrates the use of the NOTREE option. Only the directory name is displayed.

```
/ACCT/GROUP/d0:diskuse /ACCT/GROUP/d0 ;notree
     SECTORS
   TREE    LEVEL  DIRECTORY
           BELOW
    960     240  /ACCT/GROUP/d0/
/ACCT/GROUP/d0:
```

If the directory name parameter is omitted, the CWD is assumed, as seen in the following example:

```
/ACCT/GROUP/d0:diskuse
     SECTORS
   TREE    LEVEL  DIRECTORY (CWD= /ACCT/GROUP/d0)
           BELOW
    960     240  ./
```

The next example illustrates the use of the TREE option. Information is reported for the *dir_name* (d3) and all directories below.

```
/ACCT/GROUP/d0:diskuse ./d3/@ ;tree
     SECTORS
   TREE    LEVEL  DIRECTORY (CWD= /ACCT/GROUP/d0)
           BELOW
    64        0  ./d3/d7/d9/
    208 + 144  ./d3/d7/
    208        ./d3/@
/ACCT/GROUP/d0:
```

MPE syntax can also be used, as shown in the following example (note that the *dir_name* (MYDIR) is upshifted.) This example is not based on the directory structure shown.

```
DISKUSE mydir.group.acct

       SECTORS
     TREE   LEVEL  DIRECTORY
            BELOW

     2100    330  /ACCT/GROUP/MYDIR
```

NOTE The output is presented in HFS syntax, even if the directory name is supplied in MPE syntax. If wildcards were used to specify the directory name in MPE syntax, then the final line of output is the user-supplied directory name (upshifted) in MPE format.

Wildcards can be used to see a "horizontal cut" of disk s pace usage at an arbitrary directory depth. Wildcarding can be used in TREE and NOTREE output, as shown in the following examples.

```
/ACCT/GROUP/d0:diskuse ./@
     SECTORS
   TREE    LEVEL  DIRECTORY (CWD= /ACCT/GROUP/d0)
           BELOW
    64 +       0  ./d1/
    448 +    240  ./d2/
    336 +    128  ./d3/
    848           ./@
/ACCT/GROUP/d0:
```

```
/ACCT/GROUP/d0:diskuse ./@/
     SECTORS
   TREE    LEVEL  DIRECTORY (CWD= /ACCT/GROUP/d0)
         BELOW
    64 +     0  ./d1/
    96      32  ./d2/d4/
    64       0  ./d2/d5/d8/
    128     64  ./d2/d5/
    112     48  ./d2/d6/
    448 +  240  ./d2/
    64       0  ./d3/d7/d9/
    208    144  ./d3/d7/
    336 +  128  ./d3/
    848         ./@
/ACCT/GROUP/d0:
```

The last line of output contains the directory name and the total number of sectors (under the TREE column). The final TREE number always equals the sum of all other TREE numbers for directories with the same number of components as contained in the user-specified name that are designated with a plus sign (+) in the TREE column. For example, if you specified a pathname with three components, then the sum of the TREE field for all directory names with exactly three components equals the final total value.

Related Information

Commands	LISTFILE, REPORT
Manuals	None.

DISMOUNT

Releases a volume set that was explicitly reserved by the user with a MOUNT or VSRESERVE command. The equivalent native mode command is VSRELEASE. (Native Mode)

Syntax

DISMOUNT[{ * *volumesetname* }] [*.groupname*[*.acctname*]]

Parameters

* or <blank> Specifies the home volume set for the group and account specified, or for the logon group and account if *groupname* or *groupname.acctname* is not specified.

volume- setname An artificial component of a volume set name used to maintain backward compatibility with MPE V/E. The *volumesetname* can be a maximum of 8 characters.

groupname Used only for compatibility with MPE V/E. The *groupname* can be a maximum of 8 characters.

acctname Used only for compatibility with MPE V/E. The *acctname* can be a maximum of 8 characters.

Operation Notes

The DISMOUNT command allows you to release a volume set that you explicitly reserved using the MOUNT or VSRESERVE command. You can request a release only for a volume set that you have reserved; you cannot alter the status of the volume set for other users.

Volume sets in MPE/iX are not tied to groups and accounts (this differs from the MPE V/E scheme of disk partitioning).

The naming convention for MPE/iX volume sets differs from the naming convention for MPE V/E private volumes. MPE/iX volume set names may consist of any combination of alphanumeric characters, including the period (.) and the underbar (_). The name must begin with an alphabetic character and consist of no more than 32 characters.

Table 3-3 is a comparison of naming conventions between the MPE/iX VSxxxxxx and MPE V/E xxxMOUNT commands

.

Table 3-3 Command Acceptance of Naming Conventions - DISMOUNT Command

Specify	MPE V/E xxxMOUNT Command Accesses	MPE/iX VSxxxxxx Command Accesses
myset.grp. acct	The volume set named myset.grp.acct.	The volume set named myset.grp.acct.
myset	The volume set named myset.logongrp.logon acct.	The volume set myset.
*.grp.acct	The home volume set of the group grp in account acct.	Causes an error.
myset_grp_ acct	Error (name component longer than eight characters).	The volume set named myset_grp_acct.
m_g_a	The volume set named m_g_a.logongrp.logonacct, provided it exists. If it does not exist, an error is reported.	The volume set named m_g_a.

In MPE V/E, the name V.G.A indicates that V is the name of a volume set, that G is the name of a group, and that A is the name of an account.

MPE/iX accepts the V.G.A. name in that form, but no interpretation is made as to the referencing of G and A. Instead, MPE/iX treats V.G.A as a single, long string name, just as it would treat A_VERY_LONG_NAME_FOR_SOMETHING.

As a convenience to established HP users, MPE/iX accepts the naming convention that was used for MPE V/E private volumes. DISMOUNT V.G.A will succeed. DISMOUNT V will access the same volume set, provided you are logged on to account A, group G. The MPE V/E commands are able to default the logon account and group.

However, VSRELEASE V succeeds only if a volume set V exists. The MPE/iX commands do not call up any default specifications for group and account. VSRELEASE V.G.A succeeds only if a volumeset V.G.A is on line. With all VSxxxxxx commands, the .G.A component of this name is interpreted as a string, neither more nor less specific than _G _A.

If a volume set is named according to the MPE V/E naming convention (V.G.A), you must use an unambiguous reference when using the MPE/iX volume set commands.

We recommend that you do no use the MPE V/E naming convention and the xxxMOUNT commands. Instead use the MPE/iX naming convention and the VSxxxxxx commands. Alternating between MPE V/E and MPE/iX commands may lead to confusion and, in some cases, may lead to errors. For example, MOUNT X used in a job stream attempts to access a volume set named X.logongrp.logonacct, which may or may not be your intention.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. Use volumes (UV) or create volumes (CV) capability is required to use this command.

Examples

To release the volume set MYSET.B.C, that was previously reserved with a MOUNT or VSRESERVE command, enter:

```
DISMOUNT MYSET.B.C
```

You may also use the VSRELEASE command:

```
VSRELEASE MYSET.B.C
```

Related Information

Commands	MOUNT, LMOUNT, DSTAT, VSRESERVE, VSRELEASE
Manuals	*Volume Management Reference Manual*

DO

Allows the user to reexecute any command still retained in the command line history stack. It also permits the user to edit the command before reexecuting it, but without having to use the interactive mode of the REDO command. (Native Mode)

Syntax

DO[CMD=*cmdid*] [;EDIT=*editstring*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

cmdid The command to reexecute. The command may be specified by its relative or absolute order in the command line history stack, or by name (as a string), in whole or in part. The default is -1, the most recent command. MPE/iX detects an error if *cmdid* does not exist in the command line history stack. Table 3-4 defines the DO command directives.

Table 3-4 DO Command - Reexecute Directives

cmdid	**Executes**
(omitted)	Previous command.
-*n*	The *n*th command before the most recent one, where *n* is a number in the command line stack relative to the most recent command, which is -1.
m	Command number *m* in the command line stack. The number *m* is absolute (not relative).
string	The most recent command beginning with *string*.

editstring String specifying the edit to be performed on *cmdid* before it is reexecuted. If you omit *editstring*, the command is reexecuted immediately, with no editing performed.

If you specify *editstring*, it must appear, character for character, and space for space, exactly as it would if you were using the REDO command in interactive mode.

The editing directives used in *editstring* are defined in Table 3-5

Table 3-5 Editing Directives for the DO Command

Directive	Effect
i	INSERT. If text follows the i, the text following i is inserted in the current line at the position after the i.
r	REPLACE. If text follows the r, the text following r replaces the same number of characters in the current line, beginning at the position of r.
d	DELETE. Deletes a character from the current line for each specified in the edit line. Note that "d d" does not specify a range but simply deletes one character from the position above each d. Multiple d's may be followed by an insert or replace operation.
dw	DELETE WORD. Deletes a word starting at the letter d. A word is defined as all characters except a space, comma, or semicolon. If you place the d directly beneath a word delimiter, then the word and the delimiter characters are deleted. If no word exists on the command line, no delete occurs. You may follow this directive with other edits.
d*delim*	DELETE TO DELIMITER. Deletes all characters starting at the position of the d and ending at, but not including, the specified delimiter. If *delim* is not found, no delete occurs. You may follow this directive with other edits.
d>	DELETE TO EOL. Deletes to the end of the current line from the position specified by d>. It may be followed by an INSERT or REPLACE operation.
^	UPSHIFT. Upshifts the character positioned at the ^. You may specify multiple ^ characters to upshift a series of characters. Or, you may type multiple ^ characters, followed by spaces, then followed by more ^'s to upshift some characters while skipping others. You may follow this directive with other edits.
^w	UPSHIFT WORD. Upshifts the word starting at the position specified by ^. A word is defined as all characters except a space, comma, or semicolon. If you place the ^ directly beneath a word delimiter, the delimiter is skipped and only the word is upshifted. If no word exists on the command line, no upshift occurs. You may follow this directive with other edits.
^*delim*	UPSHIFT TO DELIMITER. Upshifts all characters starting at the position specified by the ^ and ending at, but not including, the specified delimiter. If *delim* is not found, no upshift occurs. You may follow this directive with other edits.
^>	UPSHIFT TO EOL. Upshifts all characters starting from the position specified by the ^ to the end of the current line. You may follow this directive with other edits.
v	DOWNSHIFT. Downshifts the character positioned at the v. You may specify multiple v's to downshift a series of characters. Or, you may type multiple v's, followed by spaces, then followed by more v's to downshift some characters while skipping others. You may follow this directive with other edits.
vw	DOWNSHIFT WORD. Downshifts the word starting at the position specified by v. A word is defined as all characters except a space, comma, or semicolon. If you place the v directly beneath a word delimiter, the delimiter is skipped and only the word is downshifted. If no word exists on the command line, no downshift occurs. You may follow this directive with other edits.

Directive	Effect
v*delim*	DOWNSHIFT TO DELIMITER. Downshifts all characters starting at the position of the v and ending at, but not including, the specified delimiter. If *delim* is not found, no downshift occurs. You may follow this directive with other edits.
v>	DOWNSHIFT TO EOL. Downshifts all characters starting from the position specified by the v to the end of the current line. You may follow this directive with other edits.
>*text*	APPEND. The > followed by text appends the text to the end of the current line. If > is positioned beyond the end of the current line, then a replacement is performed instead.
>d	DELETE FROM EOL. Deletes from the end of the current line, right-to-left. Multiple d's may be specified after >, as well as INSERT and REPLACE strings.
>dw	DELETE WORD FROM EOL. Deletes the last word in the command line. To find the last word, trailing word delimiters are skipped. If no word exists in the command line, then none is deleted. If you follow >dw with additional editing directives, each edit is performed recursively. That is, the first edit is performed (updating the current EOL), then the next edit is performed (again updating the current EOL), and so on.
>d*delim*	DELETE TO DELIMITER FROM EOL. Starting at the end of the current line, deletes all characters right-to-left up to, but not including, *delim*. If the delimiter is not found, no delete occurs. If you follow this directive with additional editing directives, each edit is performed recursively. That is, the first edit is performed (updating the current EOL), then the next edit is performed (again updating the current EOL), and so on.
>^	UPSHIFT FROM EOL. Upshifts the character at the current EOL. You may specify multiple ^'s to upshift a series of characters (read right-to-left) from the EOL. Also, you may follow this directive with other edits.
>^w	UPSHIFT WORD FROM EOL. Upshifts the last word in the command line. You may follow this directive with other edits.
>^*delim*	UPSHIFT TO DELIMITER FROM EOL. Starting at the end of the current line, upshifts all characters right-to-left up to, but not including, *delim*. If the delimiter is not found, no upshift occurs. You may follow this directive with other edits.
>v	DOWNSHIFT FROM EOL. Downshifts the character at the current EOL. You may specify multiple v's to downshift a series of characters (read right-to-left) from the EOL, and you may follow this directive with other edits.
>vw	DOWNSHIFT WORD FROM EOL. Downshifts the last word in the command line. You may follow this directive with other edits.
>v*delim*	DOWNSHIFT TO DELIMITER FROM EOL. Starting at the end of the current line, downshifts all characters right-to-left up to, but not including, *delim*. If the delimiter is not found, no downshift occurs. You may follow this directive with other edits.
>r*text*	REPLACE. Replaces characters at the *end* of the command line. The replacement is done so that the last (rightmost) character of the replacement string is at the end of the line.

Directive	Effect
c	CHANGE. Changes all occurrences of one string to another in the current line when the search string and replace string are properly delimited. A proper delimiter is a nonalphabetic character (such as ', ", / or ,). The substitution is specified as: c<*delim*> *search-string*<*delim*> [*replace-string* [<*delim*>]]. Omitting the *replace-string* causes occurrences of *search-string* to be deleted, with no substitution.
u	UNDO. A single u in column one cancels the most recent edit of the current line. Using the UNDO command twice in a row cancels all edits for the current line and reestablishes the original, unedited line. If u is placed anywhere other than column one of the current line, then a simple replacement is performed. UNDO makes sense only if you have a line on which you have performed some editing that can be "undone."
other	Simple replacement. Any other character (not i, r, d, d>, >, >d, c, or u) causes that character to be replaced in the current line at the position indicated by the character. In fact, simple replacement also occurs for the editing characters i, r, c, or > if they are not followed by text; or if > appears at or beyond the current end of line.

NOTE A word is defined as a grouping of characters delimited by a space, comma, semicolon, =, (,), “, ‘, tab.

Operation Notes

Reexecutes the command specified by *cmdid*. The user may specify an optional edit string to edit the command before it is reexecuted. This command is a companion to the enhanced MPE/iX version of the REDO command. Unlike REDO, the DO command does not permit interactive editing.

If *editstring* is specified, the edit is performed on *cmdid* before the command is reexecuted. The *editstring* must appear exactly as it would if you were using the REDO command.

Both *cmdid* and *editstring* must be surrounded by quotation marks (either " or ') if they contain any delimiters such as `; " ' [, ], =,` or a space.

Use

This command is available in a session or in BREAK. It is not available in a job or from a program. Pressing **Break** terminates recursive command executions from the history/redo stack.

Editing Samples

Practical uses of the editing commands listed above are shown in Table 3-6.

Table 3-6 Editing Samples for the DO Command

Edit	Action
u	First occurrence undoes the previous edits. The u must be in column one.
u	Second occurrence undoes all edits on the current line. The u must be in column one.
rxyz	Replaces the current text with xyz starting at the position of r.
xyz	Replaces the current text with xyz starting at the position of x.
ixyz	Inserts xyz into the current line, starting at the position immediately before the i.
ddd	Deletes three characters, one above each d.
d xyz	Deletes a single character above the d, skips one space, then replaces the current text with xyz starting at the position of x.
ddixyz	Deletes two characters, then inserts xyz in the current line in the position before the i.
d d	Deletes one character above the first d, skips two spaces and deletes a second character above the second d. It does not delete a range of characters.
d d>xyz	Deletes a single character above the first d, skips two spaces and deletes to the end of the line beginning at the second d, and then appends xyz to the end of line.
>xyz	Appends xyz to the end of the current line.
>ddxyz	Deletes the last two characters from the end of the current line and then appends xyz to the end of the line.
>rxyz	Replaces the last three characters in the current line with xyz.
>ixyz	Appends xyz to the end of the line. In this case, the i command is superfluous, because > accomplishes the same result. Using >xyz would be sufficient.
c/ab/def	Changes all occurrences of ab to def, starting at c.
c"ab"	Deletes all occurrences of "ab" starting at c.
cxyz	Replace the current text with cxyz, starting at c. Because delimiters have been specified (as they were in the previous two examples), this is a simple replacement.
dw	delete the word starting at the d
>dw	delete the last word
^w	upshift the word starting at the **^**
>**v**w	downshift the last word

Examples

`DO PAS` Reexecutes the most recent command beginning with the string PAS.

`DO 10` Reexecutes command number 10 (absolute) on the command history stack.

`DO -2` Reexecutes the second-to-last command on the stack (one command before the most recent).

`DO -2, c/5A/5B` Change all occurrences of 5A to 5B in the command preceding the most recent one before reexecuting it. The default is -1.

`do ,c/5A/5B` Change all occurrences of 5A to 5B in the most recent command before reexecuting it.

`DO RUN, ">;DEBUG"` Append ;DEBUG to the most recent RUN command and then reexecute it.

`DO 'RUN MYP', '>;LIB=G'` Find the most recent command beginning with RUN MYP and append ;LIB=G before reexecuting it.

Related Information

Commands REDO, LISTREDO, WORD evaluator function

Manuals *Using the HP 3000 Series 900: Advanced Skills*

DOIONOW

Executes the changes to the I/O configuration made with the SYSGEN utility, while the system remains online.

Syntax

```
DOIONOW
```

Parameters

None.

Operation Notes

Use the `DOIONOW` command to start the online reconfiguration of your I/O devices.

Use

This command is available from a job, session, a program, or in BREAK. Pressing **Break** has no effect on this command.

Example

After you have made changes to the system I/O configuration with SYSGEN's I/O Configurator, enter:

```
:DOIONOW
```

Related Information

Commands	SYSGEN
Manuals	*Performing System Management Tasks System Startup, Configuration, and Shutdown*

DOWN

Removes a device from normal system use. This command does not apply to the system console or to disk drives.

Syntax

DOWN *ldev*

Parameters

ldev The logical device number of the device being taken offline.

Operation Notes

When the DOWN command is issued for a device that is in use, the request is responded to when the process currently accessing it releases the device.

The system console cannot be taken down. Any attempt to do so results 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 occur with no indication to the system operator or to the user. For this reason, it is very important that every device that is not fully operational (especially those that are powered down) be taken down 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 startup.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be used only from the console unless distributed to users with the ALLOW or ASSOCIATE command.

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
```

Related Information

Commands	`SHOWDEV`, `UP`, `ABORTIO`
Manuals	*Performing System Operation Tasks*

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 an HP 2608 or HP 2563 Line Printer.

filename The fully qualified name of a file containing the download control information.

nn The print position that the first byte of data assumes. 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.

Operation Notes

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 an HP 2631B printer, only the `MARGIN=`*nn* is allowed.

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 indicates this need by way of a FORMS message. Refer to "Examples."

CAUTION Do not issue a `DOWNLOAD` command to an HP 2608S while a spoolfile is ACTIVE. This makes the device UNAVAILABLE, and it remains so until the system is restarted with a `START RECOVERY`.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console unless distributed to users with the `ALLOW` or `ASSOCIATE` command.

Examples

To respond to a forms message such as the following:

```
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)
```

enter:

```
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
```

Related Information

Commands	`SHOWDEV`, `ABORTIO`
Manuals	*Performing System Operation Tasks*

DSTAT

Displays the current status of the disk drives on the system. (Native Mode).

Syntax

DSTAT[*ldev* ALL]

Parameters

ldev An integer specifying the logical device number of the disk drive whose status is requested.

ALL Displays the status of all disk drives, both system and nonsystem. The default is that if no parameter is included, only the status of nonsystem disks is displayed.

Operation Notes

The DSTAT command is used to display the current status of one or more disk drives on the system. For example:

```
DSTAT ALL

LDEV-TYPE   STATUS     VOLUME (VOLUME SET-GEN)

  1- 07935    MASTER     MEMBER1   (MPEXL_SYSTEM_VOLUME_SET-0)
  2- 07935    MEMBER     MEMBER2   (MPEXL_SYSTEM_VOLUME_SET-0)
  3- 07935    MEMBER     MEMBER3   (MPEXL_SYSTEM_VOLUME_SET-0)
  4- 07935    MEMBER     MEMBER4   (MPEXL_SYSTEM_VOLUME_SET-0)
 15- 07935    MASTER     MEMBER1   (USER_VOLUME_SET-0)
 16- 07935    MEMBER     MEMBER2   (USER_VOLUME_SET-0)
 17- 07935    UNKNOWN
```

Table 3-7 defines the various status responses.

Table 3-7 Disk Drive Status

Status	Meaning
UNKNOWN	A volume in the UNKNOWN state does not have a label that the system can recognize. The volume may be from another system, it may be a new disk pack, or it may be a volume that has been formatted. An UNKNOWN volume is available for initialization.
SCRATCH	A volume in the SCRATCH state can be initialized. It may contain data, but by scratching the volume, the user has indicated that the data is no longer needed.

Status	Meaning
LONER	The volume is in the LONER state when its master is not mounted or when the volume set is closed by the VSCLOSE command.
MASTER	A volume in this state is the master volume of a volume set. In order for the system to recognize the volume set, the master volume must be mounted.
MEMBER	A volume in this state belongs to a volume set whose master is mounted. If the master is not mounted, the volume is in the LONER state.

If you have purchased Mirrored Disk/XL, you may see PENDING or DISABLED as well. PENDING indicates the partner disk failed to mount; DISABLED indicates the volume is not abailable to the system due to a disk failure. If you have Mirrored Disk/XL you also may see the following suffixes in the status portion of the display:

-MD	Mirrored disk
-SU	Split user volume
-SB	Split backup volume

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

To display the status of LDEV 1, enter:

```
:
:DSTAT 1

LDEV-TYPE   STATUS  VOLUME (VOLUME SET-GEN)

1- 079371   MASTER  MEMBER1 (MPEXL_SYSTEM_VOLUME_SET-0)
:
```

Related Information

Commands	SYSGEN, LMOUNT, LDISMOUNT, MOUNT, DISMOUNT, VSRESERVE, VSRELEASE, VOLUTIL Utility
Manuals	*Volume Management Reference Manual*

ECHO

Displays a message on the standard list device. (Native Mode)

Syntax

ECHO[*message*]

Parameters

message	The message to be displayed to the `$STDLIST`.

Operation Notes

Displays its argument, *message*, on the standard list device (`$STDLIST`). The command ignores delimiters. Quotation marks are not required around *message*. The `ECHO` command does not perform dereferencing of any kind. If you want variable dereferencing you must use explicit dereferencing (`!`) in the argument. A null message (**Return**) displays a linefeed.

The `ECHO` command is not suppressed by `OPTION NOLIST` in a UDC or command file, or by any setting of the `HPMSGLEVEL` variable.

Use

This command is available in a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Examples

In the following example, although there is a variable named `a` that has a string value, `ECHO` simply displays the character `a` because no dereferencing has been specified.

```
SETVAR a, 'hi there'

ECHO a
a
```

This time `ECHO` is given the value of the variable `a` argument. Explicit dereferencing has been specified and the dereferencing is done before `ECHO` is executed.

```
ECHO !a
hi there
```

Two exclamation points are resolved to one exclamation point by string substitution, and MPE/iX is prohibited from making the value substitution (even number rule).

```
ECHO a
!a
```

Triple (or any odd number of) exclamation points treat the argument as `!a`, which resolves to `!` and `!a`, giving `!hi there` (odd number rule).

```
ECHO !a
!hi there
```

If you entered the following command line in a user command, you would see a message when an error occurred:

```
IF CIERROR <> 0 THEN
ECHO ** A CIERROR OCCURRED!: (CIERR !CIERROR) **
```

The first instance of CIERROR has no dereferencing, and so ECHO treats it literally. The second instance, !CIERROR, contains explicit dereferencing, and so MPE/iX substitutes a value for the system variable CIERROR before the message is displayed to $STDLIST. So, for example, if the program generated error 975, you would see this message:

```
** A CIERROR OCCURRED!: (CIERR 975) **
```

Related Information

Commands	CALC, SET, SETVAR, COMMENT, TELL, WARN
Manuals	Appendix A, "Predefined Variables in MPE/iX"

EDITOR

Starts the EDIT/3000 subsystem, which is used to create and manipulate ASCII text or program files.

Syntax

EDITOR[*listfile*]

Parameters

listfile Actual file designator of file to receive any output resulting from EDIT/3000 LIST and XPLAIN commands 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.

You cannot backreference the formal file designator EDTLIST as an actual file designator in the command parameter list. For further information, refer to the "Implicit FILE Commands for Subsystems" discussion of the FILE command.

Operation Notes

The EDITOR command starts the EDIT/3000 subsystem.

Use

This command may be issued from a session or job. It may not be used from a program unless the user or the program has process handling (PH) capability. It may not be used from BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Example

To run EDIT/3000 during a session and specify a line printer (device class LP) as the list device for offline output, enter:

```
FILE LISTFILE;DEV=LP
EDITOR *LISTFILE
```

Because the *listfile* is often a line printer, it is often defined with the FILE command and backreferenced as in the preceding example.

Related Information

Commands	BUILD, LISTF, LISTFILE, LISTEQ,FILE
Manuals	*EDIT/3000 Reference Manual*

ELSE

Provides an alternate execution sequence within an IF statement. (Native Mode)

Syntax

ELSE

Parameters

None.

Operation Notes

The ELSE command is used only in conjunction with the IF and ELSEIF commands. 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.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

The following job listing illustrates using the ELSE command:

```
!CONTINUE
!PASXL MYPROG,MYUSL
!IF JCW>=FATAL THEN
!  TELL USER.TECHPUBS;COMPILE FAILED
!ELSEIF JCW>=WARN THEN
!  TELL USER.TECHPUBS;COMPILE COMPLETED WITH WARNINGS
!ELSE
!  TELL USER.TECHPUBS;COMPILE COMPLETE WITH NO WARNINGS
!ENDIF
```

Related Information

Commands	DELETEVAR, ELSEIF, ENDIF, IF, SETJCW, SETVAR, SHOWJCW, SHOWVAR
Manuals	None

ELSEIF

Provides an alternate execution sequence within an IF statement. Native Mode

Syntax

ELSEIF *expression* [THEN]

Parameters

expression Logical expression, consisting of operands and relational operators. The THEN keyword is optional. It may be used or omitted and has no effect on the results. The operators listed in Table 3-8 may be incorporated in *expression*.

Table 3-8 Logical Operators - The ELSEIF Command

Logical operators:	AND, OR, XOR, NOT
Boolean functions and values:	BOUND, TRUE, FALSE, ALPHA, ALPHANUM, NUMERIC, ODD
Comparison operators:	=, <>, <, >, <=, >=
Bit manipulation operators:	LSL, LSR, CSR, CSL, BAND, BOR, BXOR, BNOT
Arithmetic operators:	MOD, ABS, * , / , + , -, ^ (exponentiation)
Functions returning strings:	CHR, DWNS, UPS, HEX, OCTAL, INPUT, LFT, RHT, RPT, LTRIM, RTRIM, STR
Functions returning integers:	ABS, LEN, MAX, MIN, ORD, POS, TYPEOF
Other functions:	FINFO, SETVAR

Operation Notes

The ELSEIF command is used only in conjunction with the IF command. The ELSEIF command provides a way of avoiding nested IF statements. ELSEIF has meaning only when used after an IF construct.

Any number of ELSEIF commands may follow an IF command. In contrast, only one ELSE command may follow an IF or ELSEIF command. Refer to the ELSE and IF commands.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect unless *expression* contains the INPUT evaluator function.

Example

The following example illustrates using the ELSE command with the IF command:

```
IF EXPN1 THEN
  ...
ELSE
  IF EXPN2 THEN
   ...
  ELSE
   IF EXPN3 THEN
     ...
   ELSE
     ...
   ENDIF
  ENDIF
ENDIF
```

The same result can be accomplished more efficiently by using the ELSEIF command:

```
IF EXPN1 THEN
  ...
ELSEIF EXPN2 THEN
  ...
ELSEIF EXPN3 THEN
  ...
ELSE
  ...
ENDIF
```

Notice that only one ELSE may follow an ELSEIF, while any number of ELSEIF commands may follow an IF.

Related Information

Commands	CALC, DELETEVAR, ELSE, ENDIF, IF, SETJCW, SETVAR, SHOWJCW, SHOWVAR
Manuals	None

ENDIF

Terminates an IF block. (Native Mode)

Syntax

ENDIF

Parameters

None.

Operation Notes

The ENDIF command is used to terminate an IF block. The IF command, the optional ELSE and ELSEIF commands, and the ENDIF command 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. If false and no ELSE exists, then execution continues following the ENDIF.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

The following examples show the IF block ending with the ENDIF command:

```
IF  logical_expression
ELSE  logical_expression
  .
  .
  .
ENDIF

IF  logical_expression
ELSEIF  logical_expression
  .
  .
  .
ENDIF
```

Related Information

Commands	IF, ELSE, ELSEIF
Manuals	None

ENDWHILE

Terminates a WHILE block. (Native Mode)

Syntax

`ENDWHILE`

Parameters

None

Operation Notes

This command terminates a conditional block that begins with a `WHILE` command. The `WHILE` and `ENDWHILE` commands constitute a WHILE block. The `WHILE` command evaluates an expression, and so long as that expression evaluates as true, the command(s) between `WHILE` and `ENDWHILE` are executed. If the expression evaluates as false, execution of the WHILE block ceases and execution passes to the command following `ENDWHILE`. Execution terminates if any command not protected by a preceding `CONTINUE` causes an error.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** terminates the `WHILE` command loop.

Example

The following is an example of a simple `WHILE` block:

```
WHILE logical_expression
   .
   .
   .
ENDWHILE
```

Related Information

Commands	`WHILE`
Manuals	None

EOD

Denotes end-of-data on input stream from a job file (from an input other than $STDIN). It also terminates data initialized by the DATA command. The colon (:) is a required part of this command. (Native Mode)

Syntax

EOD

NOTE	The "&" symbol has no meaning to the input spooler when it reads records because the CI is not involved at that point.

Parameters

None.

Operation Notes

The EOD command is used to signify the end of data whose beginning was signified by a DATA command. It is also used to signify the end of a data set that was read from the standard input device.

Although in most cases programmers use EOD for delimiting data, any record beginning with a colon may delimit the data. Using a record other than EOD for this purpose, however, depends upon whether the standard input file is opened with the file name $STDIN or $STDINX.

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. (EOD is not required when using the BASIC interpreter since the subsystem provides different conventions for delimiting data.)

An EOD causes the read of the FREAD intrinsic to return the CCG condition code to the calling program. This condition code indicates the end-of-file condition on the terminal. Table 3-9 defines the various end-of-file indicators.

Table 3-9 End-of-File Indicators

Type of File	Indicators
DATA file from standard input device (for jobs and sessions)	EOD - terminates $STDIN and $STDINX. : followed by any other character - terminates $STDIN.
DATA files	EOD JOB DATA

Use

EOD is available only in a job or a session that is submitted with the STREAM command. It cannot be used directly from $STDIN or from a program.

Examples

To terminate a data file entered by using the STREAM command for a session identified as SESS1,BLACK.ACCTSP, your data file would contain EOD as its last record, as follows:

```
DATA SESS1,BLACK.ACCTSP
   .
   data
   .
EOD
```

The following program is an example of how EOD is used to terminate a set of data entered through a standard input device:

```
FORTRAN

PAGE 0001  HP32102B.01.12  (C) HEWLETT-PACKARD CO. 1986

>$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
```

Related Information

Commands	DATA
Manuals	None

EOJ

Ends a batch job. (Native Mode)

Syntax

EOJ

NOTE The "&" symbol has no meaning to the input spooler when it reads records because the CI is not involved at that point.

Parameters

None.

Operation Notes

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). MPE/iX 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.

Use

This command may be issued from a job. It may not be used from a session, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

The following example shows how EOJ is used within a job file to terminate a batch job:

```
!JOB USER.PUBS
!RUN MYPROG1
!RUN MYPROG2
!EOJ
```

Related Information

Commands	JOB
Manuals	*Using the HP 3000 Series 900: Advanced Skills*

ERRCLEAR

Zeros out all HP predefined error-related variables. (Native Mode)

Syntax

ERRCLEAR

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

None

Operation Notes

This command is equivalent to the following:

- SETVAR CIERROR 0
- SETVAR HPCIERR 0
- SETVAR HPCIERRCOL 0
- SETVAR HPFSERR 0

Use

This command is available from a job or session. It is not available from a program or in BREAK. Pressing **Break** has no effect on this command.

Example

```
errclear
continue
run database
if hpcierr < 0 then
   echo database warning ![abs(hpcierr)] detected, proceeding...
elseif hpcierr > 0 then
   echo FATAL database error !hpcierr detected, halting...
   escape
endif
```

Related Information

Commands	ESCAPE, RETURN
Manuals	None

ERRDUMP

Allows a user to dump either the process or system error stack to a specified depth. (Native Mode)

Syntax

ERRDUMP[*errorstackdepth*] [;SYS]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

errorstackdepth The number of error stack messages to be printed. If the actual error stack size is less than the *errorstackdepth* then all messages on the error stack are printed with no warnings or errors.

The process error stack currently runs from zero to sixteen. The system error stack currently runs from zero to one hundred and twenty-seven. If the *errorstackdepth* specified is beyond the boundaries of the process error stack, the process error stack is not dumped, and CIERR 9155 is displayed, as follows:

```
INVALID PROCESS STACK DEPTH;
EXPECTED A VALUE 0 - 16 (CIERR 9155)
```

If the *errorstackdepth* specified is beyond the boundaries of the system error stack, (specified with SYS) the system error stack is not dumped, and CIERR 9156 is displayed, as follows:

```
INVALID SYSTEM STACK DEPTH;
EXPECTED A VALUE 0 - 127 (CIERR 9156)
```

An *errorstackdepth* value of 0 dumps the entire error stack. The default value is 0.

SYS The SYS option specifies that the system error stack is to be dumped. If the SYS option is not used, then the process error stack is dumped.

Operation Notes

The ERRDUMP command allows the user to dump either the process or the system error stack to a specified depth. If the depth specified is greater than the number of errors on the error stack, then all errors on the error stack are dumped without any warnings or errors.

If the user specifies an *errorstackdepth* outside of the boundaries of the error stack, an error message is displayed and the error stack is not dumped.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Examples

To obtain an error stack dump, enter:

```
ERRDUMP
```

A sample system response is:

```
TYPE MANAGER; THE END-OF-FILE HAS BEEN DETECTED.
FILE SYSTEM MESSAGE 1023.
```

Another example specifies that the system error stack be dumped:

```
ERRDUMP 1;SYS
```

A sample system response is:

```
THE STATUS OF THE TIME ENTRY IS NON-ACTIVE.
TUE, FEB 9, 1988, 12:18
```

ESCAPE

Allows the CI programmer to simulate all aspects of CI error handling. (Native Mode)

Syntax

ESCAPE[[CIERR=] *errnum*]

Parameters

ERRNUM Sets the CIERROR variable to the absolute value of errnum and the HPCIERR variable is set to errnum.

Operation Notes

The ESCAPE command causes control to leave all user commands (regardless of nesting levels) and return to the CI. Batch jobs terminate (unless a CONTINUE is in effect) and sessions issue the prompt.

If no CONTINUE is active, ESCAPE causes the CI to act as it would for any error: for sessions the user command environment is cleared and the prompt is displayed; jobs terminate.

If CONTINUE is active , then ESCAPE causes the CI to execute the second command after the CONTINUE. In the following example, the CI will execute cmd2 after the ESCAPE.

```
cmd1
CONTINUE
udc1
   ucmdA
   ucmdB
   ESCAPE
cmd2
```

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **BREAK** has no effect on this command.

Example

```
errclear
continue
run database
if hpcierr < 0 then
   echo database warning ![abs(hpcierr)] detected, proceeding...
elseif hpcierr > 0 then
   echo FATAL database error !hpcierr detected, halting...
   escape
else
.
.
.
endif
```

Related Information

Commands	ERRCLEAR, RETURN
Manuals	*MPE/iX Commands Reference Manual Volumes I and II*
	Command Interpreter Access and Variables Programmer's Guide

EXIT

Terminates the command interpreter. (Native Mode)

Syntax

EXIT

Parameters

None

Operation Notes

When you are using MPE/iX you can start another Command Interpreter by running it as a program. To do so, you enter CI.PUB.SYS, or simply CI. If you enter this command more than once, you will create levels of the CI program.

To determine what level of the command interpreter you are in, use the SHOWVAR HPCIDEPTH command. Then, to back out from the CI, enter the EXIT command. If the command interpreter is the root CI, EXIT is equivalent to BYE and ends the session. Otherwise, EXIT returns to the previous process. To go beyond HPCIDEPTH=2 requires process handling (PH) capability. To end a session without backing out of the CI level-by-level with the EXIT command, enter BYE.

Use

This command is available from a job or session. It is not available from a program or in BREAK. Pressing **Break** has no effect on this command.

Example

The following example shows how to determine what level of the CI you are in, and then, using the EXIT command, to back out to the root CI:

```
SHOWVAR HPCIDEPTH
HPCIDEPTH=2
EXIT
SHOWVAR HPCIDEPTH
HPCIDEPTH=1
```

To back out from the second level to the first, use the EXIT command. To back out from the session from any level, use the BYE command.

Related Information

Commands	BYE, HELLO, RUN, XEQ,SHOWVAR
Manuals	Appendix A, "Predefined Variables in MPE/iX"

4 Command Definition F-K

FCOPY

Invokes the FCOPY subsystem.

Syntax

FCOPY[*fcopycommand*]

Parameters

fcopycommand An FCOPY subsystem command. The FCOPY 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 are found in the "Examples" section of this command. Refer to the *FCOPY Reference Manual* (32212-90003) for more information.

Operation Notes

This command runs the FCOPY subsystem from MPE/iX. 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/iX.

Use

This command may be issued from a session, job, or in BREAK. To use this command from a program, the user or the program must have process handling (PH) capability. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To access FCOPY to execute multiple commands, enter:

```
FCOPY
HP32212A.03.24
FILE COPIER (C) HEWLETT-PACKARD CO. 1984

>
```

To access FCOPY to execute a single command and return control to MPE/iX, enter the command as follows:

```
FCOPY FROM=UDC.TECHPUBS;TO=TEMP;NEW

HP32212A.03.24 FILE COPIER (C) HEWLETT-PACKARD CO. 1984

EOF FOUND IN FROMFILE AFTER RECORD 23

END OF SUBSYSTEM
:
```

Related Information

Commands	`COPY`
Manuals	*FCOPY Reference Manual*

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 NS3000/XL AdvanceNet 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. NS3000/XL AdvanceNet is not part of the HP 3000 Series 900 Computer System Fundamental Operating System and must be purchased separately.

Syntax

```
FILE formaldisgnator=[ *formaldisgnator | $NULL $NEWPASS $OLDPASS $STDIN $STDINX $STDLIST
filereference  ]
[ :nodespec  ,filedomain  ]
[ :DEV=[ [ envname] #] [ device] [ ,[ outpri] [ ,numcopies]]]
[ ;VTERM] [ ;ENV=envfile[ :nodespec]]
[ ;option] [ ;access][ ; disposition]
[ ;DEFBLK | ;OPTMBLK]
```

Parameters

formaldesignator A formal file designator in the format:

filename[.*groupname*[.*accountname*]][:*nodespec*]

The *filename*, *groupname*, and *accountname* are the identifiers that form a fully qualified file name. Each identifier may contain from one to eight alphanumeric characters, beginning with an alphabetic character. This file name 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 NS3000/XL AdvanceNet subsystem. It is not part of the fundamental operating system. MPE/iX accepts this extended formal file designator, with a node specification following a colon (`:`), in the `FILE` and `RESET` commands and in the `FOPEN` and `HPFOPEN` intrinsics.

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 backreferenced form, **formaldesignator*, is valid only if it appears on the right side of the equal sign (=).

`$NULL` Actual file designator of a system-wide file that is always treated as an empty file. When `$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/iX, but no physical output is actually performed.

Do not specify parameters or options for `$NULL` files; if you do, you will receive an error.

`$NEWPASS` The system-wide name of the temporary job file. When `$NEWPASS` is closed, it is referenced by the name `$OLDPASS`. Opening `$NEWPASS` destroys any previous `$OLDPASS` temporary file.

`$OLDPASS` The system-wide name of the last temporary file that was closed as `$NEWPASS`.

`$STDIN` The system-wide name of the standard job input device. A colon (`:`) as the first character read on this file indicates end-of-file. You will receive an error if you specify the `DEV=` option, `VTERM` parameter, or any of the *option* parameters or options with `$STDIN`; there are restrictions on the *disposition* parameters and options as well.

`$STDINX` The same as `$STDIN` except that a colon can be read as the first character and received as data. An `EOD` produces an end-of-file on `$STDINX`.

You will receive an error if you specify the `VTERM` parameter or any of the *option* parameters or options with `$STDINX`; there are restrictions on the *disposition* parameters and options as well.

`$STDLIST` The system-wide name for the standard job or session list device. You will receive an error if you specify the `VTERM` parameter or any of the *option* parameters or options with `$STDLIST`; there are restrictions on the *disposition* parameters and options as well.

filereference The actual file designator of the file. If the name does not begin with a dot (.) or slash (/), the name is considered to follow standard MPE file naming syntax rules. File names will be in the following format:

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

Each identifier may contain from one to eight alphanumeric characters, beginning with an alphabetic character. The file name resolution is as follows:

- if *filename* = `FN`, look for file `FN` in the CWD (current working directory)
- if *filename* = `FN.GP`, look for file `FN` in group `GP` of the logon account (regardless of the current working directory)
- if *filename* = `FN.GP.AC`, look for file `FN` in group `GP` of account `AC`.

In a batch job, the file fails to open if the file has a lockword that is not specified in *filereference*. In a session, MPE/iX prompts you for a lockword if one exists.

If the name begins with a dot (.) or a slash (/), the name is considered to follow the HFS file naming syntax rules:

- File names are not upshifted.
- File names can be up to 255 characters in length for absolute pathnames and 253 characters for relative pathnames.
- File names can begin with, and contain, any of the following characters:

  a-z, A-Z, 0-9, _, -, .

File names are of the form

path/filename

where the *path/filename* combination may have a maximum of 255 characters. The expected behavior of the *path/filename* resolution is as follows:

- if *filename* = `..fn`, look for file `..fn` in the CWD (current working directory)
- if *filename* = `/fn`, look for file `fn` in root directory (/)
- if *filename* = `./fn`, look for file `fn` in the CWD
- if *filename* = `../fn`, look for file `fn` in parent directory
- if *filename* = `.fn`, look for file `.fn` in the CWD

If a file has a lockword, attempts to open the file with the HFS naming syntax fail unless the file also has an ACD which defeats the lockword. It is recommended that all lockwords be removed in favor of ACDs.

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.

The *nodespec* parameter does not function when used with HFS naming syntax.

If an environment identification appears in a file designation and in the `DEV=` option, an attempt to open the file (with the `FOPEN` or `HPFOPEN` intrinsic, for example) produces an error.

`$BACK` instructs MPE/iX to "hop backward" one node toward your local system to find the specified file. This works only if the `FILE` command is issued in a remote session. If the systems involved are connected in a local area network (LAN), one "hop backward" always means returning to your local system. The `$BACK` specification is the same as `DEV=#` without an environment name.

NOTE The *nodespec* parameter and `REMOTE` command are not part of the HP 3000 Series 900 Computer System Fundamental Operating System. The NS3000/XL AdvanceNet subsystem must be purchased separately. The *nodespec* parameter is optional; if you do not have NS3000/XL AdvanceNet, omitting the *nodespec* parameter makes no difference in the performance of the `FILE` command.

However, specifying *nodespec* on a system that does not have NS3000/XL produces an error. The *nodespec* parameter is controlled by the NS3000/XL subsystem. Refer to the *NS3000/XL User/Programmer Reference Manual* (36920-90001).

filedomain The domain of the file, which may be `NEW`, `OLD`, or `OLDTEMP`:

`NEW` Creates a new file, which is 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` or `HPFOPEN` intrinsic to create the file. Refer to the `BUILD` command in this chapter.

`OLD` An existing permanent file that was saved in the system or in a movable volume set domain. The file continues to exist after the current job or session ends. Use this parameter when you are creating a file equation that back references a device link file.

`OLDTEMP` 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*, logical device number, or an X.25 node name. 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 Series 900 Computer System Fundamental Operating System. The NS3000/XL AdvanceNet subsystem must be purchased separately. The *envname* parameter is optional: if you do not have NS3000/XL AdvanceNet, omitting the *envname* parameter makes no difference in the performance of the `FILE` command.

However, specifying *envname* on a system that does not have NS3000/XL produces an error. The *envname* parameter is controlled by the NS3000/XL subsystem. Refer to the *NS3000/XL User/Programmer Reference Manual* (36920-90001).

`DEV=` If you choose the `DEV=` option, it must be followed by at least one parameter (the parameter can be simply #). The `DEV=` parameter does not accept device names, volume classes, or volume names. 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.

device The logical device class name or logical device number of a device, such as a disk, tape, printer, or a terminal. The default is `DISC`.

If you are opening a file that is to reside on a movable volume set, you must specify a device class that includes the drives upon which the home volume set is mounted. The file is then allocated to any of the volume set's volumes that fall within that device class.

outpri The output priority requested for an output spool file. This may have a value of 1 (the lowest priority) to 13 (the highest).

numcopies The number of copies requested for an output spool file. The maximum number is 127.

VTERM Instructs MPE/iX to use reverse virtual terminal service instead of remote file access. 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 that support reverse virtual terminal. Refer to *Communicator 3000, Volume 2, Issue 6* (version G.02.00 of MPE V/E U-MIT).

envfile The name of a file containing printer environment information, which controls the print output formats on the printer. Not all printers support this feature/capability to accept environment information.

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 may contain specifications for page size, character fonts, forms, and other printer requirements to be used with the HP laser printing system. The file must be in a form suitable for downloading to the printer.

For example, to specify the environment file ACCTENV.HPENV.SYS to be used when printing, enter:

```
FILE ACCTLIST;DEV=ACCTPP;ENV=ACCTENV.HPENV.SYS
```

For information on creating an environment file for your specific printer, refer to the documentation that came with your printer.

The ENV= parameter in a FILE command overrides the environment specified in the FOPEN or HPFOPEN intrinsic.

If the ENV= parameter is used and the **formaldesignator* or *filereference* is omitted the parameter is ignored. Only a fully specified environment option overrides the environment option supplied by programmatic open. Any environment file specification for a subsequent FOPEN or HPFOPEN of the device file is ignored.

option Any valid option for the FILE command.

Syntax for Option

[;REC=[*recsize*] [,[*blockfactor*] [,[F U V B] [,BINARY ,ASCII]]]] [;DEN=[*density*]] [;DISC=[*numrec*] [,[*numextents*] [*,initialloc*]]] [;CODE=*filecode*] [;RIO ;NORIO] [;STD ;MSG ;CIR ;KSAMXL ;SPOOL] [;ULABEL=numlabels] [;KEY={ ^filereference keyinfo }] [;FIRSTREC=*recnum*] [;REUSE ;NOREUSE]

Parameters for Option

recsize Record size. A positive number indicates words; 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.

For byte-stream files, *recsize* is assigned a length of 1 byte.

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 is DISC with 1023 records.

For example, a fixed-length ASCII file with a record size specified as 11 bytes has only 11 bytes available for data in each logical record. However, to determine actual block size, 12 bytes are used for the record size (block size = 12 bytes multiplied by the *blockfactor*). If the file is specified as a binary file, the 11 bytes are rounded up to 12 bytes (6 words), all of which are available for each logical record.

This is the only option parameter that applies to $STDIN, $STDINX, or $STDLIST; if you specify other option parameters for these files, FILE returns an error.

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 set to 1 after using the original value along with *recsize* to calculate maximum logical record size and physical record size. (This does not apply to message files.) The *blockfactor* is ignored for undefined-length records. Maximum size is 255.

For byte-stream files, *blockfactor* is set to 1.

F, U, V or B Defines the format of the records of the file. A file may contain fixed-length records (F), undefined-length records (U), variable-length records (V), or byte-stream format (B). Default is F for disk files.

BINARY or ASCII Indicates the type of records. BINARY indicates binary-coded records and is the default. ASCII indicates ASCII-coded records.

Byte stream files are ASCII coded.

density Corresponds to tape densities in BPI (bytes-per-inch) for new files only. This parameter is only applicable when writing to a tape mounted on the HP 7976A, HP 7978A, or HP 7980 variable-density tape drive.

The density value from a file equation takes precedence over the density specified in FOPEN or HPFOPEN. The supported densities are 800, 1600, and 6250. For details on the operation of density selection, refer to the FOPEN and HPFOPEN intrinsics in the *MPE/iX Intrinsics Reference Manual* (32650-90028).

numrec Maximum number of logical records, for new files only. For fixed-length and undefined-length files, the maximum value allowed for this field is 2,147,483,647. 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.

numextents Maximum number of disk extents. This is a value from 1 to 32.

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 0.

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 or FLABELINFO intrinsic. For this parameter, any user can specify a positive integer ranging from 0 to 32,767 or a mnemonic name. Certain integers and mnemonics have been reserved for particular Hewlett-Packard defined meanings. 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 is implicitly changed to fixed-length record. 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 is 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 a RIO file. NOBUF is not normally used by the RIO user. Refer to the *MPE/iX Intrinsics Reference Manual* (32650-90028) for a discussion of relative I/O.

STD, MSG, CIR, KSAMXL, or SPOOL Defines the type of file. The default is STD (standard MPE/iX disk 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 (circular file) acts as 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 front of file. CIR cannot be simultaneously accessed by readers and writers.

`KSAMXL` specifies a native mode KSAM file (KSAM XL file).

`SPOOL` specifies an output spool file. No spooling attributes are initialized. PRI is set to 8 and number of copies to 1. No output device is set.

This spool file will not be linked to the spool file directory (SPFDIR) and, therefore, will not be printed unless it is subsequently linked to the SPFDIR with the `SPOOLF;PRINT` command. At that time, the target output device must be set according to the rules of that command. Use of the `SPOOL` option forces the `SAVE` disposition, overriding any user-specified disposition.

The characteristics of a file created with the `;SPOOL` keyword are:

- variable length records of 1008 bytes each
- a blocking factor of 1
- ASCII format
- permanent file
- a record limit of 1023
- undefined maximum number of extents, with 0 extents initially allocated

These characteristics override any other characteristics, such as binary format, which may be specified.

numlabels The number of user label records to be created for the new file. You can specify as many as 255 labels. This parameter applies to any type of file.

^filereference or *keyinfo* Information about KSAM XL key. *keyinfo* is the information, *^filereference* is a file containing *keyinfo*; the caret (^) means the contents of the file will be used.

Use the following format for *keyinfo*:

`;KEY= (`*keyspec;keyspec*`...) Where:` *keyspec* ::= *keytype,keylocation,keysize* [,DUP ,RDUP]

You must specify one *keyspec* for each key in the KSAM file. First, describe the primary key, followed by as many as 15 subsequent *keyspec*s, each describing an alternate key.

keytype KSAM key type, specified as BYTE, INTEGER, REAL, IEEEREAL, NUMERIC PACKED, OR *PACKED. Specify with the whole word, or initial: B, I, R, E, N, P, or *. If more than one is specified, spell the word out correctly. See *keysize* parameter.

keylocation Location of the first byte of the KSAM key within the data record counting from the first byte in the record. The first byte in the data record is always numbered 1. Only one key can start at each location. This parameter applies only to KSAM files.

keysize Length of the KSAM key, in bytes. This parameter is required for all key types. Different *keytypes* have different lengths, as described below:

BYTE	1 to 255 bytes
INTEGER	1 to 255 bytes
REAL	1 to 255 bytes
IEEEREAL	4, 8, or 16 bytes
NUMERIC	1 to 28 bytes
PACKED	1 to 14 bytes (odd number of digits)
*PACKED	2 to 14 bytes (even number of digits)

DUP or RDUP These two options apply only to KSAM files. Specify the DUP option if you want duplicate key values to be permitted. If you don't specify DUP, records with duplicate key values are rejected and an error message issued when such records are written to the file. When the DUP option is used, each new duplicate key is inserted at the end of the duplicate key chain. This maintains the chronological order of duplicate.

If you specify RDUP, duplicate keys are allowed; they are inserted randomly in the duplicate key chain. This method makes insertion of such keys faster, but does not maintain the chronological order of the duplicate key chain.

The default is that duplicate keys are not allowed.

recnum If you specify 1, record numbers in the new KSAM data file are numbered starting with 1. Otherwise, by default, record numbers start with 0. (Only 1 and 0 are acceptable.)

REUSE or NOREUSE This option is used only for new KSAM files.

If you specify the REUSE option, KSAM files are compacted by reusing deleted record space. If you also specify the DUP option for a key, duplicate records are placed chronologically at the tail of the file, and all nonduplicate records are assigned to the first available space.

Deleted record space will not be reused with the NOREUSE option, which is the default.

Syntax for Access

[;NOCCTL ;CCTL] [;NOMULTI ;MULTI ;GMULTI] [;NOMR ;MR] [;W
AIT ;NOWAIT] [;ACC=
[IN OUT UPDATE OUTKEEP APPEND INOUT]] [;BUF=[*numbuffers*] ;
NOBUF] [;LOCK ;NOLOCK] [;COPY ;NOCOPY] [;FORMS=*formsmsg*] [;EX
C ;SHR ;EAR ;SEMI] [;NOLABEL ;LABEL= [[*volid*] [,[IBM ANS] [,[*e
xpdate*] [,*seq*]]]]] [;FORMID=*formid*] [;PRIVATE]

Parameters for Access

NOCCTL or CCTL Indicates whether or not carriage-control characters are 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.

NOMULTI, MULTI, or GMULTI Indicates if the sharing of files in jobs and sessions is allowed. NOMULTI prohibits sharing files in MULTI mode and is the default. MULTI allows concurrent accesses 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 (that is parent-to-child processes) with MULTI. GMULTI is the same as MULTI except it allows accesses to be in different jobs/sessions.

NOMR or MR Indicates if 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 Indicates if 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/iX; 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 it also implies NOBUF.

Only MSG files may be opened in NOWAIT mode without privileged mode.

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. This parameter has no meaning for NM files.

NOLOCK or LOCK Indicates if dynamic locking and unlocking is to be permitted. NOLOCK prohibits dynamic locking/unlocking of file through the FLOCK and FUNLOCK intrinsics. LOCK allows dynamic locking and unlocking through FLOCK and FUNLOCK intrinsics. Default is NOLOCK.

COPY or NOCOPY Indicates if files can be copied. COPY allows MSG, KSAM, CIR, and SPOOL 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, that is, 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, results in a display of blanks and the associated control character letter when the forms message appears on the system console. Default is that no forms message is sent.

`EXC, SHR, EAR, SEMI` Indicates if shared or exclusive file access is allowed. `EXC` is exclusive access; after the file is opened, no other accesses are permitted. For message files, `EXC` means one writer and one reader. For circular files `EXC` means one reader *or* one writer. `SHR` is share access; after the file is opened other accesses are permitted. `EAR` is exclusive access for one writer; it allows multiple readers. `SEMI` is intended for use with message files; it 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` Indicates if this tape is labeled or unlabeled. `NOLABEL` specifies an unlabeled tape. `LABEL` specifies a labeled tape. Default is `NOLABEL`.

volid Up to six alphanumeric characters identifying a labeled magnetic tape volume. If a special character, such as # is specified, *volid* must be surrounded by quotation marks (for example, `FILE LT;DEV=TAPE; LABEL="#12345",ANS`).

`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` or `HPFOPEN`, then `NEXT` causes 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.

formid Applies only to output spoolfiles. A string of up to eight alphanumeric characters, beginning with a letter, which uniquely identifies a special form that is to be mounted. A message displaying this *formid* is printed on the system console or `$STDLIST` of the associated user of the spooled device. The spooler process then awaits verification that the special forms have been mounted before printing the file for which the *formid* was specified. The default is that no *formid* or message is displayed.

`PRIVATE` The `PRIVATE` option generates a spool file that may be accessed in privileged mode only. This means that the file is not accessible to normal users on the system. Private spoolfiles may not be saved or copied. They may only be purged, printed, or (within limits) altered by using the `SPOOLF` command instead of using the `PURGE` or `COPY` commands.

Syntax for Disposition

[;DEL ;TEMP ;SAVE ;SPSAVE]

Options for Disposition

`DEL` The file is deleted when closed.

`TEMP` The file is saved in the job/session temporary domain when closed.

`SAVE` The file is saved in the permanent file domain when closed.

`SPSAVE` If this parameter is used, the resulting spool file is created with SPSAVE disposition. This means the spool file is not to be purged after the last copy of it has been printed, but is instead retained in the OUT.HPSPOOL group.

This option is only valid for output spoolfiles. Private spoolfiles cannot be saved with the `SPSAVE` parameter.

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.

`DEFBLK or OPTMBLK` These two options apply only to KSAM files. DEFBLK specifies that the data block size will be the default data block size of 4096 bytes. OPTMBLK specifies that KSAMXL will select the optional data block size based on the record size. The default is DEFBLK.

Operation Notes

This command allows you to change the specifications for files at run time, including the devices on which they reside, overriding specifications supplied through the `FOPEN` or `HPFOPEN` 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. The FILE command is the only way you can control or change the programmatic file specifications without changing the code which calls FOPEN or HPFOPEN.

Use

This command may be issued from a session, a job, a program, or in BREAK. Pressing **Break** has no effect on this command.

Examples

To run the program MYPROG, which references two files by the file names (*formaldesignators*) SOURCE and DEST, but to use two existing disk files INX and OUTX as the actual files for the program, enter:

```
FILE SOURCE=INX
FILE DEST=OUTX
RUN MYPROG
```

Enter the following command to send the output to a new file FILEX. The parameters entered on the command line define FILEX as having 64-word fixed-length records, blocked two records per block in ASCII code; it is limited to 800 records among 10 extents, two of which are to be immediately allocated. When MYPROG closes the file, it will be permaently saved.

```
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 come from the parameters specified in the FOPEN or HPFOPEN 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 77/XL 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:

```
FTNXL ALSFILE
```

MPE/iX 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, or FTNLIST as actual file names. The use of FTNTEXT as a file name, as in the following example, is invalid because the implicit FILE command issued by the FORTRAN compiler then backreferences itself:

```
FTNXL *FTNTEXT
FILE FTNTEXT=*FTNTEXT
```

The following is an example of using the **formaldesignator*, in this case, specifying a file on magnetic tape used as a source file during FORTRAN compilation:

```
FILE SOURCE=TAPE1,OLD;DEV=TAPE;REC=-80
FTNXL *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.

The following example uses NMS file option SPOOL:

```
FILE MYSPOOL;DISC=3000,1,1;SPOOL
PRINT DOCFILE.MYGROUP.MYACCT,*MYSPOOL
```

Because the DEV= parameter of the FILE command is defaulted to disk, the result is an unlinked output spool file. To send this file to a printer, use the following command:

```
SPOOLF MYSPOOL;PRINT;DEV=LP
```

This links MYSPOOL to the SPFDIR using the default PRI (8) and number of copies (1). Note that the DEV= parameter is required with the SPOOLF;PRINT command to link the spool file to a target device. Failure to specify DEV= (or specifying an inappropriate DEV such as *disk*) results in an error message.

HFS Examples

```
FILE X=./my_file;SAVE
PURGE *X
```

To reference the device link file TAPE7 in a file equation, enter:

```
FILE T=TAPE7,OLD
```

Related Information

Commands	BUILD, LISTEQ, LISTFILE, RESET
Manuals	*MPE/iX Intrinsics Reference Manual*

FINDDIR (UDC)

The FINDDIR UDC executes the LISTFILE command to search for a directory.

NOTE System-defined UDCs are not automatically available. Your System Manager must use the SETCATALOG command to make these UDCs available for your use. For example:

```
SETCATALOG HPPXUDC.PUB.SYS;SYSTEM;APPEND
```

Syntax

FINDDIR[[DIR=] *dir_name*] [[START=] *start_dir*]

Parameters

Refer to the LISTFILE command for a complete explanation of the parameters used with the FINDDIR UDC. The following parameters are supported with the FINDDIR UDC.

dir_name A simple directory name, including wildcards. The *dir_name* is case insensitive. It cannot be a pathname. For example, abc, @bc, and [A-M]_@ are valid *dir_names*; while /ABC/, ./Mydir, and ABC.GRP are not valid *dir_names*. The *dir_name* is optional and defaults to @.

start_dir The name of the directory where the search is to begin. For example, /SYS/PUB. The default starting directory is the root directory (/).

Operation Notes

The FINDDIR UDC finds all directories matching *dir_name*, with the search beginning at *start_dir*.

The UDC executes the following form of the LISTFILE command:

```
LISTFILE start_dir ,6 ;SELEQ=[OBJECT=DIR] ;NAME=dir_name ;TREE
```

Use

This UDC may be issued from a session, a job, a program, or in BREAK. Pressing **Break** aborts execution.

Examples

Refer to the LISTFILE command later in this chapter for examples.

Related Information

Commands	LISTFILE, FINDFILE (UDC), LISTDIR (UDC)
Manuals	None

FINDFILE (UDC)

The FINDFILE UDC executes the LISTFILE command to search for a file.

NOTE System-defined UDCs are not automatically available. Your System Manager must use the SETCATALOG command to make these UDCs available for your use. For example:

```
SETCATALOG HPPXUDC.PUB.SYS;SYSTEM;APPEND
```

Syntax

FINDFILE[FILE=] *filename* [[START=] *start_dir*]

Parameters

Refer to the LISTFILE command for a complete explanation of the parameters used with the FINDFILE UDC. The following parameters are supported with the FINDFILE UDC.

filename A simple file name, including wildcards. The *filename* is case insensitive. It cannot be a pathname. For example, abc, @bc, and [A-M]_@ are valid *filenames*; while /ABC/, ./Mydir, and ABC.GRP are not valid *filenames*. The *filename* is required.

start_dir The name of the directory where the search is to begin; for example, /SYS/PUB. The default starting directory is the root directory (/).

Operation Notes

The FINDFILE UDC searches for all files matching *filename*, with the search beginning at *start_dir*.

The UDC runs the the following form of the LISTFILE command:

```
LISTFILE start_dir ,6 ;SELEQ=[OBJECT=FILE] ;NAME=filename ;TREE
```

Use

This UDC may be issued from a session, a job, a program, or in BREAK. Pressing **Break** aborts execution.

Examples

Refer to the LISTFILE command later in this chapter for examples.

Related Information

Commands LISTFILE, FINDDIR (UDC)

Manuals None

FORMSALIGN

Configures one spooled printer or a group of spooled printers related by device class, to conditionally enter into a forms message dialog with its operator (s) when the current spoolfile includes a forms message.

Syntax

```
FORMSALIGN[DEV=]{ldev | devclass | devname }
```

```
[;[DIALOG=]{{EACHCHANGE | EACHFILE | EACHCOPY }[,{FORMIDOVERRIDE |
NOFORMIDOVERRIDE}]}]
```

```
[ ;SHOW]
```

Parameters

ldev The logical device number of a printer. The printer must be configured as an MPE Type 32 device.

devclass The device class name of a class of printers. Each printer in the class must be configured as an MPE Type 32 device. The device class must begin with a letter and consist of eight or fewer alphanumeric characters.

devname The device name of a printer. The device name must begin with a letter and consist of eight or fewer alphanumeric characters. Users should note that it is not possible to have a device class name and a device name (which are the same). If you enter an alphanumeric character string, the command will search the device class list first, and then the device name list.

`EACHCHANGE` The spooler process conducts the forms message dialog only when the (case-insensitive) forms message of the current spoolfile differs from that of the previous spoolfile printed by that process when an overriding formid specification is not in effect. Two different spoolfiles with the same forms message will print without the forms message dialog if they are printed consecutively.

`EACHFILE` The spooler process conducts the forms message dialog whenever the spoolid of the current spoolfile differs from that of the previous spoolfile printed by that process, the current spoolfile contains a forms message and an overriding formid specification is not in effect. Two copies of the same spoolfile will print without the forms message dialog if they are printed consecutively.

`EACHCOPY` The spooler process conducts the forms message dialog for every copy of every spoolfile which contains a forms message if an overriding formid specification is not in effect.

`FORMIDOVERRIDE` This is a sub-parameter of the chosen `EACHxxx` keyvalue. With this feature selected, the Native Mode Spooler first checks its current and previous spoolfiles for the same non-blank, case-insensitive formid. If the formids match, both the DIALOG option for the spooler process and any forms message in the current spoolfile are ignored, and the forms message dialog is not activated. Identical formids override all other considerations.

Note that the DIALOG option is not changed. It is ignored as long as the two formids match.

If the two formids do not match, and the formid of the current spoolfile is not empty, then the spooler conducts the forms message dialog using the forms message of the current spoolfile.

If the current spoolfile has no forms message (even though it has a forms identification), the spooler:

- Conducts no dialog if standard forms are already mounted.
- Displays the STANDARD FORMS message if special forms are mounted.

If the two formids do not match because the current spoolfile has no formid and the previous spoolfile did, the spooler will always conduct a forms message dialog, again ignoring any setting of DIALOG. If the current spoolfile has a non-empty forms message, the spooler conducts a normal forms message dialog with the device operator. If the forms message is empty, and the device has special forms mounted, the spooler prompts the device operator to mount standard forms.

Once both the previous spoolfile and the current spoolfile have no formids, the spooler operates in accordance with the selected DIALOG option once more.

NOFORMIDOVERRIDE This is a sub-parameter of the chosen EACHxxx keyvalue. With this feature selected, the Native Mode Spooler ignores any and all formids associated with the current spoolfile or the previous spoolfile. The setting of the DIALOG option always determines the conditions under which the spooler process conducts the forms message dialog. The formid is then useful only as an item in a selection equation.

NOTE The setting of (NO)FORMIDOVERRIDE only affects the spooler's function during the forms message dialog. It has no effect on the use of the FORMID keyword in a selection equation of either the SPOOLF or LISTSPF command. It is still possible to select a subset of all spoolfiles to alter, delete, or display on FORMID=, regardless of the setting (NO)FORMIDOVERRIDE for a given device. They are totally independent of each other.

If the current spoolfile has no forms message but special forms are mounted on the device, the spooler always conducts the STANDARD FORMS dialog.

SHOW Specifying this option causes the configuration for the specified devices to be displayed. If no other parameters are used, the current configuration is displayed. If other parameters are used, the configuration is first updated and then displayed. If a device class is specified, the configuration for each device in the class is displayed.

If this option is omitted, there is no display.

Operation Notes

The FORMSALIGN command can be used on a spooled or an unspooled printer, or on a device class containing any mixture of spooled and unspooled printers. When used on a spooled printer, the specified options become effective on the next copy selected for printing on that device. The choices are retained until changed by another FORMSALIGN command, even if the printer should become unspooled and a new spooler process started for it.

When used on an unspooled printer, it presently has no effect but will be retained (unless changed by another FORMSALIGN command) and will become effective immediately upon spooling the printer. Files which include a forms message, and which are directed to an unspooled printer, always trigger a forms message dialog with the printer's operator. Any formid accompanying the file is irrelevant when the file is directed to an unspooled printer.

NOTE This command effects more than one device (if applied to all devices in a class). You may get warning messages for some devices and not others. A warning message on one or more devices affects only that device. The command will continue to execute until all selected devices have been configured or shown, or an error is detected. An error terminates the command.

The options specified in the FORMSALIGN command are stored in the appropriate device files. For example, options for LDEV 6 are stored in file 00000006.DEVICES.3000devs. This is why the options are retained even when no spooler process exists for LDEV 6.

However, these device files are reconstructed at each system startup. The FORMSALIGN options set at that time are EACHCHANGE, FORMIDOVERRIDE. Your SYSSTART file should include one FORMSALIGN command per device or class for which you want to set options other than the default.

Use

This command may be issued from a session, job, program, or in BREAK. Any user may execute this command with only the ;SHOW option to display current configuration. When changing configuration, this command may be executed only from the console or by any user who has been allowed the FORMSALIGN command with the ALLOW command. You can also execute this command by assigning a user the ASSOCIATE command and specifying the device.

Examples

To display the current configuration, enter:

```
FORMSALIGN LP;SHOW
```

A sample of the output might look like the following:

```
                      FORMID
LDEV    DEVNAME    DIALOG     OVERRIDE

  6    LDEV6     EACHCHANGE     YES
 14    LDEV14    EACHCOPY        NO
 15    LDEV15    EACHFILE       YES
 19    LDEV19    EACHCHANGE      NO
```

To conduct a forms message dialog for each copy of each file printed, enter:

```
FORMSALIGN 6;DIALOG=EACHCOPY,NOFORMIDOVERRIDE
```

You may also specify the system startup options, for example:

```
FORMSALIGN 6;DIALOG=EACHCHANGE,FORMIDOVERRIDE
```

Related Information

Commands	SPOOLER, ALLOW, ASSOCIATE
Manuals	*Performing System Operation Tasks* (32650-90137)

FORTGO

Compiles, prepares, and executes a compatibility mode FORTRAN 66/V program. FORTRAN 66/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately.

Syntax

FORTGO[*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 `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 *textfile* and *masterfile*. This can be any ASCII output file. The formal file designator is `FTNNEW`. Default is that the file is not written.

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 or two double quotation marks in a row, is considered part of the string, and, therefore, not the terminating delimiter.

`INFO=`*quotedstring* is used to pass initial compiler options to a program.

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 `FILE` command.

Operation Notes

The `FORTGO` command compiles, prepares, and executes a compatibility mode FORTRAN 66/V program. If you do not specify a source file, MPE/iX expects input from your standard input device. If you do not specify *listfile*, MPE/iX 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 MPE segmenter, and cannot be accessed.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile, prepare, and execute a FORTRAN 66/V program entered from the disk file SOURCE and transmit the resulting program listing to the disk 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 file name MTAPE. Output is sent to the printer, identified by the file name 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
```

Related Information

Commands	FORTPREP, FORTRAN, RUN, XEQ, PREP, SEGMENTER
Manuals	*HP FORTRAN/3000 Reference Manual*
	MPE Segmenter Reference Manual

FORTPREP

Compiles and prepares a compatibility mode FORTRAN 66/V program. FORTRAN 66/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately.

Syntax

FORTPREP[*textfile*] [,[*progfile*] [,[*listfile*] [,[*masterfile*] [,[*newfile*]]]]] [;INFO=*quo tedstring*]

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. When you omit *progfile*, the MPE segmenter creates the program file, which resides in the temporary file domain as `$OLDPASS`. To create your own program file, you must do so in one of two ways:

- By using the MPE/iX `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, 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.

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 or two double quotation marks in a row, is considered part of the string, and, therefore, not the terminating delimiter.

`INFO=`*quotedstring* is used to pass initial compiler options to a program.

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 FILE command.

Operation Notes

This command compiles and prepares a compatibility mode FORTRAN 66/V program into a program file stored on disk. If you do not specify a source file, MPE/iX expects the input from your standard input device. If you do not specify *listfile*, MPE/iX 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 MPE 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/iX creates 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, specify *progfile*.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile and prepare a FORTRAN 66/V 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 66/V source program from a text file named TEXTX into a program file named PROGX, with the 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
```

Related Information

Commands	FORTGO, FORTRAN, RUN, XEQ, PREP, SEGMENTER
Manuals	*HP FORTRAN/3000 Reference Manual* (30000-90040)
	MPE Segmenter Reference Manual (32650-60026)

FORTRAN

Compiles a compatibility mode FORTRAN 66/V program. FORTRAN 66/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately.

Syntax

FORTRAN[*textfile*] [,[*uslfile*] [,[*listfile*] [,[*masterfile*] [,[*newfile*]]]]] [;INFO=*quoted string*]

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, this parameter indicates that the USL file was created in one of four ways:

- By using the MPE/iX SAVE command to save 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 with the MPE/iX 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.

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.

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 or two double quotation marks in a row, is considered part of the string, and, therefore, not the terminating delimiter.

INFO=*quotedstring* is used to pass initial compiler options to a program.

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 FILE command.

Operation Notes

This command compiles a compatibility mode FORTRAN 66/V program into a USL file on disk. If you do not specify *textfile*, MPE/iX expects input from your standard input device. If you do not specify *listfile*, MPE/iX sends the listing to your standard list device.

If you create the USL file (using the MPE/iX 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 are stored in the temporary file $OLDPASS.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile a FORTRAN 66/V 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 is stored in the disk file MYLIST:

```
FORTRAN MYSOURCE,MYUSL,MYLIST;INFO= "$CONTROL BOUNDS"
```

To compile a FORTRAN 66/V program and store the object code into a USL file you create with the BUILD command, enter:

```
BUILD OBJECT;CODE=USL
FORTRAN SOURCE,OBJECT,LISTFL
```

To create a USL file with the BUILD command, the code must be specified.

Related Information

Commands	FORTGO, FORTPREP, RUN, XEQ, PREP, SEGMENTER
Manuals	*HP FORTRAN/3000 Reference Manual*
	MPE Segmenter Reference Manual

FREERIN

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 one to the configured maximum.

Operation Notes

A resource identification number is used to manage a resource shared by 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, that is, the user who actually issued the GETRIN command that allocated the RIN and assigned it a password.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** has no effect on this command.

Example

To release RIN 1, enter:

```
FREERIN 1
```

Related Information

Commands	GETRIN
Manuals	*MPE/iX Intrinsics Reference Manual*
	Resource Management Programmer's Guide

FTN

Compiles a compatibility mode FORTRAN 77/V program. FORTRAN 77/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. The native mode equivalent of this command is `FTNXL`.

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/iX `SAVE` command to save the 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/iX `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`.

quotedstring A sequence of characters between two single quotation marks or between two double quotation marks that specify compiler options. You may use the delimiter as part of the string so long as the delimiter appears twice. Any occurrence of two single or two double quotation marks in a row, is considered part of the string, and, therefore, not the terminating delimiter.

Operation Notes

The `FTN` command compiles a compatibility mode HP FORTRAN 77/V program and stores the object code in a user subprogram library (USL) file on disk. If *textfile* is not specified, MPE/iX expects the source program to be entered from your standard input device. If you do not specify *listfile*, MPE/iX 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.

You cannot backreference the formal file designators used in this command (FTNTEXT, FTNUSL, and FTNLIST) 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, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles an HP FORTRAN 77/V 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 an HP FORTRAN 77 program contained in the disk file FORTSRC, and stores the object program in the USL file FORTOBJ. The program listing is stored in the disk file LISTFILE:

```
BUILD FORTOBJ;CODE=USL
FTN FORTSRC,FORTOBJ,LISTFILE
```

Related Information

Commands	FTNGO, FTNPREP
Manuals	*HP FORTRAN 77/iX Reference*
	MPE Segmenter Reference Manual

FTNGO

Compiles, prepares, and executes a compatibility mode HP FORTRAN 77/V program. HP FORTRAN 77/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. The native mode equivalent of this command is the `FTNXLGO` command.

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 `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 or two double quotation marks in a row, is considered part of the string, and, therefore, not the terminating delimiter.

`INFO=`*quotedstring* is used in the HP FORTRAN 77/V programming language to pass initial compiler options to a program.

Operation Notes

The `FTNGO` command compiles, prepares, and executes an HP FORTRAN 77/V program. If *textfile* is omitted, MPE/iX expects input from your standard input device. If you do not specify *listfile*, MPE/iX 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 `$OLDPASS`, which is passed directly to the MPE segmenter. It cannot be accessed because the segmenter also uses `$OLDPASS` to store the prepared program segments, overwriting any existing temporary file of the same name.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile, prepare, and execute an HP FORTRAN 77/V program entered from your standard input device, with the program listing sent to your standard list device, enter:

```
FTNGO
```

To compile, prepare, and execute an HP FORTRAN 77/V program from the disk file FORTSRC and send the program listing to the file LISTFILE, enter:

```
FTNGO FORTSRC,LISTFILE
```

Related Information

Commands	FTN, FTNPREP, RUN, XEQ, PREP, SEGMENTER
Manuals	*HP FORTRAN 77/iX Reference*
	MPE Segmenter Reference Manual

FTNPREP

Compiles and prepares a compatibility mode HP FORTRAN 77/V program. HP FORTRAN 77/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. The native mode equivalent of this command is the FTNXLLK command.

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. When you omit *progfile*, the MPE segmenter creates the program file, which is 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/iX 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 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 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 or two double quotation marks in a row, is considered part of the string, and, therefore, not the terminating delimiter.

INFO=*quotedstring* is used in the HP FORTRAN 77/V programming language to pass initial compiler options to a program.

Operation Notes

The FTNPREP command compiles and prepares a compatibility mode HP FORTRAN 77/V program into a program file on disk. If you do not specify *textfile*, MPE/iX expects input from the current input device. If you do not specify *listfile*, MPE/iX 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 MPE 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.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles and prepares an HP FORTRAN 77/V program entered through your standard input device and stores the prepared program segments in the file $OLDPASS. The listing is printed on your standard list device.

```
FTNPREP
```

To compile and prepare an HP FORTRAN 77/V source program from the source file FORTSRC, store it in FORTPROG, and send the listing to your standard list device, enter:

```
FTNPREP FORTSRC,FORTPROG
```

Related Information

Commands	FTN, FTNGO, RUN, XEQ, PREP, SEGMENTER
Manuals	*HP FORTRAN 77/iX Reference*
	MPE Segmenter Reference Manual

FTNXL

Compiles an HP FORTRAN 77/iX program. HP FORTRAN 77/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP FORTRAN 77/iX is installed on your system. (Native Mode)

Syntax

FTNXL[*textfile*] [,[*objectfile*] [,[*listfile*]]] [;INFO=*quotedstring*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

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`.

objectfile Actual file designator of the object file, which is the output of the compiler. This file is stored in binary form and has a file code of either `NMOBJ` (`1461`) or `NMRL` (`1033`). Its formal file designator is `FTNOBJ`. If the *objectfile* parameter is omitted, the object code is saved to the temporary file `$OLDPASS` if it exists, or to `$NEWPASS` which then becomes `$OLDPASS`.

If you specify *objectfile*, the compiler stores the object file in a permanent file of the correct size, type, and name you specified.

If either a file of the same name or the default file `$OLDPASS` already exists, the new object code overwrites the old if the file code is `NMOBJ` or is appended to the old if the file code is `NMRL`. If the file code is `NMRL`, any existing version of the code module is first purged.

The compiler may issue an error message telling you that a new or existing object file is too small to contain the compiler's output or number of modules. In that case you must build a larger file or use the Link Editor to clean the `NMRL`. You may then recompile to the new file.

You may use the MPE/iX SAVE command to store `$OLDPASS` as a permanent file under another name.

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`.

quotedstring A string of no more than 255 characters (including the single or double quotation marks that enclose it).

The *info* string used in the HP FORTRAN 77/iX programming language to pass initial compiler options to the HP FORTRAN 77/iX compiler. HP FORTRAN 77/iX places a single dollar sign ($) before the *info* string and places the string before the first line of source code in the text file.

NOTE The formal file designators used in this command (FTNTEXT, FTNOBJ, 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 FILE command.

Operation Notes

The FTNXL command compiles an HP FORTRAN 77/iX program and stores the object code in a source file on disk. If *textfile* is not specified, MPE/iX expects the source program to be entered from your standard input ($STDIN). If you do not specify *listfile*, MPE/iX sends the listing to your standard list device ($STDLIST) and identifies it by the formal file designator, FTNLIST. If you omit the *objectfile* parameter, the object code is saved in the file domain as $OLDPASS. To keep it as a permanent file, you save $OLDPASS under another name.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles an HP FORTRAN 77/iX program entered from your standard input device and stores the object program in the object file $OLDPASS. The listing is then sent to your standard list device.

```
FTNXL
```

The next example compiles an HP FORTRAN 77/iX program contained in the disk file FORTSRC, and stores the object program in the object file FORTOBJ. The program listing is stored in the disk file LISTFILE.

```
FTNXL FORTSRC,FORTOBJ,LISTFILE
```

NOTE Program development in native mode uses the MPE/iX LINK command not the MPE V/E PREP command. This produces a significant change in the method of linking code.

If you have created a program called MAIN and a subprogram called SUB, each contained in a separate file, you might choose to handle it this way in MPE V/E:

```
 FTN MAIN, SOMEUSL
 FTN SUB, SOMEUSL
:
 PREP SOMEUSL, SOMEPROG
:
 RUN SOMEPROG
```

The second command appends the code from SUB to SOMEUSL.

However, LINK (in MPE/iX native mode) does not append SUB. On MPE/iX, you must compile the source files into separate object files and then use the Link Editor to link the two object files into the program file, as in this example:

```
 FTNXL MAIN, OBJMAIN
 FTNXL SUB, OBJSUB
:
 LINK FROM=OBJMAIN,OBJSUB;TO=SOMEPROG
:
 RUN SOMEPROG
```

On the other hand, if an NMRL is used instead of an NMOBJ, the above can be simplified to the following:

```
BUILD RLFILE;DISC=10000;CODE=NMRL
FTNXL MAIN, RLFILE
FTNXL SUB, RLFILE
LINK RLFILE,SOMEPROG
RUN SOMEPROG
```

Related Information

Commands	FTNXLGO, FTNXLLK, RUN, XEQ, PREP, SEGMENTER
Manuals	*HP FORTRAN 77/iX Reference*
	MPE Segmenter Reference Manual

FTNXLGO

Compiles, links, and executes an HP FORTRAN 77/iX program. HP FORTRAN 77/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP FORTRAN 77/iX is installed on your system. (Native Mode)

Syntax

FTNXLGO[*textfile*] [,[*listfile*]] [;INFO=*quotedstring*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

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.

quotedstring A run-time parameter for the compiler. It is a quoted string of no more than 255 characters (including the single or double quotation marks that enclose it). The *info* string is used in the HP FORTRAN 77/iX programming language to pass initial compiler options to the HP FORTRAN 77/iX compiler. HP FORTRAN 77/iX places a single dollar sign ($) before the *info* string and places the string before the first line of source code in the text file.

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 FILE command.

Operation Notes

The FTNXLGO command compiles, links, and executes an HP FORTRAN 77/iX program. If *textfile* is omitted, MPE/iX expects input from your standard input device. If you do not specify *listfile*, MPE/iX sends the program listing to the formal file designator FTNLIST (default is $STDLIST).

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. The Link Editor purges the object file and writes the linked program to $OLDPASS, which is then executed and may be executed repeatedly.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Example

To compile, link, and execute an HP FORTRAN 77/iX program entered from your standard input device, with the program listing sent to your standard list device, enter:

```
FTNXLGO
```

To compile, link, and execute an HP FORTRAN 77/iX program from the disk file FORTSRC and send the program listing to the file LISTFILE, enter:

```
FTNXLGO FORTSRC,LISTFILE
```

Related Information

Commands	FTNXL, FTNXLLK, LINK, RUN, XEQ, LINKEDIT Utility
Manuals	*HP FORTRAN 77/iX Reference*
	MPE Segmenter Reference Manual

FTNXLLK

Compiles and links an HP FORTRAN 77/iX program. HP FORTRAN 77/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if HP FORTRAN 77/iX is installed on your system. (Native Mode)

Syntax

FTNXLLK[*textfile*] [,[*progfile*] [,[*listfile*]]] [;INFO=*quotedstring*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

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 linked program is written. When you omit *progfile*, the MPE/iX Link Editor creates the program file, which is stored in the temporary file domain as `$OLDPASS`. If you do create your own program file, you do so by specifying a nonexistent file in the *progfile* parameter, in which case a job/session permanent 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`.

quotedstring A run-time parameter for the compiler. It is a quoted string of no more than 255 characters (including the single or double quotation marks that enclose it). The *info* string is used in the HP FORTRAN 77/iX programming language to pass initial compiler options to the HP FORTRAN 77/iX compiler. HP FORTRAN 77/iX places a single dollar sign ($) before the *info* string and places the string before the first line of source code in the text file.

NOTE The formal file designators used in this command (`FTNTEXT`, `FTNOBJ`, 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 `FILE` command.

Operation Notes

The FTNXLLK command compiles and links an HP FORTRAN 77/iX program into a disk file. If you do not specify *textfile*, HP FORTRAN 77/iX expects your input from your standard input device. If you do not specify *listfile,* HP FORTRAN 77/iX sends the listing output to your current list device.

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. The Link Editor overwrites *progfile* and writes the linked program to $OLDPASS, if *progfile* is omitted, which can then be executed.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles and links an HP FORTRAN 77/iX program entered through your standard input device and stores the linked program in the file $OLDPASS. The listing is printed on your standard list device.

```
FTNXLLK
```

To compile and link an HP FORTRAN 77/iX source program from the source file FORTSRC, store it in FORTPROG, and send the listing to your standard list device, enter:

```
FTNXLLK FORTSRC,FORTPROG
```

Related Information

Commands	FTNXL, FTNXLGO, LINK, RUN, XEQ, LINKEDIT Utility
Manuals	*HP FORTRAN 77/iX Reference*
	MPE Segmenter Reference Manual

GETLOG

Establishes a logging identifier on the system.

Syntax

GETLOG
***logid*;LOG=*logfile*{ ,DISC ,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 log file resides, DISC, TAPE, SDISC (serial disk) or CTAPE (cartridge tape).

password Logging identifier password, 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. The password is optional. if ;PASS= is entered without a password none is assigned.

AUTO Initiates an automatic CHANGELOG if the log file becomes full. This option is ignored if TAPE is specified.

NOAUTO Prevents initiation of an automatic CHANGELOG. A CHANGELOG is not performed if the log file becomes full.

Operation Notes

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 databases, so that each test task that runs writes to a logging file. This makes data recovery easier because you know where the task failed.

The creator of the logging identifier must have user logging (LG) or system supervisor (OP) capability to execute this command. Other users can be allowed access to this logging identifier by notifying them of the identifier and password. If a password is specified, it is 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 (for example *fname001*). This naming convention works in conjunction with the file set number to generate sequential file names automatically.

If a log file is restricted to a single volume or volume class when it is created with the BUILD command, then successive log files created by User Logging will have the same restriction.

If a new log file name is specified with the ALTLOG command, the links with any previous log file are 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 determine what logging identifiers currently exist.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. User logging (LG) capability is required to use this command.

Example

To create the logging identifier FINANCE and associate it with the disk log file A, enter:

```
GETLOG FINANCE;LOG=A,DISC
```

Related Information

Commands	ALTLOG, CHANGELOG, LISTLOG, OPENLOG, RELLOG, LOG, SHOWLOG, SHOWLOGSTATUS
Manuals	*System Startup, Configuration, and Shutdown Reference Manual* *User Logging Programmer's Guide*

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.

Operation Notes

The GETRIN command acquires a global RIN from the MPE/iX 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/iX Intrinsics Reference Manual* (32650-90028).

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/iX 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/iX rejects your request and issues 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.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

To acquire a global RIN and assign to it the password MYRIN, enter:

```
GETRIN MYRIN
```

MPE/iX responds with the RIN number assigned, for example:

```
RIN: 1
```

Related Information

Commands	FREERIN
Manuals	*Resource Management Programmer's Guide*

HEADOFF

Stops header/trailer output to a device. (Native Mode)

Syntax

HEADOFF ***ldev***

Parameters

ldev	The logical device number of the printer affected by the command.

Operation Notes

Header and trailer information appears before and after a file when it is printed. This information is not a part of the file's text. This information 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.

If output is directed to a line printer, MPE/iX 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 takes effect after the corresponding trailer is printed.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be executed only from the console unless distributed to users with the ALLOW or ASSOCIATE command.

Example

To stop header/trailer output to logical device number 6, enter:

```
 HEADOFF 6
```

Related Information

Commands	HEADON
Manuals	*Performing System Operation Tasks*

HEADON

Resumes header/trailer output to a device. (Native Mode)

Syntax

HEADON *ldev*

Parameters

ldev The logical device number of the printer affected by the command.

Operation Notes

Header and trailer information appears before and after a file when it is printed. This information is not a part of the file's text. This information 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, output is directed to a line printer, and MPE/iX 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 takes effect after the current output is complete.

The header/trailer facility is always enabled at system startup.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be executed only from the console unless distributed to users with the `ALLOW` or `ASSOCIATE` command.

Example

To resume header/trailer output to logical device number 6 enter:

```
 HEADON 6
```

Related Information

Commands	`HEADOFF`
Manuals	*Performing System Operation Tasks*

HELLO

Initiates an interactive session. (Native Mode)

Syntax

HELLO[*sessionname*,] *username* [/*userpass*] .*acctname* [/*acctpass*] [,*groupname*[/*grouppass*
]] [;TERM={ *termtype* *termname* }] [;TIME=*cpusecs*] [;PRI={ BS CS DS ES }
] [{ ;INPRI=*inputpriority* ;HIPRI }] [;INFO=*ciinfo*] [;PARM=*ciparm*]

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 is assigned.

username User name, established by the account manager, that allows you to log on to 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. The user password must be preceded by a slash (/).

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 account password must be preceded by a slash (/).

groupname Group name to be used for the local file domain and the CPU and connect-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 to log on to your home group. The group password must be preceded by a slash (/).

termtype or *termname* Determines terminal type characteristics. The *termtype* parameter determines the type of terminal used for input. MPE/iX uses this parameter to determine device-dependent characteristics such as delay factors for carriage returns. It must be 10 or 18. 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 user 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 characteristics of the terminal, including data flow control, block mode, read trigger, special characteristics, echo, line feed, parity, and printer control.

cpusecs Maximum CPU-time that a session can use, entered in seconds. When the limit is reached, the session is aborted. It must be a value from 1 to 32767. To specify no limit, enter a question mark (`?`), `UNLIMITED`, or omit the parameter. Default is no limit.

`BS`, `CS`, `DS`, or `ES` The execution priority queue that MPE/iX uses for your session, and also the default priority for all programs executed within the session. BS is the highest priority, ES is the lowest. If you specify a priority that exceeds the highest priority permitted for your account or user name by the system, MPE/iX assigns the highest priority possible below BS. DS and ES are intended primarily for batch jobs; their use for sessions is generally discouraged. For information on the guidelines for these priority queues, refer to the `TUNE` command. Default is CS.

CAUTION Use care in assigning the BS queue. Processes in this priority class can lock out other processes.

inputpriority or `HIPRI` Determines the input priority of the job. The *inputpriority* option is the relative input priority used in checking against access restrictions imposed by the jobfence. 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 the current jobfence set by the system operator, the session is denied access. Default is 8.

When logging on, the `HIPRI` option is used to either override the system jobfence or to override the session limit. When using the `HIPRI` option to override the jobfence, the system first checks to see if you have system manager (SM) or system operator (OP) capability. If you have either of these capabilities, you are logged on and your `INPRI` defaults to the system's jobfence and execution limit. If you do not have either of these capabilities, the system attempts to log you on using `INPRI=13` and succeeds if the jobfence is 12 or less, and if the session limit is not exceeded. Only users with SM or OP capability can use the `HIPRI` option to override the session limit to log on. 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)
```

ciinfo An `INFO` string to be passed to the command interpreter. For the MPE/iX CI, it is the first command to be executed by the command interpreter. This parameter replaces the `( ) COMMAND LOGON` command and approximates its function. The `( ) COMMAND LOGON` command caused the session to terminate after executing the specified command. In contrast, the *ciinfo* parameter does not terminate the session unless *ciparm* is set to 1, 3, or 5.

Running the CI as a child process in this way restricts the flexibility of *ciparm*. More flexibility is available by running the CI as a standalone program.

ciparm The command interpreter parameter number you wish to use. The MPE/iX command interpreter accepts the numbers listed below. If you enter any other value, it is treated as zero (0).

0, 2, 4 Executes logon UDCs and displays the CI banner and the welcome message. This is the default.

1, 3, 5 Same as 0, but the CI terminates after processing the `INFO=` string. If the `INFO=` string is not specified, the CI terminates after executing the first user-supplied command.

-1 Prohibits cataloging of UDCs and suppress the display of the CI banner and the welcome message. Invoking this level requires system manager (SM) capability.

-2 Same as -1, but the CI terminates after processing the *info*= command. Invoking this level requires system manager (SM) capability.

The MPE/iX CI distinguishes between *ciparm*s 1, 3, 5 and 0, 2, 4 when it is run from within the CI, that is, after the session has logged on.

If a user *without* SM capability uses -1 or -2, the system substitutes a parameter value of 0 and does NOT display an error message.

Operation Notes

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 *acctname* in your logon command or MPE/iX rejects your logon attempt and displays an error message. If your logon attempt is accepted, MPE/iX displays specific logon information and prompts you for your next MPE/iX command. In the following example, a user has logged on under the *username* `USER` and the *acctname* `TECHPUBS`:

```
MPE XL:HELLO USER.TECHPUBS
HP3000 Release: X.50.40  User Version : X.50.40  THU, DEC 8, 1994, 1:15 PM
MPE/iX HP31900 B.78.11 Copyright Hewlett-Packard 1987. All rights reserved.
 :
```

When you first access an MPE/iX system to log on, the `MPE iX:` prompt is displayed. When you log off using the `BYE` command, the following message is displayed:

```
 CPU=1. CONNECT=1. THU, DEC 8, 1994, 1:50 PM
```

The `RELEASE: V.UU.FF` number in the logon banner is determined by Hewlett-Packard at operating system build time and provides an identity for software releases (also known as the MIT). This number may not be changed. (Prior to MPE/iX release A.11.70, this was referred to as `BASE`.)

The `USER VERSION: V.UU.FF` can be assigned a value during a SYSGEN and allows you to identify any changes to your total software package such as patch level, third party software, or other specifics. Any ASCII character can be used. In prior releases, this number was printed out immediately after the MPE/iX product number HP31900.

The `PRODUCT V.UU.FF`, which now immediately follows the product number HP31900, is determined by Hewlett-Packard when a new version of MPE/iX is compiled. This `V.UU.FF` number cannot be changed and is used when entering a service request (SR) against the MPE/iX operating system product for that particular release.

If the system operator has set up a welcome message, it is displayed after the MPE/iX verification of your logon.

The session number assigned by MPE/iX uniquely identifies your session to MPE/iX and to other users. MPE/iX assigns such numbers to sessions in sequential order as they are logged on. If you are on a modem and do not log on 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 on to a new session.

In certain instances, you may be required to furnish information in addition to the user and account names in your `HELLO` command. This information includes:

- Group name
- One or more passwords
- Terminal type code

Use

This command may be issued from a session. It may not be used from a job, program, or in BREAK. Pressing **Break** does not abort the execution of this command, but may prematurely terminate the printing of the welcome message or the execution of any logon UDCs. If you are already in a session, `HELLO` terminates that session before beginning a new one.

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 want this group as a logon group, you need not specify it. MPE/iX automatically assigns the home group as your logon group when you log on. But if you want to use some other group as your logon group, you must specify that group's name in your logon command in this way:

```
MPE iX:HELLO USER.TECHPUBS,MYGROUP
```

If your user name is not related to a home group, you must enter a group name in your HELLO command, or your logon attempt is rejected.

Once you log on, if the normal (default) file security provisions of MPE/iX 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 *Accessing Files Programmer's Guide* (32650-60010).

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 user names and groups belonging to his account. If you are using an account, user name, or group (other than your home group) that has a password, you must furnish that password when you log on. Include the password after the name of the protected entity, separated from that name by a slash mark (/). (In MPE/iX, the slash denotes security.)

For instance, if the group XGROUP requires a password, and if you use this group as your logon group, you could enter the password in this fashion:

```
MPE iX:HELLO USER.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 on to the system, it is more convenient to have MPE/iX prompt you for any required passwords. You do this by omitting the passwords from the logon command. When you log on, the command is printed in the normal way; MPE/iX prompts you for the password, then turns echo off so that the password is not printed. If you enter the password incorrectly, the prompt reappears and you have two more chances to enter the password correctly. After the third incorrect entry, the message INCORRECT PASSWORD (CIERR 1441) is displayed. You must then press **Return** to receive a new prompt and then enter the HELLO command to start a new logon process. Echo is turned on after all passwords are read.

Terminal Types

MPE/iX must be able to determine certain characteristics about your terminal, such as input and output speed, in order to conduct a session. If you log on using a different type of terminal than the type the system manager has configured, you must specify your terminal type when you log on. Refer to appendix C, "Terminal and Printer Types."

```
MPE iX:HELLO USER.TECHPUBS;TERM=10
```

Example

When you initially log on to access MPE/iX, the system prompt appears as:

```
MPE iX:
```

When you subsequently log on to another account or group, the system prompt by default is a colon (unless you have altered it with the SETVAR HPPROMPT command) and appears as:

```
 :
```

To start a session named ALPHA, with the user USER, the account TECHPUBS, the group XGROUP, and the group password XPASS, enter:

```
MPE iX:HELLO ALPHA,USER.TECHPUBS,XGROUP/XPASS
HP3000 Release: X.50.40  User Version : X.50.40  MON, DEC 12, 1994, 7:15 AM
MPE/iX HP31900 B.78.11 Copyright Hewlett-Packard 1987. All rights reserved.
:
```

Related Information

Commands	BYE
Manuals	None

HELP

Accesses the help subsystem (Native Mode)

Syntax

Direct access:

HELP[{ *udcname* *commandname*[{ *keyword* ,ALL }] *commandfilename* *errormessage* *programfilename function name variable name* SUMMARY CLASS HELPSTUDY EXPRESSIONS| VARIABLES | OPERATORS | FUNCTIONS }]

Interactive (subsystem) access:

>*commandname*{ space or comma} [{ *keyword* ,ALL }]

```
HELPMENU
SUMMARY
CLASS
HELP
HELPSTUDY
```

Parameters

<omitted> If you specify the HELP command with no parameters, you enter the help facility subsystem in interactive mode. To return to the CI, enter E or EXIT. Refer to "Operation Notes."

udcname Any existing UDC. To display all UDCs within a UDC file, specify the PRINT command. Refer to *commandname*.

commandname Any MPE/iX command. MPE/iX displays the command name and syntax. In addition, a list of keywords for that command is displayed.

The HELP command also provides help on UDCs, command files, or program files. The search order is UDCs, built-in commands (MPE/iX), command files, and then program files. The search order for UDC's is user level, account level, and system level. The search order for command files and program files is determined by the contents of the CI variable HPPATH. If the user's HPPATH does not contain the name of the current group, the user can print a command file from the current group, but cannot get help information.

For UDCs and command files, help displays the text of the user command, unless the file contains the NOHELP option. In those cases, the display is suppressed. In the case of program files, help displays a header identifying it as a program file and the fully qualified file name of the program file.

function name Any CI evaluator function, eg: FINFO

keyword One of the keywords described under the command parameter. All commands have the following keywords:

PARMS — PARMS is short for parameter. 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 information, and an example of the command.

variable name — Any CI predefined variable, eg: HPLASTJOB

command-filename — Any existing command file. Refer to *commandname*, "Operation Notes," and "Examples."

errrormessage — Any MPE/iX error message. The keywords are:

CIERR*nn*

program- filename Any existing program file. Refer to *commandname*, "Operation Notes," and "Examples."

SUMMARY — A brief summary of changes found in MPE/iX, including a quick overview of the operation of the help facility.

CLASS — A list of MPE/iX commands by functional class.

HELPSTUDY — A beginner's guide designed to familiarize novice users with the fundamentals of MPE/iX commands and command syntax.

EXPRESSIONS — A description of CI expresssions

FUNCTIONS — A list of all CI evaluator functions

VARIABLES — A list of all CI predefined variables

OPERATORS — A list of expression operators, like +, -, etc.

HELP — The help facility entry on the HELP command.

ALL — Displays the entire table of contents and the contents of each *keyword* for the HELP command.

EXIT — Exits the help subsystem. Help for the CI EXIT command is not available in interactive mode. To get help for the CI EXIT command, specify the direct mode in the form HELP EXIT ALL.

Operation Notes

You use the HELP command to display information about MPE/iX in one of two ways: by omitting command parameters to enter the Help subsystem or by getting information about a single command from the colon prompt.

Using HELP as a subsystem

Enter the HELP command without specifying any parameters to invoke HELP as a subsystem. You will see the first screen of Help, called HELPMENU. It lists the choices available to you so that you can review the operation of Help and get a brief overview of the changes found in the MPE/iX operating system.

Once you are in the Help Subsystem, you display information by entering the name of the command, UDC, error message, variable, expression, function or other item that you want at the greater-than (>) prompt. For example:

```
 :HELP
 >FINFO

 Syntax:   FINFO(filename, option)

 Defn:    A CI evaluator function that returns information about
       the specified file.
 Type:    String, integer, or Boolean depending upon option.
 Example:  FINFO('x.pub',"EXISTS")
 Result:   TRUE
 Example:  FINFO('jeff',"eof")
 Result:   71495

The following table summarizes the options of the FINFO function.
The description includes the option number, one or more aliases,
the data type, and a brief description of the option.
Num  Alias            Data Type   Option Description
-  -            -

 0  EXIST            Boolean    Existence of file
 1  FILENAME ONLY        String    File name
(24/225) Continue?
```

To display information up to the next keyword or command, press **Return**. HELP provides a page break for every 23 lines of output and pressing **Return** allows you to continue.

Do *not* precede the command or item name with HELP, or you will get an error message. For example:

```
:HELP
>HELP FINFO
   ^
   Can't find this keyword.
```

To exit the Help Subsystem, enter E or EXIT' or press **Break**. To stop the display and return to a system prompt, enter **CTRL Y**. temporarily stops the display, enter **CTRL S**. Use **CTRL Q** to resume.

Using HELP in direct mode

Enter HELP followed by the name of the command, UDC, error number or other keyword to display the information you need without entering the Help Subsystem. 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*.

For example:

```
HELP ABORT
ABORT
Aborts current program or operation.
Syntax
  ABORT
KEYWORDS: PARMS,OPERATION,EXAMPLE
:
```

Notice that in direct mode, MPE/iX displays the CI prompt (:) once it has displayed the information you wanted.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command.

Examples

To see the parameters for the LISTFILE command, enter:

```
:HELP LISTFILE PARMS
```

To see examples of the STORE command, enter:

```
:HELP STORE EXAMPLES
```

To see the same information from within the Help subsystem, enter:

```
:HELP
>STORE EXAMPLES
```

To see a list of predefined variables in MPE/iX, at the colon prompt enter:

```
:HELP VARIABLES

Several global variables have been pre-assigned by the command
interpreter. They may be used anywhere you would use your own
variables.
All global variables are listed in the table below. To get help
with a specific variable, at the colon (:) prompt type "HELP"
followed by the variable name, for example, "HELP HPCIDEPTH".
```

```
At the Help facility prompt (>), simply type the variable name,
for example, "HPCIDEPTH".
Global Variable Types
=================================================================
      R      READ ONLY variable (cannot be modified).
      W      READ/WRITE variable (can be modified).
      JCW     A standard MPE/iX JCW.
      I      Integer format.
      B      Boolean format (TRUE/FALSE).
(24/225) Continue?
```

If `LINKALL` is a command file, `HELP` displays the file as follows:

```
HELP LINKALL.TEST.UI
User-Defined Command File:LINKALL.TEST.UI
   Parm streamflag=...
```

....

If `VERSION.PUB.SYS` is a program file, `HELP` displays:

```
HELP VERSION.PUB
program file: VERSION.PUB.SYS
```

If the UDC `LISTF` contains the `NOHELP` option (as shown in the sample below) the `HELP` command will suppress the listing of this UDC, and displays the text for the built-in command `LISTF` instead.

```
listf
option NOHELP
showme
*****
```

If the UDC `MYUDC` (which is not the name of any MPE/iX command) contains the `NOHELP` option, then the Help facility displays an error.

Related Information

Commands None

Manuals *System Startup, Configuration, and Shutdown Reference Manual*

IF

Used to control the execution sequence of a job, UDC, or command file. (Native Mode)

Syntax

IF ***expression***[THEN]

Parameters

expression Logical expression, consisting of operands and relational operators. The operators listed in Table 4-1 may be incorporated in *expression*.

Table 4-1 Logical Operators - The IF Command

Logical operators:	AND, OR, XOR, NOT
Boolean functions and values:	BOUND, TRUE, FALSE, ALPHA, ALPHANUM, NUMERIC, ODD
Comparison operators:	=, <>, <, >, <=, >=
Bit manipulation operators:	LSL, LSR, CSR, CSL, BAND, BOR, BXOR, BNOT
Arithmetic operators:	MOD, ABS, * , / , + , -, ^ (exponentiation)
Functions returning strings:	CHR, DWNS, UPS, HEX, OCTAL, INPUT, LFT, RHT, RPT, LTRIM, RTRIM, STS
Functions returning integers:	ABS, LEN, MAX, MIN, ORD, POS, TYPEOF
Other functions:	FINFO, SETVAR

The allowed operands are any variable, integer, string, or Boolean constants, and the MPE/iX reserved words are WARN, FATAL, SYSTEM, and OK.

Compound logical expressions can be formed using the AND, NOT, XOR, and OR logical operators, and nested within parentheses.

The THEN keyword is optional. It may be used or omitted and has no effect on the results.

Operation Notes

This command begins an IF block consisting of all the commands after the `IF` command up to, but not including, the next ELSE. ELSEIF, or ENDIF statement. The ELSE, ELSEIF, 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 30 levels so long as IF is used alone. In a case where IF is used with WHILE the total nesting of IF and WHILE blocks cannot exceed 30 levels. Each IF or WHILE block read by the Command Interpreter increments the nesting count even if it resides within a *different* UDC or COMMAND file.

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 or ELSEIF block (if one exists) is executed.

NOTE You may not write an IF construct in such a way that it physically crosses from one user command (UDCs or command files) to another.

Use

This command may be issued from a job, session, program, or in BREAK. Pressing **Break** has no effect unless *expression* contains the INPUT evaluator function.

Example

The following job listing illustrates the use of an IF statement with ELSE and ENDIF statements:

```
!CONTINUE
!PASXL MYPROG,MYUSL
!IF JCW>=FATAL THEN
!  TELL USER.TECHPUBS;COMPILE FAILED
!ELSE
!  TELL USER.TECHPUBS;COMPILE COMPLETED
!ENDIF
```

Related Information

Commands	`CALC`, `ELSE`, `ELSEIF`, `ENDIF`, `WHILE`, `ENDWHILE`, `ESCAPE`, `RETURN`
Manuals	Appendix B, "Expression Evaluator Functions"

INPUT

Permits the user to assign a value interactively to any variable that could otherwise be set with the SETVAR command. The user may also create an optional prompt string and have it displayed on $STDLIST before the value is read. (Native Mode)

Syntax

INPUT[NAME=] *varname* [;PROMPT=*prompt*] [;WAIT=*seconds*] [;READCNT=*chars*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

varname Any variable (that can be set with SETVAR) in which the input string from $STDIN is stored. If *varname* does not already exist, INPUT creates it.

prompt The prompt string that is to be displayed on the standard listing device. If *prompt* is omitted, nothing displays, but INPUT then waits for an input value to store in *varname*. To include delimiters, for example, a comma (,) or semicolon (;) as part of the prompt string, you must surround the entire prompt string with quotation marks (“ or ”).

seconds A positive value specifying the number of seconds for a timed read. If a value is assigned to *seconds*, the prompt waits *seconds* for input and then terminates the command. The default is zero, no time limit.

chars The number of characters you want read from $STDIN. If chars is specified as a negative number, INPUT uses the absolute integer value. The maximum allowed (and the default) is the maximum size of a CI variable, which is currently 1024 characters.

Operation Notes

This command allows the user to assign a value interactively to a variable. It also allows the user to create an optional prompt message that is displayed on the standard list device ($STDLIST) before the value is read. This command provides a way to establish an interactive dialog with an executing UDC or command file. If it does not already exist, the variable *varname* is always created by INPUT. If you want to delete *varname* before ending a session, job, or program, use DELETEVAR *varname*. Refer to the DELETEVAR command.

CI input redirection can be used to set *varname* to a record in a file.

NOTE If a colon (:) is read by the INPUT command at any level other than the root level CI, the error message END OF FILE ON INPUT.
(CIERR 900) is returned.

INPUT reads a value from the standard input device ($STDIN) and stores it as a string in the variable named *varname*. If *varname* does not exist, INPUT creates it. If *prompt* is omitted, nothing is displayed, and INPUT waits for an input value to store in *varname*. The variable *varname* can be used as you would use any other MPE/iX string variable.

CI input redirection can be used to set varname to a record in a file.

NOTE The INPUT command does not evaluate an expression before assigning its value to *varname*. The command recognizes only strings. Expressions such as 9 + 3 are treated as strings, even though they are not surrounded by quotation marks (“ or ”).

The user may optionally specify a timed read by creating a value for *seconds*. The pending read prompt is canceled after *seconds*. The INPUT command recognizes the HPTIMEOUT variable. The length of the timed read is *seconds* or HPTIMEOUT (in minutes), whichever is smaller. If a timed read (using *seconds* or HPTIMEOUT) expires, then the pending read terminates.

- If *varname* already exists and you enter a null (a **Return**), then the value of *varname* remains unchanged.
- The same thing happens if *varname* exists and *seconds* or HPTIMEOUT expires before a value for *varname* is entered. In this case, however, a warning occurs, and CIERROR is set to 9003.
- If *varname* does not exist and a null (a **Return**) is entered for the variable value, then *varname* is created and set to null (““).
- If *varname* does not exist and *seconds* or HPTIMEOUT expires, then *varname* is created and set to null (““), and CIERROR is set to 9003.
- If the timed read expires due to the value of the HPTIMEOUT variable, for example, HPTIMEOUT=1 (in minutes) and the user executes INPUT bleep,,65, then the session is logged off.

Use

This command is available in a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command, without creating or modifying *varname*.

Examples

The INPUT command does not evaluate expressions, it stores them as a string. For example, the command INPUT bleep accepts and stores input (*somevalue*). If you want *somevalue* treated as an expression and evaluated and the result assigned to bleep (as opposed to assigning the string representation of *somevalue*), use the SETVAR command after using the INPUT command:

```
INPUT bleep
SETVAR bleep !bleep
```

The first command reads whatever value you enter and sets bleep to the string representation of that input. The second command assigns bleep the (evaluated) value that you entered.

```
INPUT MYVAR <FILEONE
```

The above example reads the first record in FILEONE into the CI variable named MYVAR. In order to read the entire contents of a file INPUT must be in a WHILE loop and the while loop must have its $STDIN redirected to the file. Eg: READFILE <FILENAME , where READFILE looks like:

```
 SETVAR EOF FINFO(HPSTDIN, ‘EOF’)
  WHILE SETVAR (EOF, EOF-1) >=0 DO
   INPUT MYVAR
    ...
  ENDWHILE
```

Table 4-2 illustrates how the INPUT command functions.

Table 4-2 INPUT Command Function

INPUT bleep and the user responds with:	What is stored in bleep:	Value* of bleep after SETVAR bleep !bleep:
001	001	1 (integer)
“001“	“001“	001 (string)
TRUE	TRUE	TRUE (Boolean)
9+3	9+3	12 (integer)
Return	(null) or bleep is not modified if it already exists	<<error from the parser>>

* The result is an error if the user responds with an *unquoted* string:

```
  INPUT BLEEP,>
 >ABC Return
  SETVAR BLEEP !BLEEP
```

ABC is not a number. And, without quotes around it, ABC is not a string, either. If ABC is not a defined variable, it has no value to extract. So, the attempt to evaluate the result of explicitly dereferencing, !BLEEP produces an error. Refer to the SETVAR command.

Related Information

Commands	DELETEVAR, SETVAR, SHOWVAR, INPUT( ) function
Manuals	*Using the HP 3000 Series 900: Advanced Skills* (31126A Opt. 002)

JOB

Defines a job to be activated with the STREAM command or an input spooled device to run in batch mode. (Native Mode)

Syntax

```
JOB[ jobname,] username [ /userpass] .acctname [ /acctpass] [ ,groupname[ /grouppass]]
[TIME=cpusecs] [;PRI= BS | CS | DS | ES]
[;INPRI=inputpriority ;HIPRI] [ ;RESTART] [;JOBQ=queuename]
[;OUTCLASS=[[DEVICE][,[OUTPUTPRIORITY][ ,NUMCOPIES]]]]
[;TERM={termtype}][ ;PRIVATE][ ;SPSAVE]
```

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 User name, established by the account manager, that allows you to log on to 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. If a password exists, but is *not* supplied in the command syntax, the STREAM command will prompt you for it if:

- The STREAM command is invoked from a session.
- Neither $STDIN nor $STDLIST is redirected.
- The JOB command is a first level JOB command (it is not nested within a second level STREAM command).

If the password is supplied in the command syntax it must be preceded by a slash (/).

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. If a password exists, but is *not* supplied in the command syntax, the STREAM command will prompt you for it if:

- The STREAM command is invoked from a session.

- Neither $STDIN nor $STDLIST is redirected.
- The `JOB` command is a first level `JOB` command (it is not nested within a second level `STREAM` command).

If the password is supplied in the command syntax it must be preceded by a slash (`/`).

queuename The name of the job queue the job will execute in. The default job queue is HPSYSJQ, which is a global queue for all jobs not associated with an individual job queue

groupname Group name to be used for the local file domain and for 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 home group if one is assigned. (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 group password is not needed when you log on to your home group. It is needed when you log on under any other group for which a password exists. If a password is needed, but is *not* supplied in the command syntax, the `STREAM` command will prompt you for it if:

- The `STREAM` command is invoked from a session.
- Neither $STDIN nor $STDLIST is redirected.
- The `JOB` command is a first level `JOB` command (it is not nested within a second level `STREAM` command).

If the password is supplied in the command syntax it must be preceded by a slash (`/`).

cpusecs Maximum CPU time allowed job, in seconds. When this limit is reached, the job is aborted. This must be a value from 1 to 32,767. To specify no limit, enter a question mark or `UNLIM`, or omit this parameter. Default is a 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 the highest priority; ES is the lowest. If you specify a priority that exceeds the highest priority permitted for your account or user name by the system, MPE/iX assigns the highest priority possible below BS. DS and ES are intended primarily for batch jobs; their use for sessions is generally discouraged. DS is the default and the maximum priority, unless modified by system management.

NOTE 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 this chapter.

inputpriority or HIPRI Determines the input priority of the job. The *inputpriority* parameter is the relative input priority used in checking against access restrictions imposed by the jobfence. The *inputpriority* parameter 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 the current jobfence set by the system operator, the job 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 jobfence, or it can be used to override the job limit. When using the HIPRI option to override the jobfence, the system first checks to see if you have system manager (SM) or system operator (OP) capability. If you have either of these capabilities, you are logged on and your INPRI defaults to the system's jobfence and execution limit. If you do not have either of these capabilities, the system attempts to log you on using INPRI=13 and succeeds if the jobfence is 12 or less, and if the job limit is not exceeded. In attempting to override the job limit (to log on after the maximum number of jobs 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 does not override the job 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)
```

RESTART Request to restart a spooled job that has been interrupted by the system termination/restart. This parameter takes effect automatically when the system is subsequently restarted with the START RECOVERY option. The effect is to resubmit the job in its original form.

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 the 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 is generated. Default is defined in the system configuration.

NOTE Nonshareable 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. This parameter is used to select the next spooled device file (on disk) for output, from among all those contending for a specific printer. Must be a value from 1 (lowest priority) to 13 (highest priority). When *outputpriority* is 1, output is always deferred. To have output printed from disk, use an *outputpriority* of 2 or greater.

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 is issued. The command still executes with the default value of 1. If the number of copies is greater than 127, an error message is printed, and 127 copies are printed. Default is 1.

termtype The `TERM=` option is obsolete now that the `JOB` command cannot be used interactively. In order to maintain backward compatability, the *termtype* parameter is still parsed, but it is not used. If the `TERM=` option is used, a warning message will be displayed.

`PRIVATE` The `PRIVATE` option forces the job output `$STDLIST` to be a private spoolfile. The spoolfile is only accessible to privileged users on the system. Private spoolfiles may not be saved or copied. They may only be purged, printed, or (within limits) altered.

`SPSAVE` If this option is used, the resulting job output `$STDLIST` spoolfile is created with an `SPSAVE` disposition. This means that the spoolfile is not to be purged after the last copy of it has been printed, but is instead retained in the `OUT.HPSPOOL` group. `SPSAVE` may not be used if `PRIVATE` has been specified.

NOTE The "&" symbol has no meaning to the input spooler when it reads records because the CI is not involved at that point.

Operation Notes

The `JOB` command is not used at the colon prompt (`:`). Rather, it is used in interactive mode with the `STREAM` command at the > prompt, or within an input jobfile, created to define a batch job. The job defined with this command is then activated (executed) with the `STREAM` command.

The `JOB` command is preceded by an appropriate substitute prompt character for the colon prompt. By default, MPE/iX expects the exclamation point (`!`) to be used. The `JOB` command must be terminated with an `EOJ` command. Refer to the `STREAM` command.

When MPE/iX begins the job, it displays the following information on the list device:

- Job number, as assigned by MPE/iX to identify the job.
- Date and time.
- “HP 3000,“ and the modified and base MPE/iX *version.update.fix* numbers.

In the `JOB` command, as in the `HELLO` command, you must always supply your *username* and *acctname*, which you obtain from your account manager. If you omit either of these parameters, or enter them incorrectly, MPE/iX rejects your job and prints error messages on the standard listing device and the console. If your job is accepted, MPE/iX begins job processing. The job is entered with the `STREAM` command or through a spooled input device. Then the job is copied to an input spoolfile. The job is initiated from that spoolfile rather than the originating diskfile (in the case of the STREAM command) or device (in the case of the input spooled device). If the standard listing file is a line printer, MPE/iX prints a header page prior to listing the JOB command. (The system operator can disable the printing of this header page with the HEADOFF console command.)

The job number assigned by MPE/iX always uniquely identifies your job to MPE/iX and other users. MPE/iX 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. However, the following also may be required:

- Logon 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 enter the JOB command through a device other than a terminal, and the standard input device is different from the standard listing device, MPE/iX 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/iX for the omitted parameters appear on the job listing.

The STREAM command prompts for any necessary passwords that are not supplied in the command syntax if:

- The STREAM command is invoked from a session.
- Neither $STDIN nor $STDLIST is redirected.
- The JOB command is a first level JOB command (it is not nested within a second level STREAM command).

All UDCs are available from a job. Any subsystem or UDC that expects input from $STDIN requires that input within your job stream file.

Use

This command may be issued only from a job file. It may not be used from a session, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

The following example illustrates creating and using an ASCII file to define a batch job and then executing it with the STREAM command:

```
RUN EDITOR.PUB.SYS
/ADD
     1 !JOB WXYZ,WRITER.TEC
     2 !EDITOR
     3 TEXT ABC
     4 LIST ALL,OFFLINE
     5 EXIT
     6 !EOJ
      //
/KEEP MYJOB
/EXIT
:
 STREAM MYJOB
```

The following example shows using the JOB command in interactive mode with the STREAM command:

```
 STREAM
>!JOB USER.TECHPUBS;OUTCLASS=12
```

Related Information

Commands ABORTJOB, ALTJOB, BREAKJOB, SUSPENDJOB, RESUMEJOB, JOBFENCE, JOBPRI, STREAM, STREAMS, SHOWDEV, NEWJOBQ, LISTJOBQ

Manuals *Using the HP 3000 Series 900: Advanced Skills*

MPE/iX Intrinsics Reference Manual

JOBFENCE

Defines the minimum input priority that a job or session must have in order to execute. (Native Mode)

Syntax

JOBFENCE ***priorityfence***

Parameters

priorityfence A number between 0 and 14, inclusive. Within this range, smaller numbers are less limiting; larger numbers more limiting.

Operation Notes

MPE/iX does 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 log on with an input priority greater than the jobfence as reported by the SHOWJOB command.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console unless distributed to users with the ALLOW 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.SUPT
#J8   WAIT  D 8  10S 12   THU 4:18P JOB2,FIELD.SUPT
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 to allow waiting jobs to log on:

```
JOBFENCE 6

16:21/#J7/34/LOGON FOR: JOB1,FIELD.SUPT ON LDEV #10
16:21/#J8/35/LOGON FOR: JOB2,FIELD.SUPT ON LDEV #10
```

Related Information

Commands	ABORTJOB, ALTJOB, BREAKJOB, JOB, SUSPENDJOB, RESUMEJOB, JOBPRI, STREAM, STREAMS, SHOWDEV
Manuals	*Using the HP 3000 Series 900: Advanced Skills* *MPE/iX Intrinsics Reference Manual*

JOBPRI

Sets or changes the default execution priority for batch jobs and sets a maximum execution priority for batch jobs. (Native Mode)

Syntax

JOBPRI[*maxsubqueue*] [*,defaultsubqueue*]

Parameters

maxsubqueue	The maximum priority at which batch jobs are allowed to run. This overrides any job priority a user may have requested with the JOB command. This parameter may be ES, DS, CS, or zero. If zero is specified, no limit is imposed on batch jobs. Default is no change in maximum priority.
default- subqueue	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.

Operation Notes

The *maxsubqueue* parameter specified in the JOBPRI command takes precedence over *defaultsubqueue*. Therefore, selecting a default parameter greater than the value of *maxsubqueue* parameter does not affect job execution. Jobs are still initiated with the maximum priority parameter.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. System supervisor (OP) capability is required to execute this command.

Example

To raise the maximum execution priority so that batch jobs can run in any subqueue requested, enter:

```
JOBPRI 0
```

Related Information

Commands	TUNE, SHOWQ, ALTPROC
Manuals	*Performing System Operation Tasks*

JOBSECURITY

Designates what level of user may request resources and control the execution of jobs. (Native Mode)

Syntax

```
JOBSECURITY[ {   HIGH   LOW } { ;PASSEXEMPT=
{ NONE} ,{ USER} ,{ XACCESS} ,{ MAX} } ]
```

Parameters

HIGH — Permits only the operator logged on at the console and users with SM capability to use job control commands.

LOW — Allows individual users to exercise control over their own jobs.

<omitted> — If you do not specify HIGH or LOW, the current job security status is displayed (high or low).

NONE, USER, XACCESS, or MAX — The PASSEXEMPT option set by the system manager, which has the following meaning:

- NONE — All users must specify the required passwords to stream a job.
- USER — Allows certain users to omit a job's password. The system manager can omit the password when streaming any job, account managers can omit passwords when streaming jobs that log onto their account and to which they have access, and users can omit passwords for jobs that match their logon identity and to which they have access.
- XACCESS — Allows users with execute access to the job file to omit passwords when the job file logs on with the same identity as its owner or creator.
- MAX — Sets both the USER and the XACCESS options of the PASSEXEMPT parameter. Specifying MAX is the only way to set both options since USER and XACCESS are otherwise mutually exclusive.

Operation Notes

The HIGH and LOW parameters of the JOBSECURITY command determine what kind of user may execute the ABORTJOB, ALTJOB, BREAKJOB and RESUMEJOB commands. When JOBSECURITY is set to HIGH, only the operator may issue these commands. When it is set to LOW, any user may issue these commands for their own jobs (i.e., those where the job's user name and account matches the user's) and Account Managers may control the execution of any job in their account.

System managers may use the PASSEXEMPT parameter of the JOBSECURITY command to control password validation when users stream a job. If you have never used the PASSEXEMPT parameter and if the HP Security Monitor is not installed, the initial state is NONE, which means that job passwords are required. When you reboot the system with a START RECOVERY the last PASSEXEMPT state is preserved.

PASSEXEMPT provides some of the functionality of the HP Security Monitor. For example, PASSEXEMPT=USER is equivalent to the stream privilege feature. PASSEXEMPT=XACCESS is similar to the stream authorize feature with one difference: you may set the USER XACCESS options independently, whereas HP Security Monitor requires you to enable stream privilege when you want to enable the stream authorize feature.

JOBSECURITY checks for the existence of HP Security Monitor and, if necessary, combines the settings to produce appropriate output. When the PASSEXEMPT parameter is issued and the interaction with the HP Security Monitor produces a different result, you will see a warning and a notification that the HP Security Monitor is installed. The resulting command output is also displayed with the warning.

Use

You may issue the JOBSECURITY command from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be executed only from the console unless distributed to users with the ALLOW command.

Example

To allow any user to abort, alter, break, or resume their own jobs, enter:

```
JOBSECURITY LOW
```

To find out the current job security status, enter:

```
:JOBSECURITY
JOB SECURITY IS HIGH. PASSEXEMPT IS NONE.
```

To set the password exemption to USER and then check the current status, enter:

```
:JOBSECURITY ;PASSEXEMPT=USER
:JOBSECURITY

JOB SECURITY IS LOW. PASSEXEMPT IS USER.
```

Suppose PASSEXEMPT is currently set to USER and you want to change it to XACCESS. To do so, enter:

```
:JOBSECURITY ;PASSEXEMPT=XACCESS
```

Then check the current status by entering:

```
:JOBSECURITY
JOB SECURITY IS LOW. PASSEXEMPT IS XACCESS.
```

If the HP Security Monitor is installed with both stream privilege and authorization turned on, the JOBSECURITY command will display a warning when the output produces a different result.

```
:JOBSECURITY ;PASSEXEMPT=USER
Security Monitor is installed. Passexempt is MAX. (CIWARN 3128)
```

Related Information

Commands	ABORTJOB, ALTJOB, BREAKJOB, RESUMEJOB, JOBFENCE
Manuals	*Performing System Operation Tasks*

LDISMOUNT

Cancels a previously issued LMOUNT or VSRESERVE command. This informs the system that the volume set is no longer reserved system-wide. The equivalent native mode command is VSRELEASESYS. (Native Mode)

Syntax

LDISMOUNT[{ * | | *volumesetname* }] [*,groupname*[*.acctname*]]

Parameters

* or <blank>	Specifies the home volume set for the group and account specified, or for the logon group and account if *groupname* or *groupname.acctname* is not specified.
volumesetname	An artificial component of a volume set name used to maintain backward compatibility with MPE V/E. The *volumesetname* can be a maximum of 8 characters.
groupname	Used only for compatibility with MPE V/E. The *groupname* can be a maximum of 8 characters.
acctname	Used only for compatibility with MPE V/E. The *acctname* can be a maximum of 8 characters.

Operation Notes

The LDISMOUNT command negates a previously issued LMOUNT or VSRESERVE command. It informs MPE/iX that the volume set is no longer reserved system-wide.

Volume sets in MPE/iX are not tied to groups and accounts. This is different from the MPE V/E scheme of disk partitioning.

Table 4-3 is a comparison of naming conventions for MPE/iX volume sets and MPE V/E private volumes. MPE/iX volume set names may consist of any combination of alphanumeric characters, including the underbar (_) and the period (.). The name must begin with an alphabetic character and consist of no more than 32 characters.

Table 4-3 Command Acceptance of Naming Conventions - LDISMOUNT Command

Specify	MPE V/E xxxMOUNT Command Accesses	MPE/iX VSxxxxxx Command Accesses
myset.grp.acct	The volume set named myset.grp.acct.	The volume set named myset.grp.acct.
myset	The volume set named myset.logongrp.logonacct.	The volume set myset.

*.grp.acct.	The home volume set of the group grp in account acct.	Causes an error.
myset_grp_acct	Error (name component longer than eight characters).	The volume set named myset_grp_acct.
m_g_a	The volume set named m_g_a.logongrp.logonacct, provided it exists. If it does not exist, an error is reported.	The volume set name m.g.a.

In MPE V/E, the name V.G.A indicates that V is the name of a volume set, that G is the name of a group, and that A is the name of an account.

MPE/iX accepts that name in that form, but no interpretation is made as to the referencing of G and A. Instead, MPE/iX accepts that name in that form, but no interpretation is made as to the referencing of G and A. MPE/iX treats V.G.A. as a single, long string name, just as it would treat A_VERY_LONG_NAME_FOR_SOMETHING.

MPE/iX does, however, accept the naming convention that was used for MPE V/E private volumes. Therefore, LDISMOUNT V.G.A succeeds, and LDISMOUNT V accesses the same volume set, provided you are logged on to account A, group G. The MPE V/E commands are able to “default“ the logon account and group.

However, VSRESERVE V succeeds only if there is a volume set V in existence. The MPE/iX commands do not call up any default specifications for group and account. VSRESERVE V.G.A succeeds only if a volumeset V.G.A is online. With MPE/iX VSxxxxxx commands, the .G.A component of this name is interpreted as a string, neither more nor less specific than _G _A.

If a volume set is named according to the MPE V/E naming convention (V.G.A), you must use an unambiguous reference when using the MPE/iX volume set commands.

It is recommended that you not use the MPE V/E naming convention and the xxxMOUNT commands. Instead use the MPE/iX naming convention and the VSxxxxxx commands. Alternating between MPE V/E and MPE/iX commands may lead to errors. For example, MOUNT X used in a job stream attempts to access a volume set named X.logongrp.logonacct, which may or may not be your intention.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be executed only from the console unless distributed to users with the ALLOW command.

Examples

To release a volume set named DATABASE.PAYROLL.ACCTNG, enter:

```
LDISMOUNT DATABASE.PAYROLL.ACCTNG
```

You may also use the VSRELEASESYS command:

```
VSRELEASESYS DATABASE.PAYROLL.ACCTNG
```

Related Information

Commands	`MOUNT`, , `LMOUNT`, `DISMOUNT`, `DSTAT`, `VSRESERVE`, `VSRELEASE`
Manuals	*Volume Management Reference Manual*

LIMIT

Limits the number of concurrently running jobs/sessions. (Native Mode)

Syntax

```
LIMIT[ { [+ | - ] numberjobs [+ | - ] ,numbersessions | numberjobs,numbersessions ]
     [;JOBQ=queuename]
```

Parameters

+ Increment the limit value

- Decrement the limit value

numberjobs The number of jobs.

numbersessions The number of sessions.

<omitted> If you specify no parameter, a message is displayed listing the current limits.

queuename The name of the job queue whose limit is being changed or displayed.

Operation Notes

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 disk in a `START RECOVERY`, the operator defined limits are retained. When any other startup option is used, the values configured by the supervisor take effect.

If you enter one parameter and omit the other, the limit of the omitted parameter remains unchanged.

No new jobs or sessions are 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.

Jobs that belong to individual job queues cannot begin execution while the specific job queue limit is exceeded. Even if a specific job queue limit is not exceeded, the global system job limit must also not be exceeded in order for the job to begin execution.

Non-`HIPRI` jobs can still be introduced when the limit is achieved, but they do not execute.

If you attempt to log on to a non-`HIPRI` session after the limit has been reached, you receive the message:

```
CAN'T INITIATE NEW SESSIONS NOW
```

The specified limits may be exceeded at the time the command is issued. This does not cause jobs or sessions executing at the time to abort. They continue to execute, but no new jobs are allowed to enter the executing state, and no new sessions are initiated.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console unless distributed to users with the `ALLOW` command.

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.S
#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
```

Related Information

Commands	`HELLO`, `JOB, SHOWJOB, LISTJOBQ`
Manuals	*Performing System Operation Tasks*

LINK

Creates an executable program file by merging the relocatable object modules from all the files in its FROM= parameter. Those files may correspond to object files, relocatable files, or a combination of them. It also searches any relocatable libraries mentioned in the RL= parameter list and merges any modules within those libraries that resolve an external reference. (Native Mode)

Syntax

```
LINK[ FROM=file[ ,file...] [ ;TO=destfile] ]
[ ;RL=rlfile[  ,rlfile...]...]
[ ;XL=xlfile[,xlfile...]...]
[ ;CAP=caplist]
[ ;NMSTACK=nmstacksize]
[ ;NMHEAP=nmheapsize]
[ ;UNSAT=unsatname]
[ ;PARMCHECK=checklevel]
[ ;ENTRY=entryname]
[ ;NODEBUG] [ ;MAP] [;SHOW] [ ;SHARE]
[ ;PRIVLEV=priv_level]
[ ;PRI=pri_level]
[ ;MAXPRI=max_pri_level
```

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

file The name of an object file or a relocatable library file. It may be any binary file of type NMOBJ or NMRL. All relocatable objects in the FROM= specified list are merged to form the program file specified by *destfile*. If you omit this parameter, LINK merges the object modules in the file $OLDPASS.

The FROM=, RL=, and XL= parameters allow a series of file names. You may name each file individually, or you may provide an indirect file by preceding that file's name with the caret symbol (^).

destfile The name of the program file (type NMPRG) where LINK places the resulting executable object module. If *destfile* does not exist, LINK creates a new one for you. If *destfile* does exist, it is destroyed and replaced by the object module created by the current link operation.

rlfile The name of a relocatable library file (type NMRL) that resolves an external reference made by an object module in the FROM= file list. LINK searches the relocatable libraries in the RL list in the order in which you list them.

If a module from one library calls a routine in another library and then that routine in turn refers to a module in the first library, you may need to include the first library twice so that `LINK` can resolve this “circular“ reference.

The `FROM=`, `RL=`, and `XL=` parameters allow a series of file names. You may name each file individually, or you may provide an indirect file by preceding that file's name with the caret symbol (`^`).

xlfile The name of an executable library (type `NMXL`). The loader searches every executable library in the `XL` list in an attempt to resolve external references that remain in a program file.

caplist The list of capability attributes to be assigned to the program file. The program runs only if the group and account have matching capabilities. (The system manager or account manager assigns these capabilities to your group and account.) Permissible values are:

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

If you omit this parameter, the BA and IA capabilities are assigned to the program file by default.

nmstacksize The maximum size, in bytes, to which the NM stack may grow. This must be a decimal number. The default is zero, which instructs MPE/iX to assign a system-defined constant as the value of *nmstacksize*.

nmheapsize The maximum size, in bytes, to which the NM heap may grow. This must be a decimal number. The default is -1, which instructs the command to assign a system-defined constant as the value of *nmheapsize*.

unsatname The name of a procedure that the loader substitutes in place of any external reference that cannot be resolved in a program file. If you omit this parameter and any external references remain unresolved, the loader reports a load-time error.

checklevel An integer specifying the maximum level of checking that `LINK` performs in binding external references to procedures. All checking levels that are indicated in external references and procedure definitions are reduced (but never increased) to the specified level. If you omit this parameter, `LINK` sets the value to 3.

Permissible values for *checklevel* are defined in Table 4-4.

If the checking level is restricted (reduced) and reportable type errors are detected, they are reported not as errors but as warnings.

Table 4-4 Checklevel Values

0	No parameter check.
1	Check of the symbol type descriptor.
2	Perform Level 1 checking, then check the number of arguments that the import procedure passed against the minimum and maximum range that were declared in the export procedure.
3	Perform Level 2 checking, then check the type of each argument that was passed.

entryname The name (label) of the point within a program where execution begins. When you omit this parameter, the loader begins execution from the primary program entry point (which corresponds to a program's main procedure or outer block). However, by including the `ENTRY=` option, you may override this default value and begin execution from the specified entry point. If the loader fails to find a symbol that matches the entry point name, it reports a load-time error.

`NODEBUG` Strips all symbolic debugging information from the resulting program file. If you omit this parameter, the file contains debugging information if the source file was compiled with this option.

`MAP` Prints a symbol map to the list file, `LINKLIST`.

`SHOW` Displays the name of each object module as it is being merged into the program file. You may include this option to verify the order in which `LINK` processes each module.

priv_level Determines the privilege level used by the executable program file. This parameter changes the privilege level of all procedures in the symbol and export tables (of the relocatable object file) that were set during compilation.

The values for priv_level are:

0	system level access
1	unused
2	privileged level access
3	user level access

The default is that privilege levels are set during compilation.

pri_level Specifies the execution priority that the program will have at run time. The *pri_level* has to be one of BS, CS, DS, ES, or a number between 100 and 255 inclusive. This value can be overridden by the `PRI=` keyword on the `RUN` command.

max_pri_level Specifies the maximum execution priority that the program can have at run time. The *max_pri_level* has to be one of BS, CS, DS, ES, or a number between 100 and 255 inclusive.

SHARE Specifies that data symbols should be exportable and importable (shared) in the resulting executable library.

Operation Notes

The Link Editor uses $STDINX, $STDIN, and $STDLIST as standard files. The Link Editor reads its commands from $STDINX. For interactive sessions this is the terminal keyboard. For a batch job, it is the job stream file.

You can redirect $STDINX to another file. The file must be an unnumbered ASCII file containing valid HP Link Editor/iX commands. Enter a RUN command with the STDIN option. For example, to use the file SCRIPT as the standard input file, enter the command:

```
RUN LINKEDIT.PUB.SYS;STDIN=SCRIPT
```

If you start the Link Editor using the LINK command, or if you execute it by passing a command in the INFO string of the RUN command, $STDINX is not used. Instead, the single command is executed and the Link Editor terminates.

The Link Editor writes all prompts, error messages, and other information to $STDLIST. During an interactive session, this is your terminal. For a batch job, the output spoolfile is used.

You can use another device for $STDLIST. Use the RUN command with the STDLIST option. Note that when you do this interactively, the command prompts do not appear on the screen. For example, to send the Link Editor output to the printer:

```
FILE LINKOUT;DEV=LP
RUN LINKEDIT.PUB.SYS;STDLIST=*LINKOUT
```

Link Editor listings and maps are sent to the file LINKLIST, not to $STDLIST. The listings and maps sent to LINKLIST are:

- The symbol map produced by the MAP option of the LINK command.
- The listing produced by the LISTPROG command.
- The listing produced by the LISTOBJ command.
- The listing produced by the LISTRL command.
- The listing produced by the MAP option of the ADDXL command.
- The listings produced by the LISTXL command.

LINKLIST output goes to $STDLIST. But you can redirect it to another file or device by using the FILE command. To send the listing of the relocatable library LIBRL to the printer:

```
FILE LINKLIST;DEV=LP
LINKEDIT
LinkEd> LISTRL RL=LIBRL
LinkEd> EXIT
```

Use

This command may be issued from a session, job, or program, but not in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

This command merges the object modules from the OBJCODE and places them into the program EXECPROG. It assigns a program stack of 50,000 bytes and requests LINK to build a map and display the name of each object module as it is being linked.

```
LINK FROM=OBJCODE;TO=EXECPROG;NMSTACK=50000;MAP;SHOW
```

The following command merges the object modules from the OBJCODE into program file EXECPROG and searches the relocatable libraries LINEDRAW and ARCDRAW to resolve external references. The resulting program file can be executed only in batch mode by anyone with user mode access.

```
LINK FROM=OBJCODE;TO=EXECPROG;RL=LINEDRAW,ARCDRAW;CAP=BA
```

To link module A and module MAIN and share data so that the data symbols in the program file myprog can be exported and imported to and from the executable library MYXL, enter:

```
LINK FROM=A,MAIN; TO=MYPROG; SHARE; RL=LIBCSHR.LIB.SYS; XL=MYXL
```

Related Information

Commands	RUN, XEQ, LINKEDIT Utility
Manuals	*HP Link Editor/XL Reference Manual* *HP Link Editor/iX Technical Addendum*

LISTACCT

Displays information about one or more accounts.

Syntax

`LISTACCT`[*acctset*] [,*listfile*] [;PASS] [;FORMAT={ SUMMARY|BRIEF|DETAIL}]

Parameters

acctset The accounts to be listed. The default is all accounts for system managers (SM). For all other users, the default is their logon account. Use the # symbol to specify a single numeric character. Use the ? symbol to specify a single alphanumeric character. Use the @ symbol to specify zero or more alphanumeric characters. By itself, @ represents all the members of a set. Each of these wildcard characters counts toward the eight character limit for group, account, and file names.

listfile The name of the output file. The default is `$STDLIST`, a temporary file that cannot be overwritten by a `BUILD` command. It is automatically specified as a new ASCII file with variable-length records, closed in the temporary domain, and with user-supplied carriage-control characters (`CCTL`), OUT access mode, and EXC (EXCLUSIVE access) option. All other characteristics are the same as they would be with the `FILE` command default specifications.

`PASS` Permits account managers and system managers to see the password.

`FORMAT` Specifies one of several display formats, listed below.

- `SUMMARY` Provides a summary of the account information. If `FORMAT` is not specified, `SUMMARY` is the default.
- `BRIEF` Generates a list of account names only.
- `DETAIL` Displays all information associated with the account.

Operation Notes

This command produces account information in an ASCII format.

Use

This command is available from a session, a job, a program, or in BREAK. Pressing **Break** aborts the execution of this command. System managers (SM) can list any account on the system; account managers (AM) and general users can list only their own account.

Examples

The presence of the password in the following display implies that the user has account manager (AM) capability and this is the user's account, or that the user has system manager (SM) capability and this is not the user's account.

```
LISTACCT HPXLII;PASS

   ...or...
LISTACCT HPXLII;PASS;FORMAT=SUMMARY

********************
ACCOUNT: HPXLII
DISC SPACE: 754115(SECTORS)  PASSWORD: ACCTPASS
CPU TIME : 3330(SECONDS)   LOC ATTR: $00000000
CONNECT TIME: 102(MINUTES)   SECURITY READ  : ANY
DISC LIMIT: UNLIMITED           WRITE : AC
CPU LIMIT: UNLIMITED           APPEND : AC
CONNECT TIME: UNLIMITED          LOCK  : ANY
MAX PRI: 150               EXECUTE : ANY
GRP UFID : $00D0001 $80001050 $00138A20 $00000008 $000001FA
USER UFID : $00D4001 $80001050 $00138C20 $00000008 $000001FB
CAP: AM,AL,GL,DI,CV,UV,LG,CS,ND,SF,IA,BA,PH,DS,MR,PM
```

```
LISTACCT @;FORMAT=BRIEF

ACCOUNT1
ACCOUNT2
BACCT1
POSIX
SYS
```

```
LISTACCT POSIX;FORMAT=DETAIL

********************
ACCOUNT   : POSIX
PASSWORD   : **
GID     : 50
DISC SPACE  : 1163440(SECTORS)
CPU TIME   : 199798(SECONDS)
CONNECT TIME : 1116561(MINUTES)
DISC LIMIT  : UNLIMITED
CPU LIMIT  : UNLIMITED
CONNECT LIMIT: UNLIMITED
MAX PRI   : 150
LOC ATTR   : $00000000
SECURITY   : R:ANY, W:ANY, A:ANY, L:ANY, X:ANY
GRP UFID   : $055A0003 $48C0B6B8 $000066B4 $918008B5 $0077B2D9
USER UFID  : $055A0004 $48C0B6B8 $000066B4 $918008B5 $0077B2DF
CAP     : SM,AM,AL,GL,DI,OP,CV,UV,LG,PS,NA,NM,CS,ND,SF,BA,IA,PM,MR,DS,PH
```

Related Information

Commands	LISTFILE, LISTGROUP, LISTUSER, NEWACCT, PURGEACCT, ALTACCT
Manuals	*Performing System Management Tasks*

LISTDIR (UDC)

The LISTDIR UDC executes the LISTFILE command to list all files that are directories.

System-defined UDCs are not automatically available. Your System Manager must use the SETCATALOG command to make these UDCs available for your use. For example,

```
SETCATALOG HPPXUDC.PUB.SYS;SYSTEM;APPEND
```

Syntax

LISTDIR[[DIR=] *dir_name*] [[FORMAT=] *format_opt*]

Parameters

Refer to the LISTFILE command for a complete explanation of the parameters used with the LISTDIR UDC. The following parameters are supported with the LISTDIR UDC.

dir_name The name of the directory to list. The *dir_name* can be in MPE or HFS syntax; wildcards may be used. For example, /SYS/PUB, /SYS/PUB/dir@, ./abc/mydir, and @abc are valid examples of directory names. If *dir_name* is not specified, the default directory name is ./@ (all directories directly under your current working directory).

format_opt An output format option. The option may be specified as a number or mnemonic. For example,

```
FORMAT=2
```

or

```
FORMAT=DISC
```

If not specified, the default is FORMAT=6 (qualify).

Refer to the LISTFILE command for a complete description of each available format option.

Operation Notes

The LISTDIR UDC lists all files that are directories. The UDC executes the following form of the LISTFILE command:

```
LISTFILE dir_name ;FORMAT=format_opt ;SELEQ=[OBJECT=DIR] ;TREE
```

Use

This UDC may be issued from a session, a job, a program, or in BREAK. Pressing **Break** aborts execution.

Examples

Refer to the LISTFILE command later in this chapter for examples.

Related Information

Commands	`LISTFILE`, `FINDDIR` (UDC)
Manuals	None

LISTEQ

Displays all active file equations for a job or session.

Syntax

`LISTEQ`[*listfile*]

Parameters

listfile The name of the output file. The default is `$STDLIST`, a temporary file that cannot be overwritten by a `BUILD` command. It is automatically specified as a new ASCII file with variable-length records, closed in the temporary domain, and with user-supplied carriage-control characters (`CCTL`), `OUT` access mode, and `EXC` (EXCLUSIVE access) option. All other characteristics are the same as they would be with the `FILE` command default specifications.

Operation Notes

The `LISTEQ` command displays all the active file equations for a job or session.

Use

This command may be issued from a session, a job, a program, or in BREAK. Pressing **Break** aborts the execution of this command.

Example

An example of `LISTEQ` is given below:

```
LISTEQ

FILE EQUATIONS

FILE TAPE1;DEV=ATAPE
FILE PP;ENV=LP2.ENV.OSE;DEV=EPOC
FILE MYFILE,NEW;REC=-80,3,F,ASCII;DISC=5000;SAVE

FILE POSIX=./mydir/myfile1
```

Related Information

Commands	`FILE`, `RESET`
Manuals	None

LISTF

Displays information about one or more permanent files. (CM)

Syntax

LISTF[fileset] [,listlevel] [;listfile]

Parameters

fileset Specifies the set of files to be listed. The default is @, which lists all files in your logon group. You may select the file(s) to be listed by using the fully or partly qualified form for fileset:

filename.groupname.accountname

You may use the @ to specify zero or more alphanumeric characters or, if used by itself, to denote all the members of a set. You may use the symbol # to specify one numeric character and the symbol ? to specify one alphanumeric character. The # and ? wildcard characters count toward the eight character limit for group, account, and file names.

listlevel Specifies the amount and format of information to display for the file(s) you select. The default is 0, which displays only the file name. The *listlevel* of the LISTF command is equivalent to the format option of the LISTFILE command. The levels are described below in Table 4-5:

Table 4-5 Format Options

Listlevel	Displayed Information
–2	Displays the file's ACD (access control definition). System Managers can view the ACD for any file. Account Managers can view the ACD for files in that account. File creators can view the ACD for their files. Other users can view an ACD only if that ACD specifies that the user has RACD (read ACD) access.
–1	Shows only the file label in hexadecimal. The hexadecimal display generated by this format option only serves a diagnostic purpose in MPE/iX and is subject to change.
0	For each directory, this option displays PATH=The name of the file is displayed in a multicolumn format. This is the default.
1	Displays the file name, file code, record size, record format, and other file characteristics such as ASCII or binary records, carriage-control option, file type, current end-of-file location, and the maximum number of records allowed in the file.

Listlevel	Displayed Information
2	Displays the file name, file code, record size, file type, current end-of-file location, and the maximum number of records allowed in the file. It also displays the blocking factor, number of sectors in use, number of extents currently allocated, and the maximum number of extents allowed.
3 –3	Displays the file name, record size, extent size, number of records, user's access rights, and other file characteristics including the date created, modified, and last accessed. The same information for MPE and HFS files is displayed except for the following differences: • Fully qualified MPE file name is replaced by an absolute pathname. • Creator field displays the fully qualified user ID of the file owner. • For MPE groups, the SECURITY field displays `SAVE`; for entries other than MPE groups it is blank. All file access matrix fields are blank for anything other than MPE accounts, MPE groups, and files in an MPE group. • The LOCKWORD field is omitted. The creator, group id, and label address are omitted in `FORMAT=3`. These can be obtained by specifying -3 if you have sufficient capability (AM or SM)
4	Displays the security matrix for the file. This includes account, group and file-level security, and the access rights for the user. For MPE groups and MPE accounts, the security matrix for group, account, and account-only are displayed. The rest of the fields of the file access matrix are blank. For HFS directories, and files within HFS directories, all the fields of the file access matrix are blank. In addition, `LISTFILE` displays the message `ACD EXISTS.`
5 –5	Shows `LISTFILE,3` data and all file-specific data in `LISTFILE,3` type format (KSAM, SPOOL, and symbolic links). If a file has no unique data, only the option 3 data is shown.
6	Shows the absolute pathname of the file.

Listlevel	Displayed Information
7	Shows all file specific data in `LISTFILE,5` type format, but does not show `LISTFILE,3` data. If a file has no unique data, only the file name is displayed.
8	Shows all accessors of the files listed. Restrictions apply
9	Shows level 8 information and details about processes accessing the files including file locking data. Restrictions apply.

listfile The name of the output file to which the file information will be written. If you omit this parameter, the output appears on $STDLIST. If you specify *listfile*, the output is sent to a temporary file created for this purpose. The temporary file is a new ASCII file with variable length records, closed in the temporary domain, and with user supplied carriage control characters (CCTL), `OUT` access mode, and `EXC` (exclusive access) option. All other characteristics are identical to the `FILE` command default specifications. You may specify a different kind of file or backreference an existing file.

When you direct `LISTF` output to $STDLIST from a job, or when you direct the output to any non-disk device, a date and time stamp preceeds the data, and *listlevel* 0 data appears as one file per record rather than in the standard multi-column format.

Operation

The `LISTF` command displays a description of the file(s) you specified in *fileset*. It only accepts MPE file name syntax, but it displays information in one of two formats, MPE or POSIX, depending upon whether or not your current group differs from your logon group. MPE format examples appear below. For examples of the POSIX format, see the `LISTFILE` command.

You may list any file, but there are restrictions on the kinds of information available to various users. A standard user may specify a *listlevel* of 0, 1, 2, 3, 4, 5, 6 or 7. If you have account manager capability (AM), you may request *listlevel* -1, -3 or -5, 8, 9 information about files in your own account. If you have System Manager capability (SM), you can specify any *listlevel* to view all information for all files on the system. List levels 8 and 9 are also available if you are the owner of the files.

For list levels 8 and 9 the IP address of remote accessors and the program name of the accessor process are restricted fields. PM, SM, OP, NA, or NM capabilities are needed to see the IP address. The rules defined by the **`SHOWPROC`** command are enforced before revealing the process name.

The `LISTF` command does not display `#SEG`, `STACK`, `MAXDATA`, `TOTAL`, `DB`, `DL` or `CAP` values for program files. That information is displayed by the `VERSION` utility. For more information, see the `VERSION` command.

You may have the information displayed on a device other than the standard listing device. To do that, you will need to name the device with a FILE command and then backreference the file in the LISTF command. For example:

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

Use

The LISTF command is available from a session, job, or a program, or in BREAK. Pressing **Break** aborts the execution of this command.

Examples

Level 0 File Display

```
:LISTF

FILENAME
CLKLIST CLOCK  EDIRC  LINKCLK  LINKFROG  LINKLIST
```

Level 1 File Display

```
: LISTF L@,1

ACCOUNT= HPXLII     GROUP=  DEVELOP
FILENAME CODE  -LOGICAL RECORD

           SIZE   TYP    EOF  LIMIT
L2          80B   FA     2     12
LINKCLK     72B   FA     1     11
LINKFROG    72B   FA     1     11
LINKLIST    72B   FA     8     18
```

Level 2 File Display

```
:LISTF L@,2

ACCOUNT= HPXLII     GROUP=  DEVELOP
FILENAME CODE  -LOGICAL RECORD- SPACE

            SIZE TYP   EOF   LIMIT R/B SECTORS #X MX
LINKCLK      72B  FA    1      11  3      8     1 1
LINKFROG     72B  FA    1      11  3      8     1 1
LINKLIST     72B  FA    8      18  3      8     1 1
```

Level 3 File Display

```
:LISTF DOCMNTS,3

********************
FILE DOCMNTS.DEVELOP.HPXLII
FCODE 0        FOPTIONS STD,ASCII,FIXED,NOCCTL
BLK FACTOR 16     CREATOR **
REC SIZE 80(BYTES)  LOCKWORD **
BLK SIZE 640(BYTES)  SECURITYREAD  : ANY
EXT SIZE 25(SECT)        WRITE : ANY
NUM REC 501           APPEND : ANY
NUM SEC 165           LOCK  : ANY
NUM EXT 7            EXECUTE: ANY
MAX REC 501     **SECURITY IS ON
MAX EXT 7      FLAGS n/a
NUM LABELS 0     CREATED FRI, 21 SEP 1986, 11:55 AM
MAX LABELS 0     MODIFIED FRI, 21 SEP 1986, 12:34 PM
```

```
DISC DEV # 3     ACCESSED FRI, 21 SEP 1986, 12:46 PM
CLASS   DISC    LABEL ADDR **
SEC OFFSET 0

VOLSET        MPEXL_SYSTEM_VOLUME_SET
or
VOLNAME       MPEXL_SYSTEM_VOLUME_SET: MEMBER1

or
VOLCLASS       MPEXL_SYSTEM_VOLUME_SET: DISC
CLASS  : DISC    LABEL ADDR: $00000010 $0010E014
```

Level 6 File Display

```
:LISTF L@,6

LINKCLK.DEVELOP.HPXLII
LINKFROG.DEVELOP.HLPXLII
LINKLIST.DEVELOP.HPXLII
```

Level 7 File Display

```
********************
FILE:  LINKCLK.DEVELOP.HPXLII
********************
FILE:  LINKFROG.DEVELOP.HLPXLII
********************
FILE:  LINKLIST.DEVELOP.HPXLII
```

Level 8 File Display

```
:listfile hppxudc.pub.sys,8
********************
FILE: HPPXUDC.PUB.SYS
15 Accessors(O:15,P:15,L:0,W:0,R:15),Share
#S265   MIKEP.HPE                P:2,L:0,W:0,R:2      LDEV: 49
#S263   JEFFV,MGR.JVNM           P:3,L:0,W:0,R:3      LDEV: 47
#S261   KROGERS.MPENT            P:2,L:0,W:0,R:2      LDEV: 50
#S231   SUSANC.MPENT             P:2,L:0,W:0,R:2      LDEV: 46
#S219   FAIRCHLD.MPENT           P:2,L:0,W:0,R:2      LDEV: 39
#S214   CATHY,MGR.BOSS           P:2,L:0,W:0,R:2      REM : 15.14.16.198
#J434   FTPMON,FTP.SYS           P:2,L:0,W:0,R:2      SPID: #O21905
```

Level 9 File Display

```
:listfile hppxudc.pub.sys,9
********************
FILE: HPPXUDC.PUB.SYS
5 Accessors(O:5,P:5,L:5,W:0,R:5),Share
#S263   JEFFV,MGR.JVNM             P:3,L:3,W:0,R:3      LDEV: 47
 #P147   (LFCI.PUB.SYS)
   ACCESS: R-excl           REC#: 0                     FNUM: 13
    LOCKSOwner   Waiter
         FLOCK
         OPEN
 #P154   (CI.PUB.SYS)
   ACCESS: R-excl           REC#: 0                     FNUM: 13
    LOCKS: none
 #P86    (JSMAIN.PUB.SYS)
   ACCESS: R-excl           REC#: 336                   FNUM: 16
    LOCKSOwner   Waiter
                     FLOCK

#J434   FTPMON,FTP.SYS             P:2,L:2,W:0,R:2      SPID: #O21905
 #P79    (CI.PUB.SYS)
   ACCESS: R-excl           REC#: 0                     FNUM: 14
    LOCKS: none
 #P47    (JSMAIN.PUB.SYS)
```

```
   ACCESS: R-excl            REC#: 336                  FNUM: 15
    LOCKSOwner   Waiter
         OPEN        FLOCK
```

Level -2 File Display

```
FILENAME    ACD ENTRIES

DOCMNTS       NO ACDS
```

Level -3 File Display

```
:LISTF DOCMNTS,-3

********************
FILE DOCMNTS.DEVELOP.HPXLII
FCODE O        FOPTIOc
NS STD,ASCII,FIXED,NOCCTL       15496000
BLK FACTOR 16     CREATOR PETE
REC SIZE 80(BYTES)  LOCKWORD RETEP
BLK SIZE 640(BYTES)  SECURITYREAD  : ANY
EXT SIZE 25(SECT)        WRITE : ANY
NUM REC 501           APPEND : ANY
NUM SEC 165           LOCK  : ANY
NUM EXT 7            EXECUTE: ANY
MAX REC 501     **SECURITY IS ON
MAX EXT 7      FLAGS n/a
NUM LABELS 0     CREATED FRI, 21 SEP 1986, 1155 AM
MAX LABELS 0     MODIFIED FRI, 21 SEP 1986, 1234 PM
DISC DEV # 3     ACCESSED FRI, 21 SEP 1986, 12:46 PM
CLASS   DISC    LABEL ADDR $00000010 $0010E014
SEC OFFSET 0
```

Level 4 File Display

```
:LISTF DOCMNTS,4

********************
FILE DOCMNTS.DEVELOP.HPXLII
SYSTEM  READ  : ANY
SECURITYWRITE  : AC
(ACCT)  APPEND  : AC
LOCK       : ANY
EXECUTE      : ANY
SYSTEM  READ  : GU
SECURITYWRITE  : GU
(GROUP) APPEND  : GU
LOCK       : GU
EXECUTE      : GU
SAVE       : GU
SECURITYREAD  : ANY     FCODE 0
(FILE)  WRITE  : ANY     CREATOR PETE
APPEND      : ANY     LOCKWORD
LOCK       : ANY    **SECURITY IS ON
EXECUTE      : ANY
FOR PETE.HPXLII READ,WRITE,APPEND,LOCK,EXECUTE
```

Level -1 File Display

```
:LISTF LINKCLK,-1

F = LINKCLK
00000001 44495343 20202020 20202020 20202020 20202020 .........@..LINK
20202020 20202020 20202020 20310000 4C495354 53202020 DEVELOP    ...
20202020 20202020 44455645 4C4F5020 20202020 20202020 HPXLII   .
00000000 4850584C 49492020 20202020 20202020 00000000 ...PETE
52455445 50202020 20202020 20202020 50455445 20202020 .|..,2....#.,7.6
20202020 20202020 00000000 FC000000 04580001 13915EF4 ,2.|..#.,2.....#
```

```
    00010405 00000000 00000300 00020CEE 0EA78B32 00020CEE .......H........
    0EA78B32 00020CEE 12F61E2D 00020CEE 0EA78B32 00000000 ................
    000000A0 000001F5 00000000 00000000 00000000 00000000 ...........  ..
    00009C90 00000000 00000000 00000000 00000050 00000500 ................
    00100000 00190007 000F0000 20200000          C.8x@.R.@.Q.......
```

Additional Information

Commands LISTFILE, VERSION, CHDIR , LISTDIR (UDC) FINDFILE (UDC)

Manuals *Performing System Management Tasks*

Performing System Operation Tasks

5 Command Definitions L-O

LISTFILE

This command lists file and directory attributes through the use of options. The `LISTFILE` information is a superset of the `LISTF` command information.

Syntax

```
LISTFILE[ [ fileset=] {   fileset   (fileset[ ,fileset] ...) } ]

              [ [ ;FORMAT=] format_opt]
              [ [ ;SELEQ=] select_eq  | ^indir]
              [ [ ;NAME=] pattern]
              [ ;PASS]
              [ ;{ PERM} { ;TEMP} [ ;PERMTEMP] ]
               [ ;USENAME] [ ;TREE] [ ;NOTREE]
```

Parameters

fileset

Specifies the set of files to be listed. The default for *fileset* is @, meaning all MPE-named files in your current working directory (CWD). If *fileset* includes more than one file, be sure to separate the file names with commas and enclose the set in parentheses, for example:

```
:listfile (test1,test2,test3)
```

The files named in the *fileset* parameter can be either in MPE or HFS syntax (explained below). The file names dot (.) and dot-dot (..) have special meaning, that is, current directory and the parent of the current directory, respectively.

Using Wildcards

You may use wildcard characters in any position in the file name. You may use the - character as a wildcard in any position *except as the first character of the file name*. These wildcards have the following meaning:

@ matches zero or more of any character

? matches one character

# matches one digit

[ ] matches one character specified between the brackets

- if used within brackets ([ ]), the hyphen (-) means a range of characters. For example, "[c - g]" means all the characters between `c` and `g` inclusive. The character on the left must alphabetically precede the character on the right.

- If used immediately after the left bracket ([), or just before the right bracket (]), hyphen (-) means the character `-' itself.

For example, "[a-c]" means one of ‘a', ‘b', or ‘c', whereas "[-a-c]" or "[a-c-]" means one of ‘a', ‘b', ‘c', or ‘-'.

It is illegal to specify [c-a], or [a-A] because ‘c' does not alphabetically precede ‘a' and uppercase ‘A' comes before lowercase ‘a' (in ASCII character evaluation). Also note that it is legal to specify [A-z] and any legal special characters.

MPE Syntax

If *fileset* does not begin with the dot or slash (indicating HFS syntax), it is parsed according to MPE syntax and has the following form:

filename[.*groupname*[.*accountname*]]

A LISTFILE command using MPE syntax does not display files that do not follow the traditional MPE naming conventions of up to eight character names for files, groups and accounts.

If the *fileset* parameter does not specify *groupname*, all the files (with uppercase names that have up to 8 alphanumeric characters) in the current working directory (CWD) are listed irrespective of whether CWD is an MPE group or not. For example, the following command lists the files in all of the groups of the logon account:

```
LISTFILE @.@
```

In contrast, the next command lists all the files in the CWD (which may be different from the logon group). However, only those files whose names are valid MPE names are displayed.

```
LISTFILE @
```

If the CWD is not an MPE group, the information about the file is displayed in an HFS format discussed below.

You may have an MPE group that also contains files with HFS syntax, for example, they are lowercase, have long names, or contain special characters. To see both MPE and HFS files in a group, type,

```
LISTFILE ./@
```

HFS Syntax

If the *fileset* begins with a dot (.) or a slash (/), it is assumed to be in HFS syntax. The characters composing the name may be selected from the following set:

```
a-z

A-Z

0 1 2 3 4 5 6 7 8 9 - _ . ‘ ~ $ % ^ * + \ { } :
```

If the *fileset* parameter begins with a slash (/), the pathname is assumed to be an absolute pathname; otherwise, it is considered to be CWD relative.

If *fileset* ends in a slash, it is treated as a directory name, and *pattern* is used to determine the file names that match. All the directories and files that match *fileset* are found, and searched recursively to display the files and directories that match *pattern*. For example, if *fileset* is `/SYS/@/`, all files and subdirectories within `SYS`, and all files and directories within those subdirectories are displayed. The default for *pattern* is `@`.

If *fileset* does not end in a slash, all of the files that match *fileset* are displayed. For example, if *fileset* is `/SYS/@`, you will see a list of all files, subdirectories and groups in the `SYS` directory, but not any files or subdirectories within those directories.

If you have specified `TREE`, a trailing slash is assumed at the end of the *fileset*. For example, the command `LISTFILE /SYS/@;TREE` behaves like `LISTFILE /SYS/@/`. On the other hand, if you specify `NOTREE`, the trailing slash, if present at the end of a fileset, is ignored. Hence, the command `LISTFILE /SYS/@/;NOTREE` behaves like `LISTFILE /SYS/@`.

format_opt A format selection. This parameter has no effect on the files selected for display, but affects the selection of information about the files that you see. If *fileset* begins with a dot (.) or slash (/), or if the CWD is different from your current MPE group, or if you specify the `;TREE` option, then you will see the HFS output style. This, in part, means that:

- Account, group, and directory names will end in a slash (/).
- File names will appear at the end of the output lines.
- Output begins in column two so that you can more easily detect filename wraparound from the previous line (which, if wrapping occurs, will begin in column one).

The following Table 5-1 displays the format options available.

Table 5-1 Format Options

Opti on	Name	Displayed Information
–2	ACD	Displays the file's ACD (access control definition). System Managers can view the ACD for any file. Account Managers can view the ACD for files in that account. File creators can view the ACD for their files. Other users can view an ACD only if that ACD specifies that the user has RACD (read ACD) access.
–1	LABEL	Shows only the file label in hexadecimal. The hexadecimal display generated by this format option only serves a diagnostic purpose in MPE/iX and is subject to change.
0	FILES	For each directory, this option displays `PATH=`The name of the file is displayed in a multicolumn format. This is the default.

Option	Name	Displayed Information
1	SUMMARY	Displays the file name, file code, record size, record format, and other file characteristics such as ASCII or binary records, carriage-control option, file type, current end-of-file location, and the maximum number of records allowed in the file.
2	DISC	Displays the file name, file code, record size, file type, current end-of-file location, and the maximum number of records allowed in the file. It also displays the blocking factor, number of sectors in use, number of extents currently allocated, and the maximum number of extents allowed.
3 –3	DETAIL DETAIL;PASS	Displays the file name, record size, extent size, number of records, user's access rights, and other file characteristics including the date created, modified, and last accessed. The same information for MPE and HFS files is displayed except for the following differences: • Fully qualified MPE file name is replaced by an absolute pathname. • Creator field displays the fully qualified user ID (user.acct) of the file owner. • For MPE groups, the SECURITY field displays SAVE; for entries other than MPE groups it is blank. All file access matrix fields are blank for anything other than MPE accounts, MPE groups, and files in an MPE group. • The LOCKWORD field is omitted. The creator, group id, and label address are omitted in FORMAT=3. These can be obtained by specifying -3 if you have sufficient capability (AM or SM)
4	SECURITY	Displays the security matrix for the file. This includes account, group and file-level security, and the access rights for the user. For MPE groups and MPE accounts, the security matrix for group, account, and account-only are displayed. The rest of the fields of the file access matrix are blank. For HFS directories, and files within HFS directories, all the fields of the file access matrix are blank. In addition, LISTFILE displays the message ACD EXISTS.
5 –5	DATA DATA;PASS	Shows LISTFILE,3 data and all file-specific data in LISTFILE,3 type format (KSAM, SPOOL, and symbolic links). If a file has no unique data, only the option 3 data is shown.
6	QUALIFY	Shows the absolute pathname of the file.

Opti on	Name	Displayed Information
7	UNIQUE	Shows all file specific data in LISTFILE,5 type format, but does not show LISTFILE,3 data. If a file has no unique data, only the file name is displayed. Default = 0 (FILES).
8	ACCESS	Shows all accessors of the files listed. Restrictions apply.
9	LOCKS	Shows level 8 information and details about processes accessing the files including file locking data. Restrictions apply.

select_eq A selection equation. Use the selection equation as a filter on *fileset*. From the set of files matching the fileset, only files that match the *select_eq* requirements are listed. You may select file types by using the FTYPE option, or you may select object types by using the OBJECT option. Selection equations have the following format:

```
[FTYPE = KSAMXL | SPOOL]
[OBJECT = ACCT | GROUP | FILE | DIR | HFSDIR | SYMLINK]
[CODE = number | |mnemonic | |PRIV
[ACCESS = INUSE | OPEN | LOCK| EXCL]
```

You must enclose selection equations in square brackets. For example:

```
LISTFILE ./@ ;SELEQ=[OBJECT=DIR]
```

You can also use your text editor to make a file that contains the OBJECT or FTYPE statement, for example [OBJECT=DIR], and save it with a filename. Thereafter, you can select this file by entering the following command:

```
LISTFILE ./@ ;SELEQ=^FILENAME
```

The OBJECT option applies to HFS files, and may have any one of the following values.

ACCT	Lists only the MPE ACCOUNT directory. You may use ACCTS, ACCOUNT, ACCOUNTS as synonyms for ACCOUNT.
GROUP	Lists only the MPE GROUP directory. You may use GROUPS as a synonym for GROUP.
FILE	Lists only the files and not directories/groups/accounts. You may use FILES as a synonym for FILE.
DIR	Lists only directories (including groups/accounts and the system root directory /). You may use DIRS, DIRECTORY, or DIRECTORIES as synonyms for DIRECTORY.
HFSDIR	Lists only directories other than root, accounts, and groups.
SYMLINK	Lists onlyfiles that are symbolic links.
NUMBER	List only files matching the specified file code number.

MNEMONIC List only files matching the specified file code mnemonic

PRIV List only files with negative file code.

INUSE Lists only files that are currently in use by users or by MPE.

OPEN Lists only files that are opened by progams. INUSE is a superset of OPEN.

LOCK List only files being locked by a program.

EXCL List only files being closed exclusively

pattern When POSIX syntax is used in the fileset, *pattern* is exactly the same as the *filename* components of *fileset* as previously described. The name parameter applies only to HFS syntax.

The LISTFILE command displays only those file names which match the *pattern*. For example,

```
LISTFILE /SYS/;NAME=OFF@
```

displays all the files/groups/directories under the SYS account that start with OFF, off, Off, and so on.

If *pattern* is specified within single or double quotes, it is case sensitive. For example,

```
LISTFILE /SYS/;NAME=`OFF@'
```

displays all the files/groups/directories under the SYS account that start with OFF. It will not display names that start with off, Off, and so on. The default for the *pattern* parameter is @; that is, it matches all names without regard to case.

NOTE You cannot use the NAME parameter for an MPE *fileset* because *pattern* can be specified as the part of the *fileset*. So, for example, instead of entering the command LISTFILE @.@.@;NAME=@DOC, enter the command LISTFILE @DOC.@.@. instead.

PASS The PASS option displays sensitive data. Using it depends on your access rights to the data; that is if you are the owner or have AM or SM capability.

PERM The PERM option displays permanent files only. “PERM" is the default.

TEMP The TEMP option displays temporary files only.

PERMTEMP The PERMTEMP option displays both permanent and temporary files. The permanent files are listed before the temporary files.

USENAME The USENAME option applies only to HFS-named filesets. This option indicates that the name is to be used to determine how many levels to display. If the fileset ends in a slash (/), then all files at all levels below the target file are displayed. If the name does not end in a slash (/), then only the files at the specified level are displayed. For example, `/@/@/@` indicates that all objects at the third level are to be displayed. `USENAME` is the default.

`TREE` If the `TREE` option is specified, objects at all lower directory levels are displayed.

`NOTREE` Indicates that only objects at the specified level are to be displayed. The `NOTREE` option overrides an HFS fileset that ends in a slash.

Operation Notes

You can use `LISTFILE` to list descriptions of one or more disk files at the level of detail you select. You must have traverse directory entries (TD) and/or read directory entries (RD) access for the directories in the pathname of the files that will be displayed by `LISTFILE`. (Refer to the `ALTSEC` command for further information on directory permissions.)

For example, if the *fileset* is `/dir1/dir@/@`, you must have TD access for the root directory (/) and dir1. Also, you must have RD access for `dir1` since the next name is wildcarded (`dir@`) and have RD access to each directory within the path specified by `/dir1/dir@` since the next (and final) name is a wildcard (@).

You may list any file, but there are restrictions on the kinds of information available to various users. A standard user may specify a *listlevel* of 0, 1, 2, 3, 4, 5, 6 or 7. If you have account manager capability (AM), you may request *listlevel* -1, -3 or -5, 8, 9 information about files in your own account. If you have System Manager capability (SM), you can specify any *listlevel* to view all information for all files on the system. List levels 8 and 9 are also available if you are the owner of the files. A file description is not listed unless the file's home volume set (PV) is mounted.

Use

This command may be issued from a session, a job, a program, or in BREAK. Pressing **Break** aborts execution.

If the *fileset* is in MPE syntax, `LISTFILE` only displays file names that follow MPE naming syntax. For example, `LISTFILE @,2` will not display the file `am_pm`, whereas `LISTFILE ./@,2` will display the file.

If *fileset* ends in a slash (/) or the `;TREE` option, then the contents of every matching directory will be displayed recursively. To see just a directory name, but not all the files under it, use the `;NOTREE` option or omit the trailing slash.

MPE Examples

```
LISTFILE @

 FILENAME

 FILE1

LISTFILE @.PUB.OFFICE,2

ACCOUNT= OFFICE  GROUP= PUB
```

```
FILENAME CODE LOGICAL RECORD- SPACE
         SIZE TYP    EOF   LIMIT R/B SECTORS #X MX
F4         80B AF     411    411 16   144 2 *
F5         80B AF     199    199 16    64 1 *
```

HFS Examples

The following figure illustrates a hierarchical directory structure. In this figure, directory names are shown as the character d plus a number (for example, d0), and file names are shown as the character f plus a number (for example, f1). The examples assume the directory structure shown. They also assume that the current working directory (CWD) is /ACCT/GROUP/d0.

Example File System

```
               /ACCT/GROUP/d0 = CWD
                   |
 -|-
 |        |            |            |  |  |
 d1       d2            d3           f1 f2 f3
         |            |
  |-   |
  |   |  |  |    |   |   |   |  |  |  |
  d4  f4 f5  d5   d6  f6  d7  f7 f8 f9 f10
  |       |    |      |
 -|     -| -|   |-
 |  |     |   |  |  |   |  |  |  |  |  |
 f11 f12    d8  f13 f14 f15  d9  f16 f17 f18 f19 f20
```

The first example below sets the `HPPROMPT" variable to show the current working directory, changes the CWD to d0, and produces a listing of all files one level below the CWD.

```
:hello manager.acct,group

:setvar hpprompt "hpcwd:"
/ACCT/GROUP:chdir ./d0
CWD is "/ACCT/GROUP/d0".
/ACCT/GROUP/d0:listfile ./@

 PATH= /ACCT/GROUP/d0/

 d1/ d2/ d3/ f1 f2 f3
```

The next example produces a listing of all files one level below the CWD using FORMAT=2 (DISC) option.

```
/ACCT/GROUP/d0:listfile ./@,2

 PATH= /ACCT/GROUP/d0/./

 CODE LOGICAL RECORD- SPACE FILENAME
     SIZE TYP    EOF   LIMIT R/B SECTORS #X MX
     16W HBD     4  67107839  1    64 2 * d1/
     16W HBD     4  67107839  1    64 2 * d2/
     16W HBD     4  67107839  1    64 2 * d3/
     80B AF     12      12  1    16 1 1 f1
     80B AF     12      12  1    16 1 1 f2
     80B AF     12      12  1    16 1 1 f3
```

In the next example, specifying the absolute pathname produces a listing of all entries one level below the group.

```
/ACCT/GROUP/d0:listfile /ACCT/GROUP/@,2

 PATH= /ACCT/GROUP/

 CODE LOGICAL RECORD- SPACE FILENAME
     SIZE TYP    EOF   LIMIT R/B SECTORS #X MX
     16W HBD     4  67107839  1    64 2 * *d0/
```

In the next example, specifying the NAME parameter produces a listing of all entries with names beginning with a lower case "d". Using the FORMAT=6 (QUALIFY) option shows the absolute pathname of all HFS entries.

```
/ACCT/GROUP/d0:listfile /;name=`d@';format=6

 /ACCT/GROUP/d0/
 /ACCT/GROUP/d0/d1/
 /ACCT/GROUP/d0/d2/
 /ACCT/GROUP/d0/d2/d4/
 /ACCT/GROUP/d0/d2/d5/
 /ACCT/GROUP/d0/d2/d5/d8/
 /ACCT/GROUP/d0/d2/d6/
 /ACCT/GROUP/d0/d3/
 /ACCT/GROUP/d0/d3/d7/
 /ACCT/GROUP/d0/d3/d7/d9/
```

The next example illustrates the use of the OBJECT=ACCT parameter to show all accounts on the system.

```
/ACCT/GROUP/d0:listfile /@,6; seleq=[object=acct]

 /ACCT/
 /SYS/
 /TELESUP/
 /TEST/

   .
   .
   .
```

The next example illustrates the OBJECT=GROUP parameter to show all groups on the system.

```
/ACCT/GROUP/d0:listfile /@/@;seleq=[object=group];format=qualify

 /ACCT/GROUP/
 /ACCT/PUB/
 /SYS/ALINE925/

   .
   .
   .
 /TELESUP/PUB/
 /TEST/PUB/
 /TEST/SPOOL/
 /TEST/SPOOLSTD/
 /TEST/TEMPLATE/

/ACCT/GROUP/d0:
```

The next example illustrates the use of the OBJECT=DIR parameter to show all directories on the system. This is similar to the FINDDIR UDC.

```
/ACCT/GROUP/d0:listfile /, qualify;seleq=[object=dir];format=qualify
 /
 /ACCT/
 /ACCT/GROUP/
 /ACCT/GROUP/d0/
 /ACCT/GROUP/d0/d1/
 /ACCT/GROUP/d0/d2/
 /ACCT/GROUP/d0/d2/d4/
 /ACCT/GROUP/d0/d2/d5/
 /ACCT/GROUP/d0/d2/d5/d8/
 /ACCT/GROUP/d0/d2/d6/
 /ACCT/GROUP/d0/d3/
 /ACCT/GROUP/d0/d3/d7/
 /ACCT/GROUP/d0/d3/d7/d9/
 /ACCT/PUB/
 /SYS/
 /SYS/ALINE925/
 /SYS/ALINK925/

   .
   .
   .
 /TELESUP/PUB/
 /TEST/PUB/
 /TEST/SPOOL/
 /TEST/SPOOLSTD/
 /TEST/TEMPLATE/

   .
   .
   .
```

The next example illustrates a summary listing (format option 1) of all files in subdirectory d3.

```
/ACCT/GROUP/d0:listfile d3/@,1

 PATH= /ACCT/GROUP/d0/./d3/

 CODE LOGICAL RECORD- FILENAME
     SIZE TYP    EOF   LIMIT
     16W DBH     4  67107839 d7/
     80B AF     12     12 f10
     80B AF     12     12 f7
     80B AF     12     12 f8
     80B AF     12     12 f9
```

The next example illustrates a detail listing (format option 3) of all files in subdirectory d3.

```
/ACCT/GROUP/d0:listfile ./d3/@,3
 ********************
 FILE: /ACCT/GROUP/d0/d3/d7/

 FILE CODE : 0          FOPTIONS: DIRECTORY
 BLK FACTOR: 1          OWNER  : **
 REC SIZE: 32(BYTES)       GROUP ID: **
 BLK SIZE: 32(BYTES)       SECURITYREAD  :
 EXT SIZE: 0(SECT)             WRITE  :
 NUM REC: 4                APPEND :
 NUM SEC: 64                LOCK  :
```

```
 NUM EXT: 2                EXECUTE :
 MAX REC: 67107839            **SECURITY IS ON
                 FLAGS  : NO ACCESSORS
 NUM LABELS: 0          CREATED : TUE, JUL 21, 1992, 2:20 PM
 MAX LABELS: 0          MODIFIED: TUE, JUL 21, 1992, 2:23 PM
 DISC DEV #: 1          ACCESSED: WED, JUL 22, 1992, 12:05 PM
 SEC OFFSET: 0          LABEL ADDR: **
 VOLCLASS : MPEXL_SYSTEM_VOLUME_SET:DISC
 ********************
.
.
.
 FILE: /ACCT/GROUP/d0/d3/f9

 FILE CODE : 0          FOPTIONS: ASCII,FIXED,NOCCTL,STD
 BLK FACTOR: 1          OWNER  : **
 REC SIZE: 80(BYTES)       GROUP ID: **
 BLK SIZE: 80(BYTES)       SECURITYREAD  :
 EXT SIZE: 13(SECT)            WRITE  :
 NUM REC: 12                APPEND :
 NUM SEC: 16                LOCK  :
 NUM EXT: 1                EXECUTE :
 MAX REC: 12               **SECURITY IS ON
 MAX EXT: 1           FLAGS  : NO ACCESSORS
 NUM LABELS: 0          CREATED : TUE, JUL 21, 1992, 2:21 PM
 MAX LABELS: 0          MODIFIED: TUE, JUL 21, 1992, 2:21 PM
 DISC DEV #: 2          ACCESSED: TUE, JUL 21, 1992, 2:21 PM
 SEC OFFSET: 0          LABEL ADDR: **
 VOLCLASS : MPEXL_SYSTEM_VOLUME_SET:DISC
/ACCT/GROUP/d0:
```

The next example illustrates the use of the `FORMAT=-3` option to show the owner. You must be the owner, or have AM or SM capability to use this option.

```
/ACCT/GROUP/d0:listfile /ACCT/GROUP/@,-3
 ********************
 FILE: /ACCT/GROUP/d0/

 FILE CODE : 0          FOPTIONS: DIRECTORY
 BLK FACTOR: 1          OWNER  : MANAGER.ACCT
 REC SIZE: 32(BYTES)       GROUP ID: ACCT
 BLK SIZE: 32(BYTES)       SECURITYREAD  :
 EXT SIZE: 0(SECT)             WRITE  :
 NUM REC: 4                APPEND :
 NUM SEC: 64                LOCK  :
 NUM EXT: 2                EXECUTE :
 MAX REC: 67107839            **SECURITY IS ON
                 FLAGS  : 1 ACCESSOR,SHARED
 NUM LABELS: 0          CREATED : TUE, JUL 21, 1992, 1:10 PM
 MAX LABELS: 0          MODIFIED: TUE, JUL 21, 1992, 2:16 PM
 DISC DEV #: 2          ACCESSED: WED, JUL 22, 1992, 11:40 AM
 SEC OFFSET: 0          LABEL ADDR: $000000E1 $0009A220
 VOLCLASS : MPEXL_SYSTEM_VOLUME_SET:DISC
/ACCT/GROUP/d0:
```

The next example illustrates the use of the `FORMAT=4` (SECURITY) option to display the security matrix for all objects one level below the group (in this case, `d0`).

```
/ACCT/GROUP/d0:listfile /ACCT/GROUP/@,4
 ********************
 FILE: /ACCT/GROUP/d0/

 ACCOUNT  READ :
        WRITE :
        APPEND :
          LOCK :
        EXECUTE :
 GROUP   READ :
         WRITE :
         APPEND :
          LOCK :
        EXECUTE :
          SAVE :
 FILE - READ :          FCODE:  0
         WRITE :          **SECURITY IS ON
         APPEND :           ACD EXISTS
          LOCK :
        EXECUTE :
 FOR MANAGER.ACCT: RACD, TD, RD, CD, DD
```

The next example illustrates the use of the FORMAT=-2 (ACD) option to display the access contol definition (ACD) for file f4 in subdirectory d2. Note that all users (@.@) have read ACD (RACD) access for this file.

```
/ACCT/GROUP/d0:listfile ./d2/f4,-2

 PATH= /ACCT/GROUP/d0/d2/

 -ACD ENTRIES- FILENAME
 @.@        : RACD       f4

/ACCT/GROUP/d0:
```

Related Information

Commands LISTF, PLISTF (UDC), LISTFTEMP, LISTSPF (for spool files), FINDFILE (UDC), FINDDIR (UDC), LISTDIR (UDC)

Manuals None

LISTFTEMP

Displays information about one or more temporary files.

Syntax

LISTFTEMP[*fileset*] [,*listlevel*] [;*listfile*]

Parameters

fileset Specifies the set of temporary files to be listed. The default is @, producing a listing of all temporary files. You may select the temporary file(s) to be listed by using the fully qualified form for *fileset*:

filename[.*groupname*[.*accountname*]]

Use the # symbol to specify a single numeric character. Use the ? symbol to specify a single alphanumeric character. Use the @ symbol to specify one or more alphanumeric characters. By itself, @ represents all the members of a set.

Refer to appendix G for examples of using wildcard characters.

listlevel Specifies the level (amount and format) of information about the temporary file(s) you select. The default is zero.

The following Table 5-2 displays the *listlevel* options available.

Table 5-2 List Options

Option	Displayed Information
–2	Displays the file's ACD (access control definition). System Managers can view the ACD for any file. Account Managers can view the ACD for files in that account. File creators can view the ACD for their files. Other users can view an ACD only if that ACD specifies that the user has RACD (read ACD) access.
–1	Shows only the file label in hexadecimal. The hexadecimal display generated by this format option only serves a diagnostic purpose in MPE/iX and is subject to change.
0	For each directory, this option displays PATH=The name of the file is displayed in a multicolumn format. This is the default.
1	Displays the file name, file code, record size, record format, and other file characteristics such as ASCII or binary records, carriage-control option, file type, current end-of-file location, and the maximum number of records allowed in the file.
2	Displays the file name, file code, record size, file type, current end-of-file location, and the maximum number of records allowed in the file. It also displays the blocking factor, number of sectors in use, number of extents currently allocated, and the maximum number of extents allowed.

Option	Displayed Information
3 –3	Displays the file name, record size, extent size, number of records, user's access rights, and other file characteristics including the date created, modified, and last accessed. The same information for MPE and HFS files is displayed except for the following differences: • Fully qualified MPE file name is replaced by an absolute pathname. • Creator field displays the fully qualified user ID (user.acct) of the file owner. • For MPE groups, the SECURITY field displays SAVE; for entries other than MPE groups it is blank. All file access matrix fields are blank for anything other than MPE accounts, MPE groups, and files in an MPE group. • The LOCKWORD field is omitted. The creator, group id, and label address are omitted in FORMAT=3. These can be obtained by specifying -3 if you have sufficient capability (AM or SM)
4	Displays the security matrix for the file. This includes account, group and file-level security, and the access rights for the user. For MPE groups and MPE accounts, the security matrix for group, account, and account-only are displayed. The rest of the fields of the file access matrix are blank. For HFS directories, and files within HFS directories, all the fields of the file access matrix are blank. In addition, LISTFILE displays the message ACD EXISTS.
5 –5	Shows LISTFILE,3 data and all file-specific data in LISTFILE,3 type format (KSAM, SPOOL, and symbolic links). If a file has no unique data, only the option 3 data is shown.
6	Shows the absolute pathname of the file.
7	Shows all file specific data in LISTFILE,5 type format, but does not show LISTFILE,3 data. If a file has no unique data, only the file name is displayed. Default = 0 (FILES).
8	Shows all accessors of the files listed. Restrictions apply
9	Shows level 8 information and details about processes accessing the files including file locking data. Restrictions apply.

SECTORS The number of sectors allocated for the file on disk. This number is always a multiple of 16 (the page size in MPE/iX). This value is an indication of the size of the file.

#X Number of extents. This number is displayed only to maintain compatibility with MPE V/E. This value does not indicate the size of the file. The variable-extent structure of MPE/iX permits a file to have a variable number of extents, all of variable size.

MX Maximum number of extents. This number is displayed only to maintain compatibility with MPE V/E. If the value is greater than 32 (the limit on MPE V/E), then * is displayed.

listfile The name of the output file. The default is $STDLIST. If you specify *listfile*, it is automatically created as a new ASCII file with variable-length records, closed in the temporary domain, and with user-supplied carriage-control characters (CCTL), OUT access mode, and EXC (EXCLUSIVE access) option. All other characteristics are the same as they would be with the FILE command default specifications.

Operation Notes

This command lists descriptions of one or more temporary files at the level you specify. You may list any file, but, based on your capabilities, there are restrictions on the kind of information that is available to you.

Use

This command is available from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command.

Examples

The following examples show the output displayed for the various levels of the LISTFTEMP command:

The next example shows "Level 0" output.

```
LISTFTEMP

TEMPORARY FILES FOR PETE.HPXLII,DEVELOP
LINKCLK.DEVELOP.HPXLII
```

The next example shows "Level 1" output.

```
LISTFTEMP ,1

TEMPORARY FILES FOR PETE.HPXLII,DEVELOP
ACCOUNT= HPXLII    GROUP= DEVELOP
FILENAME CODE   LOGICAL RECORD

          SIZE  TYP    EOF   LIMIT
LINKCLK      128W  FB     0   1023 (TEMP)
```

The next example shows "Level 2" output.

```
LISTFTEMP ,2

TEMPORARY FILES FOR PETE.HPXLII,DEVELOP
ACCOUNT= HPXLII     GROUP=  DEVELOP
FILENAME CODE -LOGICAL RECORD- SPACE

        SIZE TYP   EOF   LIMIT R/B SECTORS #X MX
LINKCLK     128W FB    0   1023  1   128 1 8 (TEMP)
```

The next example shows "Level 3" output. Fields containing "n/a" are not implemented.

```
LISTFTEMP ,3
********************
FILE: LINKCLK.DEVELOP.HPXLII

FCODE: 0        FOPTIONS: ASCII,FIXED,NOCCTL,STD
```

```
BLK FACTOR: 16     CREATOR:
REC SIZE: 80(BYTES)  LOCKWORD:
BLK SIZE: 640(BYTES)  SECURITYREAD  :ANY
EXT SIZE: 25(SECT)        WRITE :ANY
NUM REC: 501           APPEND :ANY
NUM SEC: 165           LOCK  :ANY
NUM EXT: 7            EXECUTE:ANY
MAX RED: 501      **SECURITY IS ON
MAX EXT: 7       FLAGS: n/a
NUM LABELS: 0     CREATED: FRI, 21 SEP 1986, 11:55 AM
MAX LABELS: 0     MODIFIED: FRI, 21 SEP 1986, 12:34 PM
DISC DEV #: 3     ACCESSED: FRI, 21 SEP 1986, 12:46 PM
SEC OFFSET: 0

VOLSET  :      MPEXL_SYSTEM_VOLUME_SET
or
VOLNAME  :      MPEXL_SYSTEM_VOLUME_SET: MEMBER1
or
VOLCLASS :      MPEXL_SYSTEM_VOLUME_SET: DISC
CLASS   : DISC    LABEL ADDR: $00000010 $0010E014
```

The next example shows "Level -1" output.

```
LISTFTEMP LINKCLK,-1
F = LINKCLK

00000000 44495343 20202020 20202020 20202020 20202020 .........@..
LINK
20202020 20202020 20202020 20310000 4C495354 53202020 DEVELOP    ..
20202020 20202020 44455645 4C4F5020 20202020 20202020 HPXLII   .
00000000 4850584C 49492020 20202020 20202020 00000000 ...PETE
52455445 50202020 20202020 20202020 50455445 20202020 .|..,2....#.,7.6
20202020 20202020 00000000 FC000000 04580001 13915EF4 ,2.|..#.,2.....#
00010405 00000000 00000300 00020CEE 0EA78B32 00020CEE .......H........
0EA78B32 00020CEE 12F61E2D 00020CEE 0EA78B32 00000000 ................
000000A0 000001F5 00000000 00000000 00000000 00000000 ...........  ..
00009C90 00000000 00000000 00000000 00000050 00000500 ................
00100000 00190007 000F0000 02020000          C.8x@.R.@.Q.....
```

Related Information

Commands	LISTF, LISTFILE, SAVE
Manuals	None

LISTGROUP

Displays information for one or more groups.

Syntax

LISTGROUP[*groupset*] [,*listfile*] [;PASS] [;FORMAT={ SUMMARY|BRIEF|}]

Parameters

groupset Specifies the set of groups to be listed. For account managers (AM) and system managers (SM), the default is all (@) groups within the user's logon account; for general users, the default is the logon group. You may use wildcard characters to specify more than one group. Use the ? symbol to specify a single alphanumeric character Use the # symbol to specify a single numeric character. Use the @ symbol to specify all combinations of valid characters. You may also specify *group.account* if you have system manager (SM) capability.

listfile The name of the output file. The default is `$STDLIST`, a temporary file that cannot be overwritten by a `BUILD` command. It is automatically specified as a new ASCII file with variable-length records, closed in the temporary domain, and with user-supplied carriage-control characters (`CCTL`), OUT access mode, and `EXC` (EXCLUSIVE access) option. All other characteristics are the same as they would be with the `FILE` command default specifications.

`PASS` Permits users with AM and SM capability to see the group password.

`FORMAT` Used to specify one of several display formats.

`SUMMARY` Provides a summary of the group information. If `FORMAT` is not specified, `SUMMARY` is the default.

`BRIEF` Generates a list of `group.account` names only.

Operation Notes

This command produces group information in an ASCII format.

Use

This command is available from a session, a job, a program, or in BREAK. Pressing **Break** aborts the execution of this command. If you do not have account manager (AM) or system manager (SM) capability, you can list only your logon group. Users with AM capability may list any group in their account. Users with SM capability may list any group in the system.

Example

In the following example, since the user does not have AM or SM capability, the password does not appear in the display.

```
LISTGROUP DEVELOP;PASS;FORMAT=SUMMARY

*******************
GROUP: DEVELOP.TEST

DISC SPACE: 5752(SECTORS)      PASSWORD: **
CPU TIME : 0(SECONDS)        SECURITYREAD  : GU
CONNECT TIME: 0(MINUTES)            WRITE  : GU
DISC LIMIT: UNLIMITED             APPEND : GU
CPU LIMIT : UNLIMITED             LOCK  : GU
CONNECT LIMIT: UNLIMITED            EXECUTE : GU
PRIV VOL : n/a                 SAVE  : GU
FILE UFID: $000D4001 $80001050 $000FF620 $00000008 $0000000A
MOUNT REF CNT: n/a
HOME VOL SET : MPE_SYS_VOL_SET
CAP: IA,BA
```

```
LISTGROUP @.@;FORMAT=BRIEF

ACCOUNT1.PAYROLL
ACCOUNT2.PAYROLL
DEVELOP.TEST
DOC.MASTER
JONES.TEST
PUB.SYS
```

Related Information

Commands	ALTGROUP, LISTACCT, LISTUSER, NEWGROUP, PURGEGROUP LISTFILE
Manuals	*Performing System Management Tasks* (32650-90004)

LISTJOBQ

LISTJOBQ lists all available job queues in the system.

Syntax

```
LISTJOBQ
```

Parameters

none

Operation Notes

The **LISTJOBQ** command allows the user to list all the existing job queues in the system. It displays the queue name, limit, number of jobs in the queue that are in the EXEC state and the total number jobs in the queue, Number of jobs in the EXEC state plus number of jobs in the WAIT state). This command is not allowed in the SYSSTART file.

Example

```
:listjobq

JOBQ       LIMIT    EXEC  TOTAL

HPSYSJQ     3500       1      1

MYJOBQ       100       1      1

MJQ           10       1      2
```

Related Information

Commands	NEWJOBQ, SHOWJOB, PURGEJOBQ, SHOWJOB;JOBQ
Manuals	

LISTLOG

Lists currently active logging identifiers on the system and whether automatic 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. This option can be used only by the creator of the logging identifier.

Operation Notes

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.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. System supervisor (OP) or user logging (LG) capability is required 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.PEJ
TEST1   MARK.MPEM    YES   NO   M001.KSAM3
TEST2   PAT.MPEM    NO    NO  TEST.ALVAR
```

Related Information

Commands	`ALTLOG`, `CHANGELOG`, `GETLOG`, `LOG`, `OPENLOG`, `SHOWLOGSTATUS`, `RELLOG`
Manuals	*User Logging Programmer's Guide* (32650-60012)

LISTREDO

Displays the contents of the command line history stack. You may specify the format in which the listing appears, and whether it appears on $STDLIST or in a file. (Native Mode)

Syntax

LISTREDO[START=*m*] [;END=*n*] [;OUT=*outfile*] [;{ ABS REL UNN }]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

START or END Specifies the range of commands to be displayed. Table 5-3 illustrates the effect of various START or END definitions.

Table 5-3 History Stack Ranges

Start	End	Effect
(omitted)	(omitted)	Lists all commands in the redo stack.
m	*n*	Lists commands *m* through *n*.
m	(omitted)	Displays commands *m* through the last command in the stack.
(omitted)	*n*	Displays the stack from the first command through command *n*.

If *m* and *n* are negative values, they refer to relative command numbers (relative to the most recent command, which is -1). If *m* and *n* are positive, they refer to absolute command numbers (the order in which they were entered). To display a single line, *m* must equal *n*.

REL Displays the commands in their relative sequence (from -*m* to -1), where -1 denotes the most recent command in the stack.

ABS Displays the commands in their absolute order (the order in which they were entered). ABS is the default.

UNN Suppresses numbering of the commands during display.

outfile Sends the listing to a disk file named *outfile* instead of to the default, $STDLIST. New disk files are created TEMP. File equations are ignored, unless *outfile* is preceded by an asterisk (*). You must use a file equation to overwrite a permanent file.

Operation Notes

The LISTREDO command displays the contents of the REDO command line stack. By default, the display order is from the earliest command to the most recent command. Before any command line is displayed, anything resembling a lockword is blanked out. However, any lockwords remain active and available for editing through the DO and REDO commands.

Use

This command is available in a session, job, or in BREAK. It is not available from a program. Pressing **Break** aborts the execution of this command.

Examples

If three commands are written to the REDO stack and the third command is LISTREDO, the display appears as:

```
 1) COMMANDONE
 2) COMMANDTWO
 3) LISTREDO
```

If the third command were LISTREDO ;REL, the display appears as:

```
 -3) COMMANDONE
 -2) COMMANDTWO
 -1) LISTREDO ;REL
```

To create a permanent disk file called CMDFILE containing the output from LISTREDO, enter:

```
 BUILD CMDFILE;REC=-80,,,ASCII;DISC=9
 FILE LIST=CMDFILE,OLD
 LISTREDO -10,-2;OUT=*LIST;UNN
```

CMDFILE contains a listing of nine command lines, but without the command number; -10 is 9 lines distant from the most recent command; -2 is one line distant from the most recent command. The most recent command is not listed.

Related Information

Commands	DO, REDO
Manuals	*Using the HP 3000 Series 900: Advanced Skills* (31126A Opt. 002)

LISTSPF

Produces a listing of input and output spooled files. (Native Mode)

Syntax

LISTSPF[[IDNAME=] { *spoolid* (*spoolid*[,*spoolid*] ...) }] [[;SELEQ=] { *select-eq* ^*indirect_file* }] [;DETAIL ;STATUS]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameter Definitions

spoolid

One or more spool file IDs: #I*nnn* for input spool files or #O*nnn* for output spool files. These IDs are assigned by the spooling subsystem at spool file creation time. The # is optional; but if it is used, an O or I must also be used. If it is not used, the O is also optional for output spool files; that is, if neither [#]O nor [#]I is specified, then [#]O is assumed.

- The symbol @ may be used to specify all spool files.
- The symbol O@ may be used to specify all output spool files.
- The symbol I@ may be used to specify all input spool files.
- If @, O@, or I@ is specified, it must be the only `SPOOLID` value supplied. @, O@, and I@ are mutually exclusive.
- If you specify duplicate `SPOOLID`s, the system displays a warning message.

A user with SM or OP capability or a console user who specifies O@ will see all output spool files on the system. A user with AM capability who specifies O@ will see all output spool files created by users in the same account. All other users are limited to files they have created. Similar rules apply to I@ and @. The default is all the output spool files created by the current *user.account*. The default `SPOOLID` for the console user is all the output spool files on the system.

select-eq

The selection equation is used as a filter on the set of spool files selected. Only spool files whose attributes satisfy all filter requirements will be listed.

When you use a selection equation, enclose the entire equation in square brackets, and enclose individual keyword specifications (such as `PAGES<100`) in parentheses. For example, you use the following command to display all the output spool files from `user.acct` that have less than 100 pages:

```
LISTSPF O@;SELEQ=[(OWNER=user.acct)AND(PAGES<100)]
```

If you are not an SM, OP, AM, or console user, the following command displays all the output files in your default group with a priority greater than 2 that were created before September 30, 1994.

```
LISTSPF O@;SELEQ=[(PRI>2)AND(DATE<09/30/94)]
```

Selection equations have the following format. The symbol ::= should be interpreted as "can be replaced by".

select-eq ::= [*equation*]

equation **::=** { *parm*{ > >= < <= <> = } *value* (*equation*) NOT *equation* *equation*{ AND OR } *equation* }

In a selection equation, the logical operator AND takes precedence over the logical operator OR. For example, suppose you enter the following command:

```
LISTSPF O@;SELEQ=[FILEDES=REPT OR OWNER=BOB.ACCTG AND PRI>8]
```

In this example, the selection equation `[FILEDES=REPT OR OWNER=BOB.ACCTG AND PRI>8]` is the same as `[FILEDES=REPT OR (OWNER=BOB.ACCTG AND PRI>8)]`.

value ::= Appropriate values per data type. For example, `STATE=READY` or `PRI>6`.

parm ::= The parameter (*parm*) may be one of several attributes of the spool file, used as filters. The *parm* choices are described below.

- *parm ::=* `DEV`: LDEV number, device name, or device class name. You can use wildcards for device name and device class name.
- *parm ::=* `FILEDES`: Formal or actual file designator for the spool file. You may use wildcards.

  For example, if you enter the file equation below and print to it, EPOCLONG becomes the spool file's `FILEDES`.

  ```
  FILE EPOCLONG;DEV=EPOC;ENV=LPLONG.ENV.SYS
  PRINT MYFILE,*EPOCLONG
  ```

  You may also select files based on a null string by entering `FILEDES= ""` or `FILEDES= ".` You must include such a construct if you specifically want to select such an attribute. Note that `""` is not the same as "   "; the blank is significant.
- *parm ::=* `SPOOLID`: Spoolfile identifier number in the format #O*nnn* or #I*nnn*.

  The "#" is optional; but if it is used, an O (for output) or an I (for input) must also be used. If # is not used, the O is also optional for output spool files; that is 123 is the same as #O123. The valid range of `SPOOLID`s is 1 ≤ *nnn* ≤ 9,999,999. (The commas are for clarity; do not enter any commas in the actual equation.)

- *parm ::=* PAGES: Number of pages in the spool file (if known). A positive integer number is expected. This attribute does not apply to input spool files; therefore, any logical *condition* involving the attribute always returns FALSE when tested against an input spool file.
- *parm:=* FORMID: Form name. You can use wildcards. (The *formid* is an ASCII string up to 8 characters, the first of which must be a letter.).

  This attribute does not apply to input spool files; therefore, any logical *condition* involving the attribute always returns FALSE when tested against an input spool file.

  You may also select files based on a null string by entering FILEDES= "" or FILEDES= ". You must include such a construct if you specifically want to select such an attribute. Note that "" is not the same as "   "; the blank is significant.
- *parm:=* STATE: The state can be one of READY, ACTIVE, OPEN, CREATE, PRINT, PROBLM, DELPND, SPSAVE, DEFER, XFER.
- *parm ::=* JOBNAME: Job or session name under which the spool file was created. The job name can consist of up to 8 alphanumeric characters, the first of which must be a letter.

  For a job input spool file, the JOBNAME shown is allocated to that job, *not* the job or session that streamed it.

  You may use wildcards. The JOBNAME=@ parameter is a different use of the @ symbol in that it wildcards an optional field. The omission of this optional parameter indicates that all entries are displayed whether or not a job name exists.
- *parm ::=* DISP: Disposition: SPSAVE or PURGE. This attribute does not apply to input spool files; therefore, any logical *condition* involving the attribute always returns FALSE when tested against an input spool file.
- *parm ::=* COPIES: Number of copies. Minimum is 1, maximum is 65,535.

  This attribute does not apply to input spool files; therefore, any logical *condition* involving the attribute always returns FALSE when tested against an input spool file.
- *parm ::=* PRI: Output priority. Minimum is 0, maximum is 14.
- *parm ::=* JOBNUM: Job or session number under which the spool file was created, for example: #S257, #J329, or S*n* (the "#" is optional) where 1 ≤ *n* ≤ 16,383. (The comma is shown for clarity; do not enter any commas in the actual equation.)

  For a job input spool file, the JOBNUM shown is allocated to the job, *not* the job or session that streamed it.

You may use some wildcards; J@ accepts all jobs, S@ accepts all sessions. J'@ and S'@ are also allowed, The apostrophe (') indicates an imported spool file or a spool file recovered during `START NORECOVERY`.

- *parm ::=* `RECS`: Number of records in the spool file. A positive integer is expected.
- *parm ::=* `OWNER`: The user under which the spool file was created. The format of the *owner* is *user.account*. If the account is not specified, the user's current account is assumed. You can use wildcards.

  For a job input spool file, the `OWNER` is the user logon for the job, *not* the job or session that streamed it.
- *parm ::=* `JOBABORT`: Select based on whether or not this is the `$STDLIST` of a job which aborted when an error was encountered but no `CONTINUE` was in effect.

  Valid values are TRUE and FALSE. Only "=" and "<*gt;" are allowed as relational operators.

  This attribute does not apply to input spool files; therefore, any logical *condition* involving the attribute always returns FALSE when tested against an input spool file.
- *parm ::=* `DATE`: Creation date in the format *mm/dd/yy* or *mm/dd/year*. Note that the year can be in the form of *yy*, as in 10/10/88, or in the form of *year*, as in 10/10/1988; both are legal syntax for the DATE parameter.

^indirect_file

The *indirect_file* parameter specifies the name of a file containing the selection equation. It must be preceded by a caret (^). The selection equation contained in the file may not exceed 509 characters in length, including the brackets in which it must reside. There is no restriction on the indirect file code. If the record size exceeds 509, only 509 characters per record are read and a warning is issued. Backreferencing to a formal file designator is also allowed for an *indirect_file* name; that is, ^**filename* is allowed. Any file is accepted as an *indirect_file*, unless the file system returns an error from `FOPEN` or `FREAD`.

There is no limit to the number of records in the *indirect_file*, only the total character count.

Records are processed as follows:

- Leading and trailing blanks are stripped.
- If the last nonblank character is an ampersand (&), it is also stripped; otherwise, one blank is added back to the end of the record as a delimiter.
- The character count of the record is added to that of the records processed previously. If the total character count exceeds 509, an error is returned. If the total is less than 509, the current record is appended to previous records.

- This process repeats until either 509 characters have been counted or the end-of-file is detected. Records terminating with or without ampersands may be mixed as desired in the indirect file.
- If the resulting string is ≤509 characters, it is parsed.
- If the parser detects a syntax error, or if any non-blank character follows the closing bracket (]) of the *select-eq*, an error is returned and the *select-eq* is not processed.

DETAIL Produces a two-line description of the specified spool file(s). The default is a one-line display (not detailed).

STATUS By default, LISTSPF displays a listing of selected spool files, followed by a statistical summary of those spool files, known as the status display.

Specification of the STATUS option causes only the status summary to be displayed summarizing the specified fileset. STATUS and DETAIL cannot be specified together.

Operation Notes

This command is provided to enable users to obtain a list of spool file information without having to look for it within a list that includes other files.

The display for LISTSPF is different from the SHOWIN/SHOWOUT display. LISTSPF displays both output and input spool files. The display shows output spool files, then input spool files, and finally a summary status display.

The parameters are divided into three groups: selection, detail and status.

The selection group allows a user to limit the display of spool files to a subset of the overall group of spool files on the system.

The detail parameter displays more than the default information on the files that have been selected.

The status parameter displays summary status only.

These parameters can be combined as desired except for ;DETAIL and ;STATUS, which are mutually exclusive.

This command displays status information for one or more spool files. The information reflects the status at the time the command is entered and always appears on the standard list device. You may use CI I/O redirection to redirect the output to a file.

Within device or device class, READY, CREATE, PRINT, and XFER state output spool files are displayed first, sorted by priority and then by date and time. Output spool files in DEFER, PROBLM, or SPSAVE states are shown next sorted by order of state and then priority and time.

Output spool files are displayed first, followed by input spool files and the status display. The display for input spool files is not sorted.

Display Field and Description

Below is an example of the first line of the display for LISTSPF. Following the example is a description of each field in the display.

```
 SPOOLID  JOBNUM FILEDES PRI  COPIES DEV  STATE  RSPFN  OWNER

 #01    J12345 $STDLIST  6    1 EPOC  CREATE RSPFN  THISUSER.ACCOUNT1
```

SPOOLID The unique spool file identifier.

JOBNUM The job or session identifier of the job or session that created the spool file. The exception to this is that the *jobnum* for a JOB input spool file is the job number assigned the process whose $STDIN is (or will be) this input spool file, as opposed to the *jobnum* of the process that streamed the job. Job numbers containing an apostrophe (i.e., J'123) indicate that the spool file was imported by SPFXFER, RESTORE, or was recovered after a START NORECOVERY.

FILEDES The formal or actual file designator for the spool file. Printing to a file equation such as FILE EPOCLONG;DEV=EPOC;ENV=LP88LONG.HPENV.SYS creates spool files whose formal designator is EPOCLONG.

DEV The LDEV, device name or device class name that is the destination of the spool file. LDEVs are intentionally displayed with leading zeroes to simulate a device name. When you specify LDEVs with SELEQ, you need not supply the leading zeroes.

PRI The input or output priority of the spool file.

COPIES The total number of copies of the spool file to be printed.

STATE The current state of the spool file. READY and DELPND apply to input spool files as well as output spool files.

- CREATE: An output spool file is being created; that is, an output spooled device has been opened and is being written to, generating an output spool file. When the device is closed, the spool file enters the READY state.
- READY: An output spool file is ready to be printed or an input spool file is ready to be accessed.
- ACTIVE: An input spool file is active when it is being read from a STREAM file or a spooled device to disk.
- OPEN: A JOB input spool file (the $STDIN for a batch job) is being accessed by the job's CI process or a DATA input spool file is being accessed by a process.
- PRINT: An output spool file is being printed.

  If you enter the LISTSPF command while a trailer is being printed, you may observe two spool files in the PRINT state at the same time for the same device. This is because the spooler must open its next file to print before printing the trailer of its current one. (This is required to manage headers and trailers properly). Also note that you see only one file in the PRINT state during a trailer if the next file is another copy of the current file.

- DEFER: An output spool file is in the deferred state.
- SPSAVE: The SPSAVE option was specified when the spool file was created or at any time before it would have been deleted after its final copy was printed. That final copy has been printed, so the spool file is now in this state instead of being deleted.
- PROBLM: The target device of the spool file does not match any device name or device class on the system. This usually occurs because the spool file has been restored to a system that has a different configuration than the system from which the spool file was stored.
- DELPND: Either the spooler has printed the last copy of the output spool file and is waiting for one or more users to close the spool file before purging it, or someone has requested that the spool file be deleted (using the DELETESPOOLFILE or the SPOOLF...;DELETE command) and the file management routines are waiting for the last FCLOSE of the spool file before purging it.
- XFER: The spool file has been selected for transportation from one node of a network to another. The XFER state is supported (in that it may be displayed, and used as a STATE in a selection equation), but is provided only for use as desired by third-party software providers. The spooler never places a file in this state nor uses the state as a basis for spooler actions.

RSPFN

The column under each letter R, S, P, F, and N, contains the respective letter as a flag indicating something about the spool file described in that row.

- R indicates a restartable spooled job file, that is, one for which the ;RESTART option was specified in the :JOB record.
- S indicates that SPSAVE disposition has been specified for this spool file. The spool file will be saved in the OUT.HPSPOOL group and account after the last copy is printed.
- P indicates that the spool file is private.
- F indicates that the spool file has a forms message associated with it and requires special forms on which to print. If a *formid* is present, its identity can be seen, using the ;DETAIL option, on the second line of the display for the given spool file.
- N indicates that the spool file is not complete because insufficient account-level, group-level or system disk space was available when the spool file was created or the system aborted while the spool file was being created.

`OWNER` This is the fully qualified name of the creator of the spool file.

Below is an example of the optional second line of the display, followed by an explantion of each display field.

```
 FORMID  JOBNAME   COPSRM  SECTS  RECS  PAGES  DATE     TIME

      TESTJOB      1  250   500  ~9   12/20/88  8:39
```

`FORMID` An 8-character display, the first of which is a letter. If an *F* appears in the `RSPFN` column but this field is blank, it means that the file has a forms message but *formid* was not specified.

`JOBNAME` The job or session name of the user who created the spool file or, for a job input spool file, the name of the job that will use the input spool file as its `$STDIN` file..

`COPSRM` The number of copies of this file that remain to be printed, including any currently printing copy.

`SECTS` The number of sectors occupied by the spool file.

`RECS` The number of records in the spool file.

`PAGES` The number of physical pages in the spool file. This quantity is accurate only for CIPER protocol, 2680/88 page printers, and HP5000/F1xx page printers, and then only if the device has printed at least one complete copy. The device keeps track of the pages as they are printed and returns the correct count at the end of the copy. Until the actual count is known, an approximate count calculated as *number_of_records* ÷ 60, and denoted by a leading tilde (~) is displayed.

For serial printers, even the count without the tilde is approximate because it is calculated as a best guess from the spool file data. It is not returned by the device because serial printers have no provisions for reporting this information.

`DATE` The date that the file first entered the READY state (*mm/dd/yy*).

`TIME` The time that the file first entered the READY state in 24-hour form (*hh:mm*).

`STATUS` The status display has the following format:

```
INPUT SPOOL FILES      OUTPUT SPOOL FILES
ACTIVE  = 1;        CREATE  = 2;    READY  = 3;
OPEN   = 2;        DEFER   = 1;    SELECTED = 4;
READY  = 3;        DELPND  = 0;    SPSAVE  = 1;
              PRINT   = 1;    XFER   = 0;
              PROBLM  = 0;

TOTAL IN FILES = 6;    TOTAL OUTFILES  = 8;
   IN SECTORS = 144;      OUT SECTORS = 13090;

OUTFENCE = 6
OUTFENCE = 10 FOR LDEV 6
```

This display consists of three parts. The values in the first two parts represent only those spool files selected for display.

- The itemized count of spool files in each of the various states. They are shown in two groups, input spool files to the left of the display and output spool files to the right. Of these, only SELECTED is not a state. Instead, SELECTED shows the total count of spool files whose output priority is higher than the global outfence; that is, SELECTED displays the sum of printing files plus those READY files whose output priority is above the global outfence.
- The total number of input spool files, the sector count for input spool files, the total number of output spool files, and the sector count for output spool files.
- The global outfence and any device-specific outfences.

Use

This command may be issued from a session, job, a program, or in BREAK. It is breakable. Only files to which the user has access are displayed.

Examples

Following are some examples of the displays produced by LISTSPF. The first and third examples display all output spool files for the current *user.account* not using the console. The second example displays all spool files for the current *user.account* not using the console.

```
LISTSPF

SPOOLID JOBNUM FILEDES PRI COPIES DEV     STATE RSPFN OWNER
#0123  J12   SP    13   2 PP      PRINT  F  DEV.HPE
#0124  S14   LIST    9   1 00000012   READY  F  DEV.HPE
#0128  J144  $STDLIST  8   1 EPOC     READY    DEV.HPE
#01233  S1234  OUTLIST  0   1 FASTLP    DEFER    DEV.HPE
INPUT SPOOL FILES    OUTPUT SPOOL FILES
ACTIVE = 0;      CREATE  = 0;    READY  = 2;
OPEN  = 0;      DEFER  = 1;    SELECTED = 3;
READY  = 0;      DELPND  = 0;    SPSAVE  = 0;
            PRINT  = 1;    XFER   = 0;
            PROBLM  = 0;

TOTAL IN FILES  = 0;  TOTAL OUTFILES  = 4;
   IN SECTORS = 0;     OUT SECTORS = 5964;

OUTFENCE = 6
```

```
:LISTSPF @;DETAIL

SPOOLID JOBNUM  FILEDES PRI COPIES DEV    STATE RSPFN OWNER
    FORMID  JOBNAME   COPSRM SECTS   RECS  PAGES DATE   TIME
#O123  J12   SP    13   2 PP     PRINT   F  DEV.HPE
         TESTJOB     1 250    500   125 07/09/88 8:39

#O124  S14   LIST   9   1 00000012  READY   F  DEV.HPE
    PAYCHECK TESTJOB     1 250    500    ~9 12/20/88 8:39

#O128  J144   $STDLIST 8   5 EPOC    READY      DEV.HPE
         LPJOB      3 250    127    21 12/20/88 22:19
```

```
#O1233 S1234  OUTLIST  0   1 FASTLP   DEFER DEV.HPE
         TESTJOB     1 250    500    ~9 12/20/88 8:39

#I564  J164   $STDIN       00000010  READY     DEV.HPE
         BATCHJOB      17     12       2/20/88 22:23

INPUT SPOOL FILES    OUTPUT SPOOL FILES
ACTIVE = 0;      CREATE = 0;        READY  = 2;
OPEN  = 0;      DEFER  = 1;        SELECTED = 3;
READY  = 1;      DELPND = 0;        SPSAVE  = 0;
            PRINT  = 1;        XFER   = 0;
            PROBLM = 0;

TOTAL IN FILES  = 1;  TOTAL OUT FILES  = 4;
   IN SECTORS = 17;    OUT SECTORS = 1000;

OUTFENCE = 6
```

```
:LISTSPF;STATUS

INPUT SPOOL FILES    OUTPUT SPOOL FILES
ACTIVE = 0;      CREATE = 0;  READY  = 2;
OPEN  = 0;      DEFER  = 1;  SELECTED = 3;
READY  = 0;      DELPND = 0;  SPSAVE  = 0;
            PRINT  = 1;  XFER   = 0;
            PROBLM = 0;

TOTAL IN FILES  = 0;  TOTAL OUTFILES  = 4;
   IN SECTORS = 0 ;    OUT SECTORS = 1000;

OUTFENCE = 6
```

Related Information

Commands SPOOLER, SPOOLF, SHOWIN, SHOWOUT, LISTFILE

Manuals *Native Mode Spooler Reference Manual* (32650-90166)

LISTUSER

Displays information for one or more users.

Syntax

LISTUSER[*userset*] [,*listfile*] [;PASS] [;FORMAT={ SUMMARY|BRIEF|DETAIL}]

Parameters

userset Specifies the set of users to be listed. The default is all (@) users (and accounts) within the user's capabilities (AM or SM). Use wildcard characters to specify more than one user. Use the ? symbol to specify a single alphanumeric character. Use the # symbol to specify a single numeric character. Use the @ symbol to specify zero or more alphanumeric characters.

listfile The name of the output file. The default is `$STDLIST`, a temporary file that cannot be overwritten by a `BUILD` command. It is automatically specified as a new ASCII file with variable-length records, closed in the temporary domain, user-supplied carriage-control characters (`CCTL`), OUT access mode, and `EXC` (EXCLUSIVE access) option. All other characteristics are the same as they would be with the `FILE` command default specifications.

`PASS` Permits users with account manager (AM) and system manager (SM) capability to see the user password.

`FORMAT` Used to specify one of several display formats.

`SUMMARY` Provides a summary of the account information. If `FORMAT` is not specified, `SUMMARY` is the default.

`BRIEF` Generates a list of `user.account` names only.

`DETAIL` Displays all information associated with the account.

Operation Notes

This command produces user information in an ASCII format.

Use

This command is available from a session, a job, a program, or in BREAK. Pressing **Break** aborts the execution of this command. If you do not have system manager (SM) or account manager (AM) capability, you can list only your logon user. If you have AM, you may list any user in your account. If you have SM, you may list any user in the system.

Example

In the following example, since the user has AM capability, the password is displayed:

```
LISTUSER PETE;PASS

  ...or...

LISTUSER PETE;PASS;FORMAT=SUMMARY

*******************

USER: PETE.TEST
HOME GROUP: DEVELOP         PASSWORD: MYPASS
MAX PRI  : 150           LOC ATTR: $00000000
LOGON CNT : 1                WRITE  : GU
CAP: AM,AL,GL,DI,CV,UV,LG,CS,ND,SF,IA,BA,PH,DS,MR,PM
```

```
LISTUSER @;FORMAT=BRIEF

PETE.TEST
MIKE.TEST
CHRIS.TEST
```

```
LISTUSER PETE;FORMAT=DETAIL

*******************

USER    : PETE.TEST
PASSWORD  : **
UID    : ##
GID    : ##
MAX PRI  : 150
LOC ATTR  : $00000000
LOGON CNT : 2
HOME DIR  : /UI/DEVELOP
LOGON CI  : /SYS/PUB/CI
CAP    : AM,AL,GL,DI,CV,UV,LG,PS,CS,ND,SF,BA,IA,PM,MR,DS,PH
```

NOTE In the above example, the "##" in the UID and GID fields indicate that no UID or GID is associated with the user. The PXUTIL utility should be run to create UID and GID entries.

Related Information

Commands	ALTUSER, LISTACCT, LISTGROUP, NEWUSER, PURGEUSER, PXUTIL
Manuals	None

LMOUNT

Requests a logical reservation of a volume set. This informs the system that the volume set is to be reserved system-wide. The equivalent native mode command is VSRESERVESYS. (Native Mode)

Syntax

LMOUNT[{ * *volumesetname* }] [.*groupname*[.*acctname*]] [;GEN=[*genindex*]]

NOTE For the MOUNT, DISMOUNT, LDISMOUNT, and LMOUNT commands a volume set name such as V.G.A can have no more than eight characters in any part of the name. If the length of V, G, or A exceeds eight characters, an error is reported.

Parameters

* or <blank> Specifies the home volume set for the group and account specified, or for the logon group and account if *groupname* or *groupname.acctname* is not specified.

volume- setname An artificial component of a volume set name used to maintain backward compatibility with MPE V/E.

groupname Used only for compatibility with MPE V/E.

acctname Used only for compatibility with MPE V/E.

genindex A value from -1 to 32,767 specifying which generation of the home volume set is to be reserved. A value of -1 indicates that any generation is permitted. If omitted, the system ignores the generation when attempting to satisfy the MOUNT request.

Operation Notes

When the console operator executes the LMOUNT command, all disk drives containing members of the specified volume set become RESERVED. Each volume set is logically attached to the drive until an LDISMOUNT command is executed, at which time the disk drive is no longer reserved on a system-wide basis. A VSCLOSE may then be issued to remove the volume set. (Refer to the VSCLOSE command in this chapter.)

Executing an LMOUNT command does not prevent users from issuing a MOUNT command for the volume set in question. Users may issue a DISMOUNT command for the specified volume set, but doing so has no effect; the LMOUNT command takes priority over a general user command.

System users issue mount requests implicitly through their programs, or explicitly with a MOUNT command.

If the mountable volumes facility was enabled with VMOUNT ON,AUTO, MPE/iX automatically attempts to satisfy the mount request; the LMOUNT succeeds if the specified volume set is physically connected to the system.

If the mountable volumes facility was enabled with VMOUNT ON (omitting the AUTO parameter), you must reply to your own mount request, even though the volume set may already be mounted and in use.

Volume sets in MPE/iX are not tied to groups and accounts. This is different from the MPE V/E scheme of disk partitioning.

The MPE/iX new naming convention for volume sets differs from the MPE V/E naming convention for private volumes. MPE/iX volume set names may consist of any combination of alphanumeric characters, including the underbar (_) and the period (.). The name must begin with an alphabetic character and consist of no more than 32 characters.

Table 5-4 illustrates the difference between naming conventions for MPE/iX volume sets and MPE V/E private volumes.

Table 5-4 Command Acceptance of Naming Conventions - LMOUNT Command

Specify	MPE V/E xxxMOUNT Command Accesses	MPE/iX VSxxxxxx Command Accesses
myset.grp.acct	The volume set named myset.grp.acct.	The volume set named myset.grp.acct.
myset	The volume set named myset.logongrp.logonacct.	The volume set myset.
*.grp.acct	The home volume set of the group grp in account acct.	Causes an error.
myset_grp_acct	Error (name component longer than eight characters).	The volume set named myset_grp_acct.
m_g_a	The volume set named m_g_a.logongrp.logonacct, provided it exists. If it does not exist, an error is reported.	The volume set named m_g_a.

In MPE V/E, the name V.G.A indicates that V is the name of a volume set, that G is the name of a group, and that A is the name of an account.

MPE/iX accepts that name in that form, but no interpretation is made as to the referencing of G and A. Instead, MPE/iX treats V.G.A as a single, long string name, just as it would treat A_VERY_LONG_NAME_FOR_SOMETHING.

As a convenience to established Hewlett-Packard users, MPE/iX accepts the naming convention that was used for MPE V/E private volumes.Thus MOUNT V.G.A succeeds and MOUNT V accesses the same volume set, provided you are logged on to account A, group G. The MPE V/E commands are able to default the logon account and group.

However, VSRESERVE V succeeds only if there is a volume set V in existence. The MPE/iX commands does not call up any default specifications for group and account. VSRESERVE V.G.A succeeds only if a volumeset V.G.A is online. With MPE/iX VSxxxxxx commands, the .G.A component of this name is interpreted as a string, neither more nor less specific than _G _A.

If a volume set is named according to the MPE V/E naming convention (V.G.A), you must use an unambiguous reference when using the MPE/iX volume set commands.

It is recommended that you not use the MPE V/E naming convention and xxxMOUNT commands. Instead use the MPE/iX naming convention and VSxxxxxx commands. Alternating between MPE V/E and MPE/iX commands may lead to errors. For example, MOUNT X used in a job stream attempts to access a volume set named X.logongrp.logonacct which may or may not be your intention.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It is executable only at the console unless distributed to users with the ALLOW command.

Examples

To reserve a volume set named DATABASE.PAYROLL.ACCTNG, enter:

```
LMOUNT DATABASE.PAYROLL.ACCTNG
```

You may also use the VSRESERVESYS command:

```
VSRESERVESYS DATABASE.PAYROLL.ACCTNG
```

Related Information

Commands	MOUNT, DISMOUNT, DSTAT, VSRESERVE, VSRELEASE
Manuals	*Volume Management Reference Manual* (32650-90045)

LOG

Starts, restarts, or stops user logging.

Syntax

LOG ***logid***{ ,RESTART ,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.

Operation Notes

This command allows you to start, restart, or stop user logging. For further discussion of user logging, refer to the *User Logging Programmer's Guide* (32650-60012).

To change log files without the delay normally caused by executing a LOG command, use the CHANGELOG command to enable interactive log file changing. Use the AUTO parameter of the ALTLOG and GETLOG commands to enable automatic log file changing.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

This command can be executed only by users to whom it has been *allowed* (see ALLOW command) or to users logged onto the console (or to a terminal that has taken the console via the CONSOLE command). System Supervisor (OP) capability is also required.

Example

To start the logging process identified by *logid* LOGPROCX, enter:

```
LOG LOGPROCX,START
```

Related Information

Commands	ALTLOG, CHANGELOG, GETLOG, SHOWLOGSTATUS
Manuals	*User Logging Programmer's Guide* (32650-60012)

=LOGOFF

Aborts all executing jobs/sessions and prevents any further logons. You may optionally specify one job or one session that is to remain logged on.

Syntax

=LOGOFF[#S*nnn*]

or

=LOGOFF[#J*nnn*]

Parameters

#S*nnn* or #J*nnn* The number of the session or the job that is to remain logged on after all others are aborted. Default is that all sessions and all jobs are logged off.

Operation Notes

This command sets the job and session execution limits to 0 and aborts all jobs and sessions, including the session logged on to the system console. You may leave one session or one job logged on by specifying that session or job with either the #S*nnn* or #J*nnn* parameter.

Execution of this command leaves the system in a job/session inactive state, unless you specify one job or session that is to remain logged on. Job and session introduction is disabled. No other jobs or sessions are logged on until a **CTRL A** LOGON is entered.

Any pending requests that require a =REPLY from the system console must be satisfied before issuing =LOGOFF, or MPE/iX

Use

This command may be issued only from the physical console. Pressing **Break** has no effect on this command.

Examples

To abort all executing jobs/sessions, enter:

```
  CTRL A
=LOGOFF
16.53/25/ALL JOBS LOGGED-OFF
```

To abort all executing jobs and sessions except #S2, enter:

```
  CTRL A
=LOGOFF #S2
```

To perform the MPE/iX **CTRL A** logoff, enter the following commands:

```
  CTRL A
 =LOGOFF #S1
 =LOGON
  LIMIT 0,0
  JOBFENCE 0
```

This logs off all users except #S1 and allows only users with system manager (SM) and system supervisor (OP) capability to log on. It is assumed here that the console operator controls #S1.

Related Information

Commands	=LOGON, ABORTJOB, TELL, WARN
Manuals	*System Startup, Configuration, and Shutdown Reference Manual* (32650-90042)
	Performing System Operation Tasks (32650-90137)

=LOGON

Enables job/session processing following a `=LOGOFF` command.

Syntax

```
=LOGON
```

Parameters

None.

Operation Notes

This command enables the processing of jobs/sessions following the execution of the `=LOGOFF` command. The `=LOGON` command reestablishes the job/session limits that were in effect before the execution of a `=LOGOFF` command and allows jobs/sessions to log on again.

Use

This command may be issued from a session, program, or in BREAK, but not from a job. Pressing **Break** has no effect on this command. It may be issued only from the physical console.

Example

To enable job/session processing, enter:

```
  CTRL  A
=LOGON
```

Related Information

Commands	`=LOGOFF`
Manuals	*System Startup, Configuration, and Shutdown Reference Manual* (32650-90042)
	Performing System Operation Tasks (32650-90137)

MOUNT

Sends a request to the system to reserve a volume set (keep it online). The set must be online in order to have the command take effect. (Native Mode)

Syntax

MOUNT[{ * *volumesetname* }] [.*groupname*[.*acctname*]] [;GEN=[*genindex*]]

Parameters

* or <blank> Specifies the home volume set for the group and account specified, or for the logon group and account if *groupname* or *groupname.acctname* is not specified.

volume- setname An artificial component of a volume set name used to maintain backward compatibility with MPE V/E. The *volumesetname* can be a maximum of 8 characters.

groupname Used only for compatibility with MPE V/E. The *groupname* can be a maximum of 8 characters.

acctname Used only for compatibility with MPE V/E. The *acctname* can be a maximum of 8 characters.

genindex A value from -1 to 32,767 specifying which generation of the home volume set is to be reserved. A value of -1 indicates that any generation is permitted. If omitted, the system ignores the generation when attempting to satisfy the MOUNT request.

Operation Notes

The MOUNT command reserves a specific volume set for use. It notifies the system that the volume set is to remain online and is not to be taken offline by a VSCLOSE command.

Volume sets in MPE/iX are not tied to groups and accounts. This is different from the MPE V/E scheme of disk partitioning.

The MPE/iX naming convention for volume sets differs from the MPE V/E naming convention for private volumes. MPE/iX volume set names may consist of any combination of alphanumeric characters, including the underbar (_) and the period (.). The name must begin with an alphabetic character and consist of no more than 32 characters.

Table 5-5 illustrates the differences between the MPE/iX and MPE V/E naming conventions for volume sets.

Table 5-5 Command Acceptance of Naming Conventions - MOUNT Command

Specify	MPE V/E xxxMOUNT Command Accesses	MPE/iX VSxxxxxx Command Accesses
myset.grp.acct	The volume set named myset.grp.acct.	The volume set named myset.grp.acct.
myset	The volume set named myset.logongrp.logonacct.	The volume set myset.
*.grp.acct	The home volume set of the group grp in account acct.	Causes an error.
myset_grp_acct	Error (name component longer than eight characters).	The volume set named myset_grp_acct.
m_g_a	The volume set named m_g_a.logongrp.logonacct, provided it exists. If it does not exist, an error is reported.	The volume set named m_g_a.

In MPE V/E, the name V.G.A indicates that V is the name of a volume set, that G is the name of a group, and that A is the name of an account.

MPE/iX accepts that name in that form, but no interpretation is made as to the referencing of G and A. Instead, MPE/iX treats V.G.A as a single, long string name, just as it would treat A_VERY_LONG_NAME-FOR_SOMETHING.

As a convenience to established Hewlett-Packard users, MPE/iX accepts the naming convention that was used for MPE V/E private volumes. Thus MOUNT V.G.A succeeds and MOUNT V accesses the same volume set, provided you are logged on to account A, group G. The MPE V/E commands are able to default the logon account and group.

However, VSRESERVE V succeeds only if there is a volume set V in existence. The MPE/iX commands do not call up any default specifications for group and account. VSRESERVE V.G.A succeeds only if a volumeset V.G.A is online. With all MPE/iX VSxxxxxx commands, the .G.A component of this name is interpreted as a string, neither more nor less specific than _G_A.

If a volume set is named according to the MPE V/E naming convention (V.G.A), you must use an unambiguous reference when using the MPE/iX volume set commands.

Various user commands that give you access to your logon group's home volume set implicitly initiate reservation requests if the volume set is not reserved already. An example of one of these commands (BUILD) is:

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

To issue a reserve request programmatically, you may issue an FOPEN call referencing a file residing on an unreserved volume set; this causes an implicit user initiated reserve request. An FOPEN reserve 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/iX Intrinsics Reference Manual* (32650-90028) for a description of a programmatic reserve request.

It is recommended that you not use the MPE V/E naming convention and xxxMOUNT commands. Instead use the MPE/iX naming convention and VSxxxxxx commands. Alternating between MPE V/E and MPE/iX commands may lead to errors. For example, MOUNT X used in a job stream attempts to access a volume set named X.logongrp.logonacct, which may or may not be your intention.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. Use volumes (UV) or create volumes (CV) capability is required to use this command.

Examples

You are logged on to account MYACCT in group GRP. To request the system operator to reserve volume set MYSET in that group and account, with a generation index of 43, enter:

```
MOUNT MYSET;GEN=43
```

If you are logged on in another *group.account*, enter:

```
MOUNT MYSET.GRP.MYACCT;GEN=43
```

Related Information

Commands	DISMOUNT, LMOUNT, DSTAT, VSRESERVE, VSRELEASE
Manuals	*Volume Management Reference Manual* (32650-90045)
	MPE/iX Intrinsics Reference Manual (32650-90028)
	Volume Management Reference Manual (32650-90045)

NEWACCT

Creates a new account with 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*]] [;ONVS=*volumesetname*] [;GID=[*gid*]] [;UID=[*uid*]] [;USERPASS=[{ REQ OPT }]]

The USERPASS parameter is only available if the HP Security Monitor has been installed.

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. Table 5-6 lists the default capabilities assigned to an account manager.

Table 5-6 Account Manager Default Capabilities

Attribute	Default
password	None
capabilitylist	Same as the account capability
subqueuename	Same as the account maximum priority
localattribute	Same as account local attributes
Home Group	PUB
UID	A unique identifier
GID	A unique identifier

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 Disk 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, and so 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 The 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
Save Files = SF
Access to Nonshareable
 I/O Devices  = ND
Use Volumes   = UV
Create Volumes  = CV
Use Communication
 Subsystem   = CS
Programmatic Sessions = PS
User Logging   = LG
Process Handling = PH
Extra Data Segments  = DS
Multiple RINs  = MR
Privileged Mode = PM
Interactive Access  = IA
Batch Access   = BA
```

Default is AM, AL, GL, SF, ND, IA, BA.

fileaccess The restriction on file access pertinent to this account. Default is R,L,A,W,X:AC, 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
A  =  Append
W  =  Write
X  =  Execute
S  =  Save
```

LOCK allows exclusive access to the file. APPEND implicitly specifies LOCK. WRITE implicitly specifies APPEND.

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
```

The default is no security restrictions at the account level. Two or more 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 nonpriority 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. While it is not part of standard MPE/iX security provisions, programmers may define local attributes (which are checked by the `WHO` intrinsic) to enhance the security of their software. Default is double word 0.

`ONVS` Specifies a particular volume set on which the account is to be built. It must be a volume set already defined and recognized by the system. A `NEWACCT` must be specified twice, once without the `ONVS` parameter, and once with it. The first `NEWACCT` builds the account on the system volume set (from which the account is accessed). The second `NEWACCT` builds the account on the volume set where files in this account will exist.

The only other parameter that works with `ONVS` is the `FILES` parameter.

volume- setname Volume set names consist of from 1 to 32 characters, beginning with an alphabetic character. The remaining characters may be alphabetic, numeric, the underscore, and periods.

If you specify a *volumesetname*, you must specify the full name of the volume set. When `ONVS=`*volumesetname* is specified, the volume set directory is assumed. When `ONVS=` is specified without *volumesetname*, the system directory is assumed.

gid Group ID to be added to the group database. The *gid* must be an unique positive (non-zero) 32-bit integer. Default is for MPE to create a value. Duplicate id numbers are not allowed.

uid User ID to be created for the account manager in the user database. The *uid* must be an unique positive (non zero) 32-bit integer. Default is for MPE to create a value. Duplicate id numbers are not allowed. The *uid* is associated to the manager of the account.

`REQ` Specifies that all users in the account are to have non-blank passwords. If you require user passwords, MPE/iX assigns the account manager a blank, expired password. The account manager must select a new password the first time the Manager logs on. It is available only if the HP Security Monitor has been installed.

OPT Specifies that users of the account may or may not have passwords. This is the default. It is available only if the HP Security Monitor has been installed.

Operation Notes

The NEWACCT command may be executed only 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= **Return**), 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 designates account managers for those accounts, the new account managers may log on and redefine their own attributes and those of their PUB groups. Account managers can also define new users and groups. The capabilities and attributes that the account manager assigns 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 linking programs that use extra data segments).

The PUB group is initially assigned the same capability class attributes, permanent file space limit, CPU limit, and connect-time limit as the account, but no password. Its initial security allows READ and EXECUTE access to all users who successfully log on 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).

NOTE If you specify volume-related commands or parameters for a volume set that is not currently mounted, or for an account that does not exist, MPE/iX returns an error message.

Use

This command may be issued from a session, a job, a program, or in BREAK. Pressing **Break** has no effect on this command. System manager (SM) capability is required to use this command.

Examples

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
```

To create the account DOCTOR on the system volume set, with the manager named WHO, and on the volume set called MY_VOL, you must create it with two parallel commands:

```
NEWACCT DOCTOR,WHO;CAP=IA,BA,GL,AM,AL
NEWACCT DOCTOR,WHO;ONVS=MY_VOL
```

The second command connects the accounting structures established on the system volume and on the volume set. By default, however, the PUB group of this account is on the system volume set.

To place the PUB group on the volume set MY_VOL, you need to use the PUB parameter in the first command:

```
NEWACCT DOCTOR,WHO;CAP=IA,BA,SF,ND,GL,AM,AL
NEWACCT DOCTOR,WHO;ONVS=MY_VOL
ALTGROUP PUB.DOCTOR;HOMEVS=MY_VOL
```

To create the account DOCTOR on the system volume set, with the manager named WHO, and a UID of 150 and a GID of 120, enter:

```
NEWACCT DOCTOR,WHO;UID=150;GID=120;CAP=IA,BA,SF,ND,GL,AM,AL
```

Related Information

Commands	ALTACCT, ALTUSER, LISTACCT, NEWGROUP, NEWUSER, PURGEACCT, REPORT, DISKUSE
Manuals	*Native Mode Spooler Reference Manual* (32650-90166)

NEWDIR

Creates a directory. (Native Mode)

Syntax

`NEWDIR`[DIR=] *dir_name* [;SHOW | NOSHOW]

Parameters

dir_name The name of the directory that you are creating (required). The *dir_name* is assumed to be an MPE name unless it begins with a a dot (.) or a slash (/), which indicates an HFS directory.

The *dir_name* may not end in a slash, have wildcard characters, or reference a file equation.

`SHOW` Echoes the absolute pathname of the newly created directory to $STDLIST. `SHOW` is the default.

`NOSHOW` Suppresses the display of the absolute directory name.

Operation

The `NEWDIR` command creates a directory named *dir_name*. All parent directories must already exist. The new directory inherits the group ID (GID) from its parent directory and the user ID (UID) from the user creating the directory. The special directory entries dot (.) and dot-dot (..) are automatically created under *dir_name*.

By default `NEWDIR` creates an MPE-named directory, which means that *dir_name* must follow all MPE naming rules. Since the MPE name syntax defines three levels, fully (or partially) qualified MPE-named directories can only be created under MPE groups. Unqualified MPE-named directories are created relative to the CWD.

If *dir_name* begins with a dot (.) or a slash (/), then HFS naming rules are enforced.

Directories do not support lockwords, file equations, or system defined file names (for example, `$NEWPASS`).

You must have create directory entries (CD) permission for the parent directory and save files (SF) capability. Furthermore, traverse directory entries (TD) ac cess is required for each directory component named in *dir_name*. (Refer to the `ALTSEC` command in this chapter for further information on directory permissions.)

Use

The `NEWDIR` command may be invoked from a job, a session, a program, or in BREAK. Pressing **Break** has no effect on this command.

Examples

In the following two examples, a user creates a directory called DIR1. In the first example, the full pathname of the directory is specified in all uppercase since HFS syntax is case-sensitive. In the second example, the user enters the information in lower case using the MPE syntax *dir_name.groupname.acctname*. (Any case-lower-, mixed-, or uppercase could be used since the CI will automatically shift pathnames entered in MPE syntax to uppercase.)

```
  NEWDIR /MYACCT/MYGRP/DIR1

 NEWDIR dir1.mygroup.myacct
```

The following example creates an HFS-named directory called john by specifying the full pathname of the directory. Since the directory will reside in the MPE/iX account MYACCT, and since HFS syntax **is** case-sensitive, the user enters "MYACCT" in uppercase.

```
  NEWDIR /MYACCT/jones/cmdf/john
```

The following example creates an MPE-named directory called DIR1 in the current working directory (CWD). Note that the *dir_name* is shifted to uppercase.

```
  NEWDIR dir1
```

The following example creates an HFS-named directory called dir1 in the current working directory (CWD). Note that in this example, the *dir_name* is **not** shifted to uppercase.

```
  NEWDIR ./dir1
```

The following example creates an HFS-named directory called dir2 by specifying POSIX syntax:

```
  NEWDIR ./dir2
```

The next example creates an MPE-named directory called A.*group.logon_acct*.

```
  NEWDIR a.group
```

Related Information

Commands	LISTFILE, CHDIR, PURGEDIR, LISTDIR (UDC), FINDDIR (UDC), NEWACCT, NEWGROUP
Manuals	*Performing System Management Tasks* (32650-90004)

NEWGROUP

Creates a new group within an account.

Syntax

NEWGROUP *groupname* [.*acctname*] [;PASS=[*password*]] [;FILES=[*filespace*]] [;CPU=[*cpu*]] [;CONNECT=[*connect*]] [;CAP=[*capabilitylist*]] [;ACCESS=[(*fileaccess*)]] [;ONVS=*volumesetname*] [;HOMEVS=*volumesetname*]

Parameters

groupname The name of the new group, which must consist of one to eight alphanumeric characters, beginning with an alphabetic character.

acctname The account in which the group is to reside. System manager (SM) capability is required to use this parameter.

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 Disk 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. Default is a storage limit equivalent to the account's *filespace*.

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 are warned when logging on; other users are denied access.

The CPU limit for a group cannot be set to a value greater than the corresponding limit currently defined for the account in which that group resides. Default is unlimited CPU-time. The counter may be set to zero with the RESETACCT command.

connect

The limit on the total cumulative session connect-time, in minutes, that the group is allowed. This limit is checked at logon and whenever the session initiates a new process. The maximum value you may specify with this command is 2,147,483,647 minutes. Default is the account connect limit.

A group's connect limit cannot be specified as greater than the corresponding limit currently designed for the account in which the group resides. 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.

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

where R, L, A, W, X specify modes of access by types of users (ANY, AC, GU, AL, GL) 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
```

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

ONVS — Specifies a particular volume set on which the group is to be built. The volume set must be already defined and recognized by the system. The NEWGROUP command must be specified twice before files can be created in this group on a mountable volume set. The first NEWGROUP builds the group on the system volume set (from which the account is accessed). The second NEWGROUP then builds the account on the mountable volume set. Create volumes (CV) capability is required to use this parameter.

HOMEVS — Sets the home volume set to the set specified by *volumesetname*. Create volumes (CV) capability is required to use this parameter.

volume- setname — Volume set names consist of from 1 to 32 characters, beginning with an alphabetic character. The remaining characters may be alphabetic, numeric, the underscore, and periods.

If you specify a *volumesetname*, you must specify the full name of the volume set.

Refer to the VSxxxxxx commands in this chapter.

Operation Notes

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 and the same capability class attributes, permanent file space limit, and CPU-time limit as the account but no password. Its initial security grants READ (R) and EXECUTE (X) access to all users (ANY) who successfully log on 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.

NOTE If you specify volume-related commands or parameters for a volume set that is not currently mounted, or for an account that does not exist, MPE/iX returns an error message.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. Account manager (AM) capability is required to use this command.

Examples

To create a new group named GROUP1 (on the system volume set), which will be assigned all default capabilities, enter:

```
NEWGROUP GROUP1
```

To create a new group named G2 in the account GRIMSBY (on the system volume set) and give it process handling (PH) and multiple RINs (MR) capabilities, enter:

```
NEWGROUP G2.GRIMSBY; CAP=PH,MR
```

To create the group LEELA on the nonsystem volume set TIME_LORD, you must use two parallel commands, as follows:

```
NEWGROUP LEELA;CAP=IA,BA,ND;HOMEVS=TIME_LORD
NEWGROUP LEELA;ONVS=TIME_LORD
```

The first command creates the group on the system volume set, but also informs the system that the files are to reside on another volume set that will be the home volume set for the files.

The second command builds the group on the volume set TIME_LORD.

Related Information

Commands	NEWACCT, NEWUSER, NEWDIR, LISTGROUP, ALTGROUP
Manuals	*Performing System Management Tasks* (32650-90004)

NEWJOBQ

The NEWJOBQ command creates a new job queue.

Syntax

```
NEWJOBQ qname  [;limit=n]
```

Parameters

qname Name of the queue to be created. If a queue of this name already exists, an error is indicated.

limit Maximum number of jobs that can be allowed in this queue. The limit value can be changed using the *:limit [+-]n;jobq= command*. If omitted, a value of zero is assumed.

Operation Notes

Limit is the only queue controlling property. The jobs in the queue are sorted by their INPRI. In case of a tie for INPRI, jobs are sorted by their INTRO time.

The global limit takes precedence over individual queue limits. That is, even if a jobqueue has a slot available, if the overall limit has been reached, jobs have to wait till one of the jobs finish or the global limit is increased. When a global slot becomes available, the next job is picked from among the eligible jobqueues (those which haven't yet reached their individual limits).

The job queues persist across reboots, provided a START RECOVERY is done. Any other system starts will cause the job queues to be deleted and they will have to be created again.

This command is available in a session, job, or in BREAK. Pressing [Break] has no effect on this command. This command is not allowed in the SYSSTART file.

SM/OP capability is required to execute this command.

Examples

```
:NEWJOBQ MYJOBQ;limit=100
```

Related Information

Commands	LISTJOBQ, PURGEJOBQ, SHOWJOB, ALTJOB
Manuals	

NEWLINK

Creates a link to a file, group, account, or directory. (Native Mode)

Syntax

NEWLINK[LINK=] *linkname* [;TO=] *sourceobject* [{ ;SYMBOLIC}]

Parameters

linkname The *pathname* that points to the file, that when created, will contain the link. *linkname* must resolve to a unique name. It may not be the name of an existing symbolic link, even if that link resolves to the name of a file or directory object that does not exist.

This is a required parameter. When specifying *linkname*, you may not use wildcard characters, file equations, or name a system defined file (such as $NULL).

sourceobject The path name to which a link is to be created. The *sourceobject* does not need to exist when creating symbolic links. This path must resolve to either a file, group, account, or directory name.

Security provisions of *sourceobject* do not affect the creation of symbolic link(s) to *sourceobject*.

This is a required parameter. When specifying *sourceobject*, you may not use wildcard characters, file equations, or name a system defined file (such as $NULL).

SYMBOLIC Specifies that the link to be created is a symbolic link. This is the default.

Operation Notes

You can use the `NEWLINK` command to create a link to a file, group, account, or directory.

When *newlink* represents a path to a symbolic link, the target of that symbolic link is used as the name of the new link that is being created. The `NEWLINK` command fails if the path represented by *linkname* points to a file or directory that already exists.

The following table lists all the CI commands that operate on files, groups, accounts, or directories and are affected by symbolic linking. Keep in mind the following data points when using Table 5-7 below:

- Typically, a symbolic link always resolves to its target name.
- The `Follow Link` column applies to the *filename* portion (last component) of an HFS path.

Table 5-7 CI Commands Affected by Symbolic Links

Command Name	Follow Link	Notes
CHGROUP	No	None
DISKUSE	Yes/No	Link is resolved before the operation is performed. If a symbolic link exists under the account that link is not resolved. Therefore disk space usage of its target is not included in the calculations.
LISTACCT	No	None
LISTFILE	No	Link is not resolved. Therefore, operation is performed on t he name specified. LISTFILE formats 5 and 7 may be used to determine the im mediate target of a symbolic link.
LISTGROUP	No	None
NEWLINK	No	The LINK parameter may not name a symbolic name. The TO parameter is not checked at all.
PURGE	Yes	This behaves differently than the UNIX rm command.
PURGEACCT	No/No	Link is not resolved. Therefore, operation is performed on the specified name. If a symbolic link exists under the account, that link is not resolved and its target is not removed.
PURGEDIR	Yes/No	Link is resolved before the operation is performed. If a symbolic link exists under the directory, that link is not resolved before it is removed. Therefore, its target is not affected.
PURGEGROUP	No/No	Link is not resolved. Therefore, operation is performed on the specified name. If a symbolic link exists under the account that link is not resolved and its target is not removed.
REPORT	No	Link is not resolved. Therefore, operation is performed on the name specified. Note that REPORT treats its first parameter as a group name. Therefore, if a link name is specified, that name is treated as a group name regardless of the type of its target.
RESTORE	No	Link is not resolved. Therefore, operation is performed on the name specified.
STORE	No	Link is not resolved. Therefore, operation is performed on the name specified.

You can issue the NEWLINK command from a session, job, program, or in BREAK. NEWLINK requires Save Files (SF) capability, Create Directory entry (CD) and Traverse Directory (TD) permissions.

Examples

The following tree structure will be used to construct the examples that follow it. Assume that the CWD is /ACCT1/PUB.

```
               ROOT
         |-
        |       |       |
       ACCT1      dir     SOFTWARE
        |       / \      |
     -   f1 f2
    |        |      |         |
    PUB      dir1     PUB        CODE
    |      /  \      |         |
  -   file1 file2  ACCTORG  -
  |     |               |   |    |    |
ACCTUDC  FILE3            COMMON TERMIO COMPALL  dir2
                                 / | \
                                f1 f2 dir3
```

To create a symbolic link named PAYCODE to the file PAYROLL.CODE.SOFTWARE, enter the following command:

```
:NEWLINK LINK=PAYCODE; TO=PAYROLL.CODE.SOFTWARE
```

Or, optionally use the positional parameters and enter:

```
:NEWLINK PAYCODE, PAYROLL.CODE.SOFTWARE
```

You now can access PAYROLL.SAFE.SOFTWARE through PAYCODE. For example, if you have read access to the file PAYROLL.CODE.SOFTWARE, you may enter the following command to print the contents of the file:

```
:PRINT PAYCODE
```

To create a symbolic link named FARFILE in PUB.ACCT1 that references /SOFTWARE/CODE/dir2/f1, enter the following command:

```
:NEWLINK LINK=FARFILE; TO=/SOFTWARE/CODE/dir2/f1
```

Suppose that file COMMON.CODE.SOFTWARE contains information that is used frequently. To display the contents of the file the following command has been used:

```
:PRINT COMMON.CODE.SOFTWARE
```

By creating a symbolic link to the file, you can simplify what users need to type to print it. For example:

```
:NEWLINK COMMON, COMMON.CODE.SOFTWARE
```

```
:PRINT COMMON
```

Suppose that a user is currently logged on as USER1 in the group PUB.SOFTWARE. To access the files in /ACCT1/dir1 directory without entering the full path name each time, USER1 may establish a link named "morecode" to that directory as follows:

```
:NEWLINK LINK=./morecode; TO=/ACCT1/dir1
```

Then, to get a list of the files under /acct1/dir1/, the user enters:

```
:LISTFILE ./morecode/
```

Absolute symbolic links

The following command creates FILE3 as a symbolic link to the nonexistent file SOURCE1.CODE.SOFTWARE.

```
:NEWLINK LINK=FILE3.PUB.ACCT1; TO=SOURCE1.CODE.SOFTWARE
```

The following command creates a symbolic link FILE4 as a link to an existing file.

```
:NEWLINK LINK=FILE4.PUB.ACCT1; TO=/SOFTWARE/CODE/dir2/f1
```

The following command creates /ACCT1/PUB/softPUB which points to /SOFTWARE/PUB, which is the group PUB in SOFTWARE account:

```
:NEWLINK LINK=/ACCT1/PUB/softPUB; TO=/SOFTWARE/PUB
```

The following command creates the symbolic link FILE9 as a link to the root directory:

```
:NEWLINK LINK=FILE9.PUB.ACCT1; TO=/
```

Relative symbolic links

The following examples show how to create symbolic links that are relative to the current working directory (CWD). For these examples assume that CWD is /SOFTWARE/CODE/dir2

The following command creates a symbolic link /SOFTWARE/CODE/F1 which points to the file ./f1:

```
:NEWLINK LINK=../F1; TO=./f1
```

The following command creates a symbolic link /SOFTWARE/CODE/F2 which points to the file ./f2:

```
:NEWLINK LINK=F2.CODE; TO=./f2
```

The following command creates the link /SOFTWARE/CODE/dir2/dir which points to the directory ../../../dir:

```
:NEWLINK LINK=./dir; TO=../../../dir
```

If you enter the following command, you will get an error message:

```
:NEWLINK LINK=FILE1.PUB.ACCT1; TO=/dir/f1

 Duplicate name in directory. (CIERR 906)
```

Similarly, the following command also generates an error message:

```
 :NEWLINK LINK=../TERMIO; TO=./f1

 Duplicate name in directory. (CIERR 906)
```

Related Information

Commands	PURGELINK, PURGE, LISTFILE
Manuals	None

NEWUSER

Creates a new user.

Syntax

NEWUSER
username[.*acctname*] [;PASS=[*password*]] [;CAP=[*capabilitylist*]] [;MAXPRI=[*subqueuename*]] [;LOCATTR=[*localattribute*]] [;HOME=[*homegroupname*]] [;UID=[*uid*]] [;USERPASS=[{ REQ OPT }] [*Expired*]]

The USERPASS parameter is only available if the HP Security Monitor has been installed.

Parameters

username The name of the user. The name must consist of one to eight alphanumeric characters, beginning with an alphabetic character.

acctname The account in which the user is to reside. System manager (SM) capability is required to use this parameter.

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 The list of capabilities, separated by commas, permitted to this user. Each capability is denoted by a two letter mnemonic, as shown in Table 5-8.

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 denied to the user. The user's capabilities are always verified to be a subset of the account's capabilities at logon. This prevents a user from being granted a capability not assigned the account. Note that CV capability, which allows users to define mountable non-system volumes, also gives the user UV capability, so that they may use mountable, non-system volumes. Default is IA, BA, ND, and 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 nonpriority 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 that can be used to further classify users. While it is not involved in standard MPE/iX 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.

uid User ID to be created for the account manager in the user database. The *uid* parameter must be an unique positive (non zero) 32-bit integer. Default is for MPE to create a value. Duplicate id numbers are not be allowed. The *uid* parameter provides file owner class security for MPE/iX.

REQ Specifies that the user must have a non-blank password. It is available only if the HP Security Monitor has been installed.

OPT Specifies that a user password is optional. This is the default. It is available only if the HP Security Monitor has been installed.

Expired The password expires immediately. The user cannot logon without selecting a new password. It is available only if the HP Security Monitor has been installed.

Operation Notes

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, a user ID, capabilities, and may 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 are assigned the user.

Table 5-8 User Capabilities

Capability	Mnenonic
System Manager	SM
Account Manager	AM
Account Librarian	AL
Group Librarian	GL
Diagnostician	DI
System Supervisor	OP
Network Administrator	NA
Node Manager	NM
Save Files	SF
Access to Nonshareable I/O Devices	ND
Use Volumes	UV
Create Volumes	CV
Use Communication Subsystem	CS
Programmatic Sessions	PS
User Logging	LG
Process Handling	PH
Extra Data Segments	DS
Multiple RINs	MR
Privileged Mode	PM
Interactive Access	IA
Batch Access	BA

Use

This command may be issued from a session, a job, a program, or in BREAK. Pressing **Break** has no effect on this command. Account manager (AM) or system manager (SM) capability is required to execute this command.

Examples

To define a new user named LHSMITH, assign a *password* of SMITTY and a home group of HOMEGPX, with the next available UID, enter:

```
NEWUSER LHSMITH;PASS=SMITTY;HOME=HOMEGPX
```

To define a new user named LHSMITH, assign a *password* of SMITTY, a home group of HOMEGPX, and assign a UID of 120, enter:

```
NEWUSER LHSMITH;UID=120;PASS=SMITTY;HOME=HOMEGPX
```

Related Information

Commands	ALTUSER, LISTUSER, NEWACCT, NEWGROUP, PURGEUSER
Manuals	*Performing System Management Tasks* (32650-90004)

NSCONTROL

Controls the Network Service subsystem.

Syntax

NSCONTROL function[;function] ...

function may be

START=[*service*[,*service*] ...] STOP= [*service*[,*service*] ...] ABORT AUTOLOGON=
[{ ON OFF }] [{ ,ALL [,*service*[,*service*]] }] LOADKEYS LOG=
[{ ON OFF }] [,ALL ,RPM ,ENV ,DSDAD ,DSSERVER ,VTSERVER] [{
,LOW ,HIGH }] SERVER= { *servername* *ALL* } [,*minservers*] [,*maxservers*] STA
TUS= [USERS SERVICES SERVERS SUMMARY ALL [,...]] VERSION [
=MOD]

Parameters

START [=*service* [,*service*]...]... Starts Network Services that are installed on your system. By default, all installed services are started. Optionally you may specify one or more specific services to be started. Possible specific services include:

LOOPBACK Enables remote users to run loopback diagnostic programs that connect to the local node.

The following services are available if you have the NS/3000 product installed:

NFT Enables remote users to transfer files to or from the local node using the DSCOPY command and intrinsic.

NFTL Enables local users to transfer files to or from remote nodes using the DSCOPY command and intrinsic.

NSSTAT Enables remote users to use the NSSTATUS intrinsic and DSLINE;SERVICES command to retrieve NS information from the local node.

NSSTATL Enables local users to use the NSSTATUS intrinsic and DSLINE;SERVICES command to retrieve NS information from the local and remote nodes.

PTOP Enables remote users to create and communicate with PTOP slave processes on the local node. The VT service must also be started. PTOP can be used only by HPDESK.

PTOPL Enables local users to create and communicate with PTOP slave processes on remote nodes. The VTL service must also be started. PTOPL can be used only by HPDESK.

RFA Enables remote users to access files and data bases on the local node.

`RFAL` Enables local users to access files and data bases on remote nodes.

`RPM` Enables remote users to create and kill processes on the local node using the Remote Process Management (RPM) service.

`RPML` Enables local users to create and kill processes on the local and remote nodes using the Remote Process Management (RPM) service.

`VT` Enables remote users to logon to the local node using HP's TCP message mode.

`VTA` Enables remote users to logon to the local node using TCP stream mode.

`VTL` Enables local users to log onto remote nodes using the REMOTE HELLO command.

`VTR` Enables remote users to access local terminals using the Virtual Terminal service.

`VTRL` Enables local users to access terminals on remote nodes using the Virtual Terminal service.

There may be additional services that can be enabled if other network products, such as Personal Productivity Center, are installed. Refer to that network product's documentation to obtain the appropriate service names.

`STOP` [=*service* [,*service*]...]... Stops Network Services, although existing users of the service(s) can continue until they terminate their use. By default, all installed services are stopped. Optionally you may specify one or more specific services to be stopped. Possible services include any of the same service names that you are allowed to specify for the START parameter described above.

`ABORT` Immediately terminates all the servers and services.

`AUTOLOGON` Enables or disables the autologon feature of the NFT, RFA and/or RPM services. Default: ON,ALL.

`AUTOLOGON=` [{ ON OFF }] [{ ,ALL [,*service*[,*service*]] }]

`LOADKEYS` Loads the Network Service command keywords while NS/3000 is active.

`LOG=`
[{ ON OFF }] [,ALL ,RPM ,ENV ,DSDAD ,DSSERVER ,V
TSERVER] [{ ,LOW ,HIGH }]

`ON` Enables detailed event logging of the specified module.

`OFF` Disables detailed event logging of the specified module.

For each Network Service software module, two levels of event logging are provided. These are HIGH, which logs all events, and LOW, the default, which logs a subset of the events, as specified below.

`ALL LOW` Logs LOW events for all modules. HIGH Logs HIGH events for all modules.

`RPM LOW` Logs RPMCREATE and RPMKILL requests. HIGH Same as LOW.

ENV LOW Logs environment information from DSLINE and REMOTE HELLO commands.

ENV HIGH Same as LOW, plus environment table locking and use counts.

DSDAD LOW Logs creation and deletion of sockets, ports, and server processes.

DSDAD HIGH Same as LOW, plus all received service requests and internal messages between DSDAD and server processes.

DSSERVER LOW Logs internal initialization messages between DSDAD and DSSERVER processes.

DSSERVER HIGH Same as LOW, plus all received messages from other processes.

VTSERVER LOW Logs internal initialization messages between DSDAD and VTSERVER processes.

VTSERVER HIGH Same as LOW, plus all received messages from other processes.

SERVER={ servername ALL } [,minservers] [,maxservers]

Dynamically alters the minimum or maximum number of servers. By default applies to all servers. Optionally you may specify one or more specific servernames. Possible servernames and their default minserver and maxserver values are:

LOOPBACK For the Loopback Service. Default minserver,maxserver values are 0,300.

NSSTATUS For the NSSTAT service (NSSTATUS intrinsic and DSLINE; SERVICES command). Default minserver, maxserver values are 0,300.

VTSERVER For VT and Reverse VT. Default minserver,maxserver values are 0,300.

The following servers are available if you have the NS/3000 product installed:

NFT For NFT (DSCOPY). Default minserver,maxserver values are 0,300.

DSSERVER For RFA, RDBA, PTOP and RPM. Default minserver, maxserver values are 0,300.

There may be additional servers to control if other network products, such as Personal Productivity Center, are installed. Refer to that network product's documentation to obtain the appropriate server names.

minservers The minimum number of servers which must be available at all times. Available servers which are not in active use are kept in reserve until a service request is received. If necessary, additional servers are created immediately to fit the new minimum specified. Valid range: 0..1250; however, see note below. Default for all current servers is 0.

maxservers The maximum number of servers of this type allowed to be active at one time. If necessary, reserved servers are terminated to fit the new maximum. Valid range: 0..32767; however, see note below. Default varies by server.

NOTE The total number of all active servers may not exceed 1250. The sum of all minservers must always be 1250 or less. You may specify a number greater than 1250 as one or more maxservers values, but there will never be more than a total of 1250 servers of all kinds at any one time.

STATUS Displays current status information about NS3000/XL Services.

```
STATUS[=USERS|SERVICES|SERVERS|SUMMARY|ALL[,...]]
```

The STATUS parameter can be unqualified, or can be keyword equated to one or more of the following values:

USERS Display the jobs and sessions on this node that are using Network Services.

SERVICES Display information about the services.

SERVERS Display information about the servers.

SUMMARY Display a summary of information about services, servers, and users.

ALL Same as specifying SERVICES, SERVERS and USERS.(DEFAULT)

VERSION Displays the overall version of the NS/3000 software. If qualified with the MOD keyword, also displays the version of each of the Network Services software modules.

VERSION[=MOD]

Operation Notes

NSCONTROL START Starts the Network Services subsystem.

NSCONTROL STOP Stops the Network Services subsystem. STOP executes a shutdown of Network Services. Existing users may continue using the Network Services until they complete their NS activity, but new users are prevented from using the services. When all users have finished using the NS subsystem, the subsystem will stop entirely.

NSCONTROL ABORT Immediately terminates all the servers and services of the Network Services. Note that STOP is the normal way to shutdown Network Services. The ABORT function should only be used in abnormal situations.

NSCONTROL AUTOLOGON Enables or disables the autologon feature of the NFT, RFA and RPM services. When disabled, remote users are required to establish a regular session via VT and :REMOTE HELLO before they can use NFT, RFA or RPM on this system. This is useful if you wish to force all remote users to execute a logon security UDC before they access anything on this system. When NS is first started, this feature is ENABLED.

NSCONTROL LOADKEYS Loads the Network Services command keywords from the ASCAT.NET.SYS catalog. You need to use this command only if the catalog is modified, such as for localization.

NSCONTROL LOG Enables or disables detailed event logging for the Network Service.

NSCONTROL SERVER Alters the characteristics of the Network Service processes.

NSCONTROL STATUS Displays information about the Network Services.

NSCONTROL VERSION Displays the overall version of the Network Services subsystem, and optionally the version of each of its modules.

Examples

Start the transport subsystem on the "LAN1" and "WIDE" networks, then start all of the NS network services:

```
:NETCONTROL START;NET=LAN1
:NETCONTROL START;NET=WIDE
:NSCONTROL START
```

Stop all Network Services, while letting existing users continue their work:

```
:NSCONTROL STOP
```

Stop the VT and Reverse VT services only. Let all other started services remain available:

```
:NSCONTROL STOP=VT,VTR
```

Enable logging of information from DSLINE and REMOTE HELLO commands, and service requests received by the DSDAD process:

```
:NSCONTROL LOG=ON,ENV,LOW;LOG=ON,DSDAD,HIGH
```

Set the minimum number of running DSSERVER processes to 2 and the maximum to 10:

```
:NSCONTROL SERVER=DSSERVER,2,10
```

Show the status of Network Services:

```
:NSCONTROL STATUS=SERVICES
SERVICE  TYPE  SERVER  DESCRIPTION
.
RPM REMOTE DSSERVER INCOMING REMOTE PROCESS MANAGEMENT
VTL LOCAL  VTSERVER OUTGOING VIRTUAL TERM
VT  REMOTE VTSERVER INCOMING VIRTUAL TERMINAL
.
.
.
```

Display the overall version and product number of the Network Services subsystem:

```
:NSCONTROL VERSION

Network Services overall subsystem version:  B.00.10
NS3000/XL SERVICES:  36920B
```

Related Information

Commands NETCONTROL

Manuals *Migration Process Guide* (30367-60003)

OCTCOMP

Converts a compiled MPE V/E program into native mode (NM) code for the HP 3000 Series 900. (Native Mode)

CAUTION Before using this command be sure your logon group and account does not contain files of the form Yn, Ynn or Ynnn where n is any alphanumeric character. `OCTCO`
`MP` may create temporary files named in this format and similarly named permane nt files may cause an error condition.

Syntax

OCTCOMP[*input*] [,[*targetfile*] [,[*list*]] [;INFO=*quotedstring*]] or OCTCOMP [*input*] [,[*targetfile*] [,[*list*] [,[INFO=] *quotedstring*]]] *

* Refer to the *help* option of the `INFO=`*quotedstring*

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

<omitted> If no parameters are specified `OCTCOMP` returns a command usage message and then exits.

input Name of the valid input program or SL file to be translated. A valid program is a CM PROG or SL file that can be loaded by the CM loader.

targetfile Name of the file to hold the translated output. (Refer to "Operation Notes" for a description of the default for this parameter.) If *targetfile* does not exist, it is created. If it already exists, it is purged and a new file is created.

list Name of the file to which object code translator writes listing and error messages. If you omit this parameter, the messages are sent to `$STDLIST`. All parameter parsing errors are written to `$STDLIST`.

`INFO=` *quotedstring* A list of parameters to define format and content of translated output. This parameter list must be surrounded with double or single quotation marks („ or '), and each parameter set in the list must be delimited with a semicolon if more than one set is given in command string.

help Print detailed description of `OCTCOMP` parameters. This is the only option that does not require a source. `OCTCOMP;INFO=„HELP„` is valid; so is `OCTCOMP ,,,„HELP„`.

add=seglist [;] Add translated segments to the file named in the command string. Note that the named file may already contain translated code. When you specify this option, `OCTCOMP` replaces already translated segments.

If you use the *add* option, the *targetfile*, *ignore*, and *trans* parameters are not permitted. This option works only for SL files.

errors [=*count*][;] Specify maximum number of errors to be reported before `OCTCOMP` terminates. The *count* value must be greater than zero. Errors are sent to named *list* file or, by default, to `$STDLIST`. If this parameter is given without the optional =*count*, all errors are reported. If you omit this parameter, `OCTCOMP` reports the first error, then terminates.

ignore= seglist[;] Do not translate specified segments. If *seglist* is omitted, an error is issued. If you enter the *ignore* parameter, you cannot use the *add* or *trans* parameters.

map [=*seglist*][;] Generate PMAP listing for specified segments. If *seglist* is omitted, PMAP listing is generated for all segments.

noovf [=*seglist*][;] Selectively ignore overflow traps in translating code. If *seglist* is omitted, *noovf* action is assumed for all segments. Specifying this option gives `OCTCOMP` permission to decide whether or not to catch overflow. The default is that `OCTCOMP` follows the behavior of the emulator.

Specifying this option improves the performance of integer arithmetic functions.

systemsl Inform the OCT utility that the user intends to make the file `SL.PUB.SYS`. This option is for users creating new systems. Several `SL.PUB.SYS` and system-dependent code improvements are performed when this option is specified.

trans= seglist[;] Translate only specified segments. If the named file contains translated code segments that are not listed, these segments will be set emulated and the translated code removed.

Where:

seglist

= *segnum*[...,*segnum*].

and

segnum

= 0 .. 9 - Decimal (default)

or 0 .. 7 - Octal

or $0 .. F - Hexadecimal

or A[..] .. Z[..] - Alpha (SL only) *

or ^*filename* (an indirect file) **

* In this form, a *segnum* identifier may consist of as many as 16 characters, beginning with an alphabetic character.

** You must number indirect files, and you cannot nest them. If you enter the *trans* parameter, you may not use the *add* or *ignore* parameters.

Operation Notes

The `OCTCOMP` command translates MPE V/E instructions into native mode instructions. If you specify *targetfile*, a new file is created. If you do not specify *targetfile*, `OCTCOMP` attempts to append the translated instructions to input file. The append fails and an error message is displayed if the input file is too small to qualify as an output file. In such a case, the solution is to specify *targetfile*.

User-defined labels are stripped from the input file, and they may not be added to a translated file.

After a new master installation tape is loaded, you must retranslate the file on which you used the *systemsl* option (to create `SL.PUB.SYS`). Otherwise, it runs in emulator mode.

The *noovf* parameter can improve the code generated. However, the user must ensure that the necessary conditions hold for code translated using this parameter. For the *noovf* parameter, the input code must not use the overflow trap mechanism.

The `OCTCOMP` command does not support the following:

- File equations involving the *input*, *targetfile*, or *list* files (backreferencing is not supported).
- `$NULL`, `$STDIN`, `$STDLIST`, or `$NEWPASS` for *input*, *targetfile*, or *list*; `$OLDPASS` for *targetfile* or *list*; but `$OLDPASS` is supported for *input*.
- Using an explicit or implicit `RUN` command to execute the `OCTCOMP` command.

Use

This command is available in a session, job, or program. It is not available in BREAK. Pressing **Break** aborts the execution of this command.

Examples

The following set of examples illustrates the use of the *add*=, *ignore*=, and *trans*= parameters and the effect each of them has on the content of the translated code output file with each succeeding invocation of OCTCOMP. In each example, the *input* file is assumed to consist of seven segments, 0 through 6.

In the following example, the translated output file, OCTOUT, consists of the SL file SOURCEIN and translated segments 1, 2, 3, and 4 only.

```
OCTCOMP SOURCEIN,OCTOUT;INFO="TRANS=1,2,3,4"
```

In the following example, the output in OCTOUT consists of the existing SOURCEIN object code image, existing translated segments 1, 2, 3, and 4, with translated segments 0, 5, and 6 appended to the file. Segment 5 does not have overflow detection.

```
OCTCOMP OCTOUT;INFO="ADD=0,5,6;NOOVF=5"
```

In the following example, the output in OCTOUT2 consists of the object code image from the existing file OCTOUT, with translated segments 0, 3, 5, and 6 only. This time segment 5 has overflow detection in OCTOUT2.

```
OCTCOMP OCTOUT,OCTOUT2;INFO='IGNORE=1,2,4'
```

This output would be the same if the call to OCTCOMP were given using the original object code *input* file SOURCEIN, as:

```
OCTCOMP SOURCEIN,OCTOUT2;INFO="IGNORE=1,2,4"
```

Using an indirect file:

OCTCOMP INSL;INFO="*add*=^*adlist*"

Here *adlist* is an unnumbered file in which segments (names or numbers) are separated by a blank, a comma, or a new line:

```
FSSEG1,FSSEG2
12
TIMAGE09
```

In this case, *add* is applied to all of the segments specified in the indirect file (^*adlist*).

Related Information

Commands	None
Manuals	*Migration Process Guide* (30367-60003)

OPENQ

Opens the spool queue(s) for a specified logical device, or device name or all device members of a device class. (Native Mode)

Syntax

OPENQ{ *ldev*[;SHOW] *devclass*[;SHOW] *devname*[;SHOW] @ }

Parameters

ldev The logical device number of the device.

devclass The device class name of the devices. The *devclass* parameter must begin with a letter and consist of eight or fewer alphanumeric characters.

devname The device name of the device. The *devname* parameter must begin with a letter and consist of eight or fewer alphanumeric characters. Note that it is not possible to have a device class name and a device that are the same. If you enter an alphanumeric character string, the command searches the device class list first, and then the device name list.

`SHOW` The `SHOW` parameter displays the current state (enabled or disabled) of the devices specified with the `OPENQ` command.

@ The @ parameter globally reenables all currently open spooling queues that were disabled because the system ran out of system domain disk space, a file limit was encountered on the `HPSPOOL` account or its groups, or the `SHUTQ @` command was entered.

If the spooling queues are disabled globally because the system is out of disk space or a file limit is encountered on the `HPSPOOL` account or its groups, the problem should be resolved before globally enabling spooling queues with the `OPENQ @` command.

Refer to the *Native Mode Spooler Reference Manual* (32650-90166) for more discussion on globally enabling and disabling spooling queues.

Use the @ option without any other parameter. The `SHOW` option entered with the @ option returns an error.

Operation Notes

The `OPENQ` command enables the operator to control the spool queue of a specified device or all devices of a device class without affecting the operation of spooler processes. It also gives the operator access to spool queues for which no spooler or physical device exists.

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 reopen it with an `OPENQ` command when the queue is clear.

The `OPENQ` command also serves as an option to the `STOPSPOOL` and `SPOOLER` commands.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be executed only from the console unless distributed to users with the ALLOW or ASSOCIATE command.

Example

To open the spool queue for logical device 6, enter:

```
OPENQ 6
```

To show the state of queues and other information about the specified device, enter:

```
OPENQ 6;SHOW
```

NOTE Classes are collections of devices, so operations (such as OPENQ) on a device class are applied to all devices in the class. Thus, if class LP consists of LDEVs 6, 11, and 19:

```
OPENQ 6  opens spool queues for LDEV 6
OPENQ LP  opens spool queues for LDEVs 6, 11, and 19
```

Related Information

Commands	STOPSPOOL, SHUTQ, SPOOLER
Manuals	*Performing System Operation Tasks* (32650-90137)

OPTION

Modifies the runtime environment of user-defined commands and command files. It is used within the body of a user command to set up and change the environment dynamically. (Native Mode)

NOTE Be sure to distinguish between the `OPTION` command and `OPTION` used in the header of a user command.

The `OPTION` command (described here) accepts only the `LIST/NOLIST` and `RECURSION/NORECURSION` parameters. `OPTION` used in the header of a UDC or a command file accepts the `HELP/NOHELP`, `LOGON/NOLOGON`, `BREAK/NO BREAK`, and `PROGRAM/NOPROGRAM` parameters, in addition to the `LIST/NOLIST` and `RECURSION/NORECURSION` parameters.

Syntax

```
OPTION[ {   LIST   NOLIST } ] [ ,] [ {   RECUR   NORECURSION } ]
```

Parameters

`LIST` Displays the command lines in a user command (UDC or command file) before each command in the user command is executed.

`NOLIST` Suppresses the display of the command lines in a user command when it is executed. `NOLIST` is the default.

`RECURSION` Begins the search for UDCs at the beginning of the cataloged commands list. `RECURSION` and `NORECURSION` do not have any meaning in a command file, because command files are not cataloged.

`NORECURSION` Begins the UDC search at the command currently executing and continues, in order, through the UDC catalog, as in MPE V/E. Default. `RECURSION` and `NORECURSION` do not have any meaning in a command file, because command files are not cataloged.

NOTE `OPTION` values are set to defaults whenever a command file or UDC is executed. If `OPTION` is specified as part of the user command definition then `OPTION` will be reset to this value if another UDC or command file is called from the user command. If `OPTION` is not set in the header of a UDC or command file then it's value will not be retained across calls to other UDC/command files.

Operation Notes

The `OPTION` command modifies the environment of user-defined commands (UDCs) and command files, giving users more flexibility in modifying the user command environment. When `OPTION` appears in a user command header, it is static and affects the entire command.

The LIST/NOLIST option specifies whether command lines in a UDC are printed before execution of each command. RECURSION/ NORECURSION determines the search order for commands cataloged.

RECURSION starts the UDC search at the beginning of the cataloged commands. NORECURSION, the default setting, starts the search at the command currently executing. RECURSION and NORECURSION do not have any meaning in a command file, because command files are not cataloged. The default is NORECURSION.

Nesting of IF and WHILE blocks in UDC's is limited to a combined total of 30 levels. Each IF or WHILE block read by the Command Interpreter increments the nesting count even if it resides within a *different* UDC. It is especially important to remember this when using the *recursion* option which may make it easy to increment the nesting count beyond 30.

Use

This command is available in a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

To send a line-by-line listing of the command file to $STDLIST as it executes, within the command file, enter:

```
OPTION LIST
```

Related Information

Commands	SETCATALOG, SHOWCATALOG, UDC header for static options
Manuals	None

OUTFENCE

Defines the minimum priority that an output spoolfile needs in order to be printed. (Native Mode)

Syntax

OUTFENCE ***outputpriority***[;LDEV=*ldev*] [;DEV={ *ldev* *devclass* *devname* }]

Parameter

outputpriority A number between 1 and 14, inclusive. A larger number is more limiting.

ldev The logical device number of an output device.

devclass A device class containing at least one output spoolable device. The *devclass* parameter must begin with a letter and consist of eight or fewer alphanumeric characters.

devname The name of the spooled device. The *devname* parameter must begin with a letter and consist of eight or fewer alphanumeric characters. Note that it is not possible to have a device class name and a device name that are the same. If you enter an alphanumeric character string, the command searches the device class list first, and then the device name list.

Operation Notes

This command 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 is printed or not. Individual output spoolfiles that are in the `READY` state are printed only if their *outputpriority* is higher than the current outfence. To prevent any spoolfiles from being printed, set the outfence to 14. To prevent a subset of spoolfiles from printing, set the outfence higher than the *outputpriority* of any spoolfile in the group.

To alter the printing priority of a single file without affecting the entire system, change the output priority of the specific spoolfile(s) with the `ALTSPOOLFILE` or `SPOOLF` command.

Notice that a device-specific outfence takes precedence over the system-wide (global) outfence, as seen in the example below.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It is executable only from the console unless distributed to users with the `ALLOW` command.

Examples

To defer all output spoolfiles except those waiting to be printed by LDEV 6, which is usually configured as 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 LISTSPF or SHOWOUT command, as in the example below. Note that the summary statistics at the bottom of the listing immediately reflects the new outfence. Once any currently ACTIVE spoolfile is finished, no files directed toward a device other than LDEV 6 can become ACTIVE.

```
SHOWOUT

DEV/CL   DFID  JOBNUM FNAME    STATE  FRM  SPACE  RANK  PRI
6     #O999  #J19  $STDLIST  OPENED    512      8
6     #O1030 #S77  EDLIST   OPENED    512      8
SLOWLP   #O1029 #S71  OUT     READY     232   D   7
LP     #O1001 #J60  $STDLIST  OPENED
11     #O1022 #S33  GALLIST   READY     768   D   7

5 FILES:
  0 ACTIVE
  2 READY; INCLUDING 2 SPOOFLES, 2 DEFERRED
  3 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
```

Related Information

Commands	ALTSPOOLFILE, LISTSPF, SHOWIN, SHOWOUT, SPOOLER, SPOOLF
Manuals	*Performing System Operation Tasks* (32650-90137)

6 Command Definitions P-R

PASCAL

Compiles a compatibility mode Pascal/V program. Pascal/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. The native mode equivalent of this command is PASXL.

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. $STDIN is the current input device, usually your terminal.

uslfile Actual file designator of the user subprogram library (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/iX 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/iX 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. $STDLIST is usually the terminal if you are running Pascal/V interactively, or the printer if you are running a batch job.

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 or two double quotation marks 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/V brackets the *quotedstring* with dollar signs and places it before the first line of source code in the text file.

Operation Notes

The PASCAL command compiles a compatibility mode Pascal/V program and stores the object code in a user subprogram library (USL) file on disk. If *textfile* is not specified, MPE/iX expects the source program to be entered from your standard input device. If you do not specify *listfile*, MPE/iX sends the program listing to your standard list device and identifies it by the formal file designator, PASLIST.

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.

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.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles a Pascal/V 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/V program contained in the disk file PASCSRC, and stores the object code in the USL file PASCOBJ. The program listing is stored in the disk file LISTFILE.

```
PASCAL PASCSRC,PASCOBJ,LISTFILE
```

Related Information

Commands	PASCALGO, PASCALPREP, PASXL, PASXLGO, PASXLLK PREP, RUN, LINK, LINKEDIT
Manuals	*MPE Segmenter Reference Manual* *HP Pascal/iX Reference Manual*

PASCALGO

Compiles, prepares, and executes a compatibility mode Pascal/V program. Pascal/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. The native mode equivalent of this command is PASXLGO.

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 $STDIN. $STDIN is the current input device, usually your terminal.

PASTEXT 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 FILE command.

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. $STDLIST is usually your terminal if you are running Pascal/V interactively, or the printer if you are running a batch job.

PASLIST 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 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 or two double quotation marks in a row is considered part of the string, and, therefore, not the terminating delimiter.

INFO=*quotedstring* is used in the Pascal/V programming language to pass initial compiler options to a program. Pascal/V brackets the *quotedstring* with dollar signs and places it before the first line of source code in the text file.

Operation Notes

The PASCALGO command compiles, prepares, and executes a compatibility mode Pascal/V program. If *textfile* is omitted, MPE/iX expects input from your standard input device. If you do not specify *listfile*, MPE/iX 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 MPE 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.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile, prepare, and execute a Pascal/V 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/V program from the disk file PASCSRC and send the program listing to the file LISTFILE, enter:

```
PASCALGO PASCSRC,LISTFILE
```

Related Information

Commands	PASCAL, PASCALPREP, PASXL, PASXLGO, PASXLLK PREP, RUN, LINK, LINKEDIT
Manuals	*MPE Segmenter Reference Manual*
	HP Pascal/iX Reference Manual

PASCALPREP

Compiles and prepares a compatibility mode Pascal/V program. Pascal/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. The native mode equivalent of this command is PASXLLK.

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. $STDIN is the current input device, usually your terminal.

PASTEXT 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 FILE command.

progfile Actual file designator of the program file to which the prepared program segments are written. When *progfile* is omitted, the MPE segmenter creates the program file, which is 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/iX 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. $STDLIST is usually your terminal if you are running Pascal/V interactively, or the printer if you are running a batch job.

PASLIST 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 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 or two double quotation marks 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/V brackets the *quotedstring* with dollar signs and places it before the first line of source code in the text file.

Operation Notes

The PASCALPREP command compiles and prepares a compatibility mode Pascal/V program into a program file on disk. If you do not specify *textfile*, MPE/iX expects input from the current input device. If you do not specify *listfile*, MPE/iX 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 MPE segmenter. You may access it only if you do not use the default for *progfile*. This is because the MPE segmenter also uses $OLDPASS to store the prepared program segments, overwriting any existing temporary file of the same name.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles and prepares a Pascal/V program entered through your standard input device, and stores the prepared program segments in the file $OLDPASS. The listing is printed on your standard list device.

```
PASCALPREP
```

To compile and prepare a Pascal/V source program from the source file PASCSRC, store it in PASCPROG, and send the listing to your standard list device, enter:

```
PASCALPREP PASCSRC,PASCPROG
```

Related Information

Commands	PASCALGO, PASCAL, PASXL, PASXLGO, PASXLLK PREP, RUN, LINK, LINKEDIT
Manuals	*MPE Segmenter Reference Manual*
	Pascal/3000 Reference Manual

PASSWORD

Creates or changes a user password. (Native Mode)

Syntax

PASSWORD

Parameters

None.

Use

This command may be issued from a session or in BREAK. It is breakable (aborts execution). It cannot be used if $STDIN or $STDLIST are redirected.

Operation

This command allows users to establish or change their own passwords. It may be issued interactively or programmatically within a session and prompts the user for required input. Passwords are not echoed (displayed) during input.

Example

```
PASSWORD
ENTER OLD USER PASSWORD:
ENTER NEW USER PASSWORD:
ENTER NEW USER PASSWORD AGAIN:
PASSWORD WAS CHANGED SUCCESSFULLY.
```

The old user password is requested only if it exists.

Related Information

Commands	LISTUSER, ALTUSER
Manuals	None

PASXL

Compiles an HP Pascal/iX program. HP Pascal/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. (Native Mode)

Syntax

PASXL[*textfile*] [,[*objectfile*] [,[*listfile*] [*,libfile*]]] [;INFO=*quotedstring*]

Parameters

textfile The name of the text file that contains the source code to be compiled. This is an ASCII file that you prepare with an editor such as EDIT/V. The formal file designator is PASTEXT.

If you are running HP Pascal/XL from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile*, then the default file is $STDIN. $STDIN is the current input device, usually your terminal.

When *textfile* is your terminal, you can enter source code interactively in response to the > prompt. When you have entered all the source code, type a colon (:) in response to the > prompt to end the interactive input.

The source code to be compiled can be a program or a list of modules.

objectfile Actual file designator of the object file to which the object code is stored. This file is stored in binary form and has a file code of either (1461) or NMRL (1033). Its formal file designator is PASOBJ. If the *objectfile* parameter is omitted, the object code is saved to the temporary file $OLDPASS.

If you specify *objectfile*, the compiler stores the object file in a permanent file of the correct size and type, and with the name you specified. If a file of the same name already exists, the object code overwrites that file.

If the compiler issues an error message telling you that a new or existing object file you are trying to compile to is too small, build the object file with a larger size and recompile to it.

You may use the MPE/iX SAVE command to store $OLDPASS as a permanent file under another name.

listfile The name of the file on which the compiler writes the program listing. It can be any ASCII file. The default is $STDLIST. $STDLIST is usually the terminal if you are running HP Pascal/iX interactively, or the printer if you are running a batch job. The formal file designator is PASLIST.

If your terminal is both *textfile* and *listfile*, the compiler does not write the program listing on the terminal.

If *listfile* is $NULL or a file other than $STDLIST, the compiler displays on $STDLIST those lines that contain errors.

libfile The name of the HP Pascal/iX library file that the compiler searches if a search path is not specified with the compiler option SEARCH. The default is PASLIB in your group and account.

quotedstring A string of no more than 132 characters (including the single or double quotation marks that enclose it).

The *quotedstring* string is used in the HP Pascal/iX programming language to pass initial compiler options to the compiler. HP Pascal/iX brackets the *quotedstring* string with dollar signs ($) and places the string before the first line of source code in the text file.

NOTE The formal file designators used in this command (PASTEXT, PASOBJ, PASLIST, and PASLIB) 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.

Operation Notes

The PASXL command compiles an HP Pascal/iX program and stores the object code in a permanent file (*objectfile*) or in $OLDPASS if you do not specify an object file. If *textfile* is omitted, the compiler expects the source program to be entered from your standard input device. If you do not specify *listfile*, the compiler sends the program listing to the formal file designator PASLIST (default is $STDLIST).

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles an HP Pascal/iX program entered from your standard input device and stores the object program in the object file $OLDPASS. The listing is then sent to your standard list device.

```
PASXL
```

The next example compiles an HP Pascal/iX program contained in the disk file SOURCE and stores the object program in the object file OBJECT. The program listing is stored in the disk file LISTFILE.

```
PASXL SOURCE,OBJECT,LISTFILE
```

NOTE Program development in native mode uses the MPE/iX LINK command not the MPE V/E PREP command. This produces a significant change in the method of linking code.

If you have created a program called MAIN and a subprogram called SUB, each contained in a separate file, you might choose to handle it this way in MPE V/E:

```
 PASCAL MAIN, SOMEUSL
 PASCAL SUB, SOMEUSL
:
:
 PREP SOMEUSL, SOMEPROG
:
 RUN SOMEPROG
```

The second command appends the code from SUB to SOMEUSL.

However, LINK (in MPE/iX native mode) does not append SUB. In MPE/iX, you must compile the source files into separate object files and then use the Link Editor to link the two object files into the program file, as in this example:

```
 PASXL MAIN, OBJMAIN
 PASXL SUB, OBJSUB
:
 LINK FROM=OBJMAIN,OBJSUB;TO=SOMEPROG
:
 RUN SOMEPROG
```

However, if an NMRL is used instead of an NMOBJ, the above can be simplified to the following:

```
BUILD RLFILE;DISC=10000;CODE=NMRL
PASXL MAIN, RLFILE
PASXL SUB, RLFILE
LINK RLFILE,SOMEPROG
RUN SOMEPROG
```

Related Information

Commands PASCALGO, PASCALPREP, PASCAL, PASXLGO, PASXLLK PREP, RUN, LINK, LINKEDIT

Manuals *HP Pascal/iX Reference Manual*

HP Link Editor/XL Reference Manual

PASXLGO

Compiles, links, and executes an HP Pascal/iX program. HP Pascal/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. (Native Mode)

Syntax

PASXLGO[*textfile*] [,[*listfile*] [,[*libfile*]]] [;INFO=*quotedstring*]

Parameters

textfile The name of the text file that contains the source code to be compiled. This is an ASCII file that you prepare with an editor such as EDIT/V. The formal file designator is PASTEXT.

If you are running HP Pascal/iX from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile,* then the default file is $STDIN. $STDIN is the current input device, usually your terminal.

When *textfile* is your terminal, you can enter source code interactively in response to the > prompt. When you have entered all the source code, type a colon (:) in response to the > prompt to end the interactive input.

The source code to be compiled can be a program or a list of modules.

listfile The name of the file on which the compiler writes the program listing. It can be any ASCII file. The default is $STDLIST. $STDLIST is usually the terminal if you are running HP Pascal/iX interactively, or the printer if you are running a batch job. The formal file designator is PASLIST.

If your terminal is both *textfile* and *listfile,* the compiler does not write the program listing on the terminal.

If *listfile* is $NULL or a file other than $STDLIST, the compiler displays on $STDLIST those lines that contain errors.

libfile The name of the HP Pascal/iX library file that the compiler searches if a search path is not specified with the compiler option SEARCH. The default is PASLIB in your group and account.

quotedstring A string of no more than 132 characters (including the single or double quotation marks that enclose it).

The *quotedstring* string is used in the HP Pascal/iX programming language to pass initial compiler options to the compiler. HP Pascal/iX brackets the *quotedstring* string with dollar signs ($) and places the string before the first line of source code in the text file.

NOTE The formal file designators used in this command (PASTEXT, PASLIB, 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.

Operation Notes

The PASXLGO command compiles, links, and executes an HP Pascal/iX program. If *textfile* is omitted, the compiler expects input from your standard input device. If you do not specify *listfile*, the compiler sends the program listing to the formal file designator PASLIST (default is $STDLIST).

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. The Link Editor purges the object file and writes the linked program to $OLDPASS, which is then executed and may be executed repeatedly.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile, link, and execute an HP Pascal/iX program entered from your standard input device, with the program listing sent to your standard list device, enter:

```
PASXLGO
```

To compile, link, and execute an HP Pascal/iX program from the disk file SOURCE and send the program listing to the file LISTFILE, enter:

```
PASXLGO SOURCE,LISTFILE
```

Related Information

Commands	PASCAL, PASCALGO, PASCALPREP, PASXL, PASXLLK PREP, RUN, LINK, LINKEDIT
Manual	*HP Pascal/iX Reference Manual* (31502-90001)

PASXLLK

Compiles and links an HP Pascal/iX program. HP Pascal/iX is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. (Native Mode)

Syntax

PASXLLK[*textfile*] [,[*progfile*] [,[*listfile*] [,*libfile*]]] [;INFO=*quotedstring*]

Parameters

textfile The name of the text file that contains the source code to be compiled. This is an ASCII file that you prepare with an editor such as EDIT/V. The formal file designator is `PASTEXT`.

If you are running HP Pascal/iX from your terminal, you will probably specify a disk *textfile*. If you do not specify *textfile,* then the default file is `$STDIN`. `$STDIN` is the current input device, usually your terminal.

When *textfile* is your terminal, you can enter source code interactively in response to the > prompt. After you enter the source code, type a colon (`:`) in response to the > prompt to end the interactive input.

The source code to be compiled can be a program or a list of modules.

progfile The name of the program file on which the MPE/iX linker writes the linked program. The default is `$NEWPASS`.

listfile The name of the file on which the compiler writes the program listing. It can be any ASCII file. The default is `$STDLIST`. `$STDLIST` is usually the terminal if you are running HP Pascal/iX interactively, or the printer if you are running a batch job. The formal file designator is `PASLIST`.

If your terminal is both *textfile* and *listfile,* the compiler does not write the program listing on the terminal.

If *listfile* is `$NULL` or a file other than `$STDLIST`, the compiler displays those lines that contain errors on `$STDLIST`.

libfile The name of the HP Pascal/iX library file that the compiler searches if a search path is not specified with the compiler option `SEARCH`. The default is `PASLIB` in your group and account.

quotedstring A string of no more than 132 characters (including the single or double quotation marks that enclose it). The *quotedstring* is used to pass initial compiler options to the HP Pascal/iX compiler. HP Pascal/iX brackets the *quotedstring* with dollar signs ($) and places the string before the first line of source code in the text file.

NOTE The formal file designators used in this command (PASTEXT, PASLIB, 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.

Operation Notes

The PASXLLK command compiles and links an HP Pascal/iX program into a file on disk. If you do not specify *textfile*, the compiler expects input from the standard input device. If you do not specify *listfile*, the compiler sends the program listing output to the formal file designator PASLIST (default $STDLIST).

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. Link Editor overwrites *progfile* and writes the linked program to $OLDPASS, if *progfile* is omitted, which can then be executed.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles and links an HP Pascal/iX program entered through your standard input device and stores the linked program in the file $OLDPASS. The listing will be printed on your standard list device.

```
PASXLLK
```

To compile and link an HP Pascal/iX source program from the source file SOURCE, store it in PROG, and send the listing to your standard list device, enter:

```
PASXLLK SOURCE,PROG
```

Related Information

Commands	PASCAL, PASCALGO, PASCALPREP, PASXL, PASXLGO, PREP, RUN, LINK, LINKEDIT
Manuals	*HP Pascal/iX Reference Manual*
	HP Pascal/iX Programmer's Guide
	HP Link Editor/XL Reference Manual

PAUSE

The PAUSE command allows the current task to be suspended or “sleep” for a specifiec number of seconds.)

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Syntax

PAUSE [*num_seconds***]**
[[;JOB=]*jobid***]**
[[;INTERVAL=] *interval_secs***]**
[;EXIST | WAIT | NOTEXIST]

Parameters

Collectively **EXIST, WAIT** and **NOTEXIST** are referred to as the "while_state", since **PAUSE** sleeps "while" the specified state is true.

num_seconds If *num_seconds* is specified without *jobid* **PAUSE** sleeps for that many seconds, or until the process issuing the pause is interrupted by the break signal. If "jobid" is also supplied then "num_seconds" has a different meaning. In this case it indicates the maximum duration for the PAUSE command, such that **PAUSE** should continue while the selected jobs are in their "while_state" or when *num_seconds* has expired, whichever is shortest. Thus, *num_seconds* represents the maximum length of the pause. If **PAUSE** completes but one or more jobs are still in their "while state" a CIWARN is reported.

NOTE to pause while a job is in its "while_state" or until *num_seconds* has expired, whichever is LONGEST, one can execute the following two commands:

PAUSE x

PAUSE job=y ;z

If after X seconds job Y is still in state Z then the second **PAUSE** continues while state Z applies. On the other hand, if after X seconds job Y is not in state Z then the pause is complete. or equal to zero.

jobid can be one of: [#]Jnnn, [#]Snnn, [*jobname*,]*user.acct*, @, @J, @S. Note if *jobname* is included than the *jobid* must be quoted since the comma is a command token delimiter.

If the JOB= parameter is specified then **PAUSE** sleeps while *jobid* is in its "while_state". *jobid* can be an executing, waiting, scheduled job, or a session. *jobid* can also name many jobs or sessions. Wildcarding is supported, and a non-wildcarded [*jname*,]*user.acct* can match several jobs or sessions. The job name value can be " ," or " @," to match all jobs or sessions without a job name. When more than one job or session matches *jobid* **PAUSE** sleeps while all matching jobs are in their "while_state". If the job executing **PAUSE** matches *jobid* it will not be selected.

interval_secs — If *interval_secs* is specified **PAUSE** sleeps for this many seconds between attempts to see if *jobid* is still in its "while_state". Otherwise, **PAUSE** sleeps a variable amount of seconds depending on the job state and the number of previous times a particular job has been polled. This computed method favors executing jobs that terminate quickly.

EXIST — (default) means to pause while all jobs and sessions matching "jobid" exist. These jobs can be scheduled, waiting, executing, etc., but as long as the SHOWJOB command displays one or more of the jobs defined by "jobid", the pause continues

WAIT — means to pause while the selected job or jobs are waiting. As soon as all the matching jobs are no longer waiting (meaning all the job states are no longer "introduced", "waiting", or "scheduled") the pause ends. The life cycle of a job is typically: [sched or waiting->] intro-> initializing-> exec-> [susp-> exec->] terminate. Waiting jobs are considered all job states left of and excluding "initializing". Non-waiting jobs are all jobs right of and including "initializing"

NOTEXIST — means to pause while the matching job or jobs do not exist. As soon as any jobs matching "jobid" exist (in any state) the pause completes. PAUSE might miss finding jobs that log off quickly. This is particularly true for a match on a single job/session number. A more practical use might be:

```
PAUSE job=@J;notexist
```

which means to sleep while no jobs exist. As soon as the first job is streamed the above pause stops.

Operation Notes

The value of this command lies in providing a way to suspend one activity while another process waits for a specific condition to exist, for example, forcing a job to "idle" while waiting for the creation of a key file or the setting of a crucial flag. You may use several MPE/iX commands to query user or system variables, or the system itself, in order to verify the existence of the desired condition.

In its simpliest form, the PAUSE command sleeps for "num_seconds", or less if BREAK is pressed. In this simple case no "jobid" is specified and all other command arguments are ignored. If the "jobid" parameter is specified then "interval_secs" and the remaining command parameters are relevant. When "jobid" is supplied PAUSE typically sleeps until the jobs or sessions matching "jobid" have terminated.

Use

This command is available from a program or in BREAK. You can execute BREAK while PAUSE is active. BREAK terminates the pause.

Examples

If a job must read data from a file called LOGDAT.GXK.PROCCTRL, which is to be created by a session, then the job may suspend activity pending a test for the existence of the vital file.

The example below shows how the PAUSE command can be used to synchronize a session to some job activity via the existence of a known file:

```
STREAM JLOGEND
#J123
...
...
SETVAR START_CPU HPCPUSECS

WHILE NOT FINFO("LOGDAT.GXK.PROCCTRL","EXISTS") AND &
      HPCPUSECS-START_CPU <5 DO
 PAUSE 2
ENDWHILE
DELETEVAR START_CPU
```

NOTE The CPU seconds used by the WHILE loop is not allowed to exceed 5 seconds.

If the file does not exist and the WHILE loop has consumed less than five CPU seconds, then the job pauses for two seconds. This pause does not use CPU-time. The CPU check is included to prevent an infinite loop that may result if JLOGEND aborted unexpectedly and thus did not get a chance to build the LOGDAT file.

The following example pauses while job #J24 exists in the system job table, (JMAT) i.e., it is visible in **SHOWJOB** output.

```
:PAUSE job=#j24
```

The next example sleeps as long as MANGER.SYS has any jobs or sessions running or waiting.

```
:PAUSE job=manager.sys; exists
```

The next example pauses until the job just streamed starts executing.

```
:STREAM myjob
:PAUSE job=!hplastjob; wait
```

Or, sleeps until the job you just streamed completes.

```
:PAUSE , !hplastjob
```

The following example sleeps until all jobs have logged off or 5 minutes, whichever occurs first.

```
:PAUSE 300, @J
:IF hpcierr = -9032 then
```

# pause terminated but one or more jobs are still running

The next example pauses while all jobs (by naming convention only) in the PROD account are running.

```
 :PAUSE job="J@,@.PROD"
```

# note the quotes are required

The next example sleeps while the backup job ("JBACKUP,OP.SYS") has not been streamed. **PAUSE** reports CIWARN 9032 if the job is not streamed within 30 minutes.

```
:PAUSE 1800, job="jbackup,op.sys"; notexist
```

The final example polls the system job table every 3 minutes looking for any job or session matching a user name that includes the letters "MGR", and waits for all such job/sessions to terminate before the pause ends.

```
:PAUSE , @mgr@.@ , 180
```

Related Information

Commands	`WHILE`, `INPUT, SHOWJOB`
Manuals	None

PLISTF (UDC)

The PLISTF UDC executes the LISTFILE command to list descriptions of one or more disk files.

System-defined UDCs are not automatically available. Your System Manager must use the SETCATALOG command to make these UDCs available for your use. For example:

```
SETCATALOG HPPXUDC.PUB.SYS;SYSTEM;APPEND
```

Syntax

PLISTF[*fileset*] [,*format_opt*] [;*outfile*]

Parameters

The following parameters are supported with the PLISTF UDC. Refer to the LISTFILE command for a complete explanation of the parameters used with the PLISTF UDC.

fileset Specifies a set of files to be listed, including MPE and HFS files. If *fileset* is not specified, the default is @.

format_opt An output format option. If this parameter is omitted, the default is FORMAT=0, which shows only the file names. The format option must be specified as a numeric value. Format names (for example, QUALIFY) are not supported by this UDC. Refer to the LISTFILE command for a complete description of each available format option.

outfile The name of the output file. If this parameter is omitted, the output is displayed to $STDLIST. The *outfile* supports both MPE and HFS syntax. The *outfile* cannot be $NEWPASS.

Operation Notes

The PLISTF UDC lists descriptions of one or more disk files at the level of detail you select. The UDC executes the following form of the LISTFILE command:

LISTFILE ***fileset* , *format_opt***[>*outfile*]

Use

This UDC may be issued from a session, a job, a program, or in break mode. Pressing **Break** aborts execution.

If a permanent file exists with the same name as specified as *outfile*, then CIOR defaults are used rather than the PLISTF CCTL default.

Examples

Refer to the LISTFILE command earlier in this chapter for examples.

Related Information

Commands	`LISTF`, `LISTFILE`, `LISTDIR` (UDC), `FINDFILE` (UDC), `FINDDIR` (UDC)
Manuals	None

PREP

Prepares a compatibility mode 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*] [;DL=*dlsize*]

[;NOSYM] [{;FPMAP | ;NOFPMAP}]

Parameters

uslfile Actual file designator of user subprogram library (USL) file into which the 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/iX 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 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 of what you specify, MPE/iX 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 MPE 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 is 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. Specify a value between -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   =   Extra Data Segments
MR   =   Multiple RINs
PM   =   Privileged Mode
```

You can only specify those capabilities 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 applies 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/V 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`.

Operation Notes

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/iX creates a file of the optimum size and characteristics. (Refer to the "Examples" 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.

NOTE The MPE segmenter employs temporary files named T999SYM, SEGTMP01, and SEGTMP00. If you have created temporary files having these names, the segmenter attempts to purge them.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

In the following example, you use the PREP command to prepare a program from the USL file USLX and the MPE segmenter stores it in the program file PROGX. Since the MPE segmenter creates PROGX for you, it is a temporary file, and you must subsequently save it in the permanent file domain.

```
PREP USLX,PROGX
SAVE PROGX
```

Although you will get the best results by having the MPE segmenter create the program file for you, you can also 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
```

Related Information

Commands	PREPRUN, RUN
Manuals	*MPE Segmenter Reference Manual*

PREPRUN

Prepares and executes a compiled compatibility mode program.

Syntax

```
PREPRUN uslfile[,entrypoint]
[;NOPRIV] [;PMAP] [;NOCB] [;DEBUG] [;INFO=quotedstring]
[;LMAP[;STDIN [{*formaldesig =fileref $NULL}]]]
[;MAXDATA=segsize] [;PARM=parameternum] [;STDLIST=[ { *formaldesig fileref[ ,NEW] $NULL }]]
[;STACK=stacksize][ ;DL=dlsize] [;PATCH=patchsize]
[ ;LIB={ G | P | S}]
[;NOSYM] [{;FPMAP | ;NOFPMAP}] [;CAP=capabilitylist]
```

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 are to be placed in nonprivileged (user) mode. This parameter is for programs prepared with privileged mode (PM) capability and makes them accessible to nonprivileged users. Normally, program segments containing privileged instructions are executed in privileged mode only if the program was prepared with privileged mode 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 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 of what you specify, MPE/iX 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 MPE 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, the account library, and 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.

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	=	Extra Data Segments
MR	=	Multiple RINs
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 does, 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 quotation marks, or two double quotation marks in a row, is considered part of the string, and, therefore, not the terminating delimiter.

The INFO=*quotedstring* parameter is used in some programming languages (for example, COBOLII, Pascal) to pass compiler options to a program. These options appear before the first line of source code in the text file.

$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 disk 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/iX, 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 disk 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 applies to all segments within the program file. The value specified must be within -1 and 16,380 words.

NOSYM Suppresses the symbolic DEBUG option. Refer to the *HPToolset/V Reference Manual* 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.

Operation Notes

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 MPE 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.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering RESUME continues the execution.

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 from 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 disk 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 disk 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 appears on the printout as follows:

```
A TEST WITH "AND" CHARACTERS
```

Related Information

Commands	PREP, RUN, XEQ
Manuals	*MPE Segmenter Reference Manual*

PRINT

Prints the contents of a file.

Syntax

PRINT *filename* [OUT=*outfile*] [START=*m*] [END=*n*] [PAGE=*p*] [;UNN | NUM] [;NONUM]

Parameters

filename — Actual file name of the file to be printed to $STDLIST, unless *outfile* is specified as a destination. To specify an HFS file, begin the filename with a dot (.) or slash (/). The *filename* may specify either a temporary or a permanent disk file.

File equations are ignored unless an asterisk (*) precedes *filename*, indicating a backreference.

The *filename* may be $STDIN or $STDINX.

If you do not specify a file name, PRINT takes its input from $STDINX and continues to do so until you enter the :EOD command on a new line.

outfile — Specifies a destination other than $STDLIST for *filename*. If *filename* has embedded carriage-control characters (CCTL), PRINT inserts a blank in place of the CCTL in the *outfile*. New files are created TEMP. File equations are ignored unless an asterisk (*) precedes *outfile*, indicating a backreference. You must use a file equation to overwrite a permanent file.

You must use the ;SAVE option in the file equation to overwrite a permanent file.

If *outfile* is not interactive with the user's $STDIN file, the PAGE parameter is ignored. (Refer to the FRELATE intrinsic for additional information on "interactive pair" of files.)

To redirect output to the line printer (DEV=LP), you could use the following commands:

```
FILE PRT;DEV=LP;CCTL
PRINT MYFILE;OUT=*PRT
```

Specifies the record number of the first file record to be displayed. An *m* is relative to 1. If *m* is a negative number, it specifies a record location relative to the end-of-file, that is, -5 indicates the fifth record from the end-of-file. Zero is an invalid specification. Default is the first record of the file.

For byte-stream files, the first line (or "record") corresponds to the bytes from the beginning of the file to the first newline character, the second line contains bytes between the first newline character and the second newline character, and so on.

n Specifies the last record of the file to be displayed. An *n* is relative to 1. If *n* is a negative number, it specifies a location relative to the end-of-file, that is, -5 indicates the fifth record from the end-of-file. Zero is an invalid specification. Default is the last record of the file.

NOTE For byte stream files, you cannot display one or more records by specifying a negative number with the keywords `START=` or `END=`. If you try to do so, the result will be unpredictable because the end-of-file for byte stream files is the total byte count of the file, and not the number of the last record.

p Specifies the number of lines to be displayed before a page break occurs. Default is 23 lines for interactive users and 0 (continuous) for non-interactive users (that is, in a job). Specifying 0 for *p* suppresses page breaks in the output and produces continuous output from the beginning to the end of the file.

If *filename* contains more than *p* records and you are working interactively, the command displays *p* lines and then prompts you for a reply indicating whether or not more output is desired, as follows:

```
(NEXT/EOF) CONTINUE?
```

`NEXT` is the next record number to be printed, and `EOF` is the end-of-file value that would be displayed by `LISTF` <*filename*>`,``2`. If you are reviewing a byte-stream file, `NEXT` displays the next logical record, whereas `EOF` is the byte count of the file.

Table 6-1 defines the range of valid responses to control the output.

Table 6-1 PRINT Command Control

Response	Result
`Y, Yes`	Continue printing at record *next*
`N, NO`, **Break**	Stop printing
-*m* (integer)	Continue printing at record *next*-*m*
+*m* (integer)	Continue printing at record *next*+*m*
m (integer)	Continue printing at record *m*
other, **Return**	Continue printing at record *next* (default)

Responses are case insensitive. Note that **Return** instructs `PRINT` to continue printing.

In jobs, no prompt for continuing output is generated. Instead, a page-eject control character is written to outfile every *p* lines. A page value of zero suppresses all page breaks, and *filename* is printed from *m* through *n*, inclusive. This is the default for jobs.

`UNN` Suppresses line numbering in the display, regardless of whether the disk file is numbered or unnumbered. `UNN` is the default.

NUM Specifies numbering of the lines as they are displayed. The numbers appear in front of the line (record) being displayed. The number displayed is the actual line number for numbered files; for unnumbered files, relative numbering begins with 1.

NONUM Requests that trailing digits at the end of each record in the file be displayed as part of the file content, rather than being interpreted as line numbers.

Operation Notes

This command prints the contents of *filename* to the standard list device, unless another destination is specified with the *outfile* variable.

If an interactive user takes more than HPTIMEOUT minutes to respond to the page number prompt, MPE/iX terminates the CI. This occurs only if HPTIMEOUT has been set to a positive value.

In a batch job, in which the *filename* defaults to $STDINX, some MPE/iX commands such as :EOD, EOF, JOB, EOJ, and DATA do not execute as part of the original job when they follow a PRINT command. For example, if a JOB command follows a PRINT command, only those commands preceding PRINT are executed in the original job, and nothing is printed. The JOB command following the PRINT command is taken as the start of a new job, which is then streamed as a second job.

Use

This command is available in a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command.

Examples

To send the contents of MYFILE to the line printer, enter the following commands:

```
FILE XXX;DEV=LP
PRINT MYFILE, *XXX
```

In this example, the file XXX is equated with the line printer. Then the file MYFILE is "printed" to the file *XXX.

Use EDIT/V to create the command file TAIL which prints the last 10 lines of a file:

```
PARM FILE, LAST=10
PRINT !FILE; START = -!LAST
```

The first line defines FILE as a required parameter of the command file and creates an optional parameter, LAST the default value of which is 10.

The second line instructs the PRINT command to print the dereferenced *value* of FILE (the value entered by the user). The second line also tells the command to use the negative of the dereferenced value of LAST (10 by default) as the starting point for printing (that is, 10 records from the end).

To print the last 10 records of the file called MYFILE, enter:

```
TAIL MYFILE
```

To print the last 45 records of MYFILE, because entering the value 45 overrides the default value of 10, enter:

```
TAIL MYFILE, 45
```

NOTE The PRINT command itself can be used to create a file:

```
PRINT $STDIN,TAILB
PARM FILE, LAST=5
PRINT !FILE; START = -!LAST
:EOD
SAVE TAILB
```

The SAVE command is used to make the file TAILB permanent since the default is temporary.

PRINT infile;NUM;NONUM

would print the line numbers as in the case of UNNUMbered files, ie, line numbers starting from 1 for the first record and so on.

PRINT infile;UNN;NONUM

PRINT infile;NONUM

would consider the file as UNNUMbered file even when the file is a NUMbered file and the print the contents as it is in the file.

[UFILEYES is an unnumbered file with trailing 8 characters as digits.]

```
PRINT UFILEYES
```

```
aaaaaaaaaaaa
bbbbbbbbbbbb
cccccccccccc
dddddddddddd
eeeeeeeeeeee
ffffffffffff
gggggggggggg
hhhhhhhhhhhh
iiiiiiiiiiii
jjjjjjjjjjjj
kkkkkkkkkkkk
llllllllllll
```

NOTE The above file was considered by PRINT to be a numbered file and thus the trailing 8 bytes are truncated

PRINT UFILEYES;NONUM

```
    aaaaaaaaaaaa00010001
    bbbbbbbbbbbb00010002
    cccccccccccc00010003
    dddddddddddd00010004
    eeeeeeeeeeee00020001
    ffffffffffff00020002
    gggggggggggg00020003
    hhhhhhhhhhhh00020004
    iiiiiiiiiiii00030001
    jjjjjjjjjjjj00030002
    kkkkkkkkkkkk00030003
    llllllllllll00030004
```

HFS Example

The following command entry will print the last 10 records of the file called `posix/doc/print.doc` in the current working directory (CWD).

```
  PRINT ./posix/doc/print.doc;start=-10
```

Related Information

Commands	FCOPY, COPY
Manuals	None

PURGE

This command deletes one or more files from the system.

Syntax

```
PURGE filereference
    [ ;TEMP] [ [ ;ONERROR=] {   CONTINUE   QUIT } ]
     [{   ;AUTOLOCKWORD    ;NOAUTOLOCKWORD } ]
    [{   ;CONFIRM   ;NOCONFIRM   ;CONFIRMALL } ]
     [{   ;NOSHOW   ;SHOW } ] [ {   ;SHOWERRORS   ;NOSHOWERRORS } ]
```

Parameters

filereference The actual file designator of the file to be deleted, interpreted according to MPE-escaped semantics *filereference*, can be either an MPE file (i.e., one that uses MPE syntax) or it can be a POSIX file name beginning with a dot or a slash. For example, you can use the escaped pathname `/SYS/PUB/FILE` since it is equivalent to the MPE name `FILE.PUB.SYS`.

`TEMP` Specifies that the file is a temporary file in the job/session temporary file domain. You can specify a *filename* in MPE or HFS syntax and may name a symbolic link that resolves to a *filename*. You must enter this parameter to delete a temporary file. The default is that a permanent file is assumed.

`CONTINUE` Allows `PURGE` to continue until the end of the list is reached, regardless of errors. `CONTINUE` is the default option.

`QUIT` Quits the execution of `PURGE` when it encounters an error and sets the CIERROR variable to the last execution error.

`AUTOLOCKWORD` Directs `PURGE` to look up and resolve file lockwords automatically. Users with system manager (SM) capability can specify `AUTOLOCKWORD` for all files on the system. Users with account manager (AM) capability can specify `AUTOLOCKWORD` for all files within their account.

`NOAUTOLOCKWORD` Requires the user to specify a file's lockword before the file is purged. This is the default.

`CONFIRM` Verifies the *filereference* parameter by requiring you to validate the purge during command execution. Valid responses are "YES" or "NO". If you respond "YES", the `PURGE` command is executed. Pressing **Break** at the prompt is equivalent to responding "NO". `CONFIRM` is the default for sessions, unless the *filereference* designates a single file.

`NOCONFIRM` Continues the purge without verification from the user. `NOCONFIRM` is the default for jobs or if the *filereference* designates a single file.

`CONFIRMALL` Requests verification for each file before the purge is executed. A proper response is one of the following:

- "Y" or "YES" to purge the file

- "N", "NO", or **Return** to retain the file
- "Q", "QUIT", or **Break** to stop the PURGE command

The CONFIRMALL option is ignored in jobs and when you are purging a single file.

NOSHOW Suppresses the display of each successfully purged file. NOSHOW is the default.

SHOW Displays the name of each successfully purged file.

SHOWERRORS Displays each lower-level error which prevents a file from being deleted. The name of the file is shown, followed by the error message. By default lower-level errors are not displayed. You may also enter this option in the singular form, i.e. SHOWERROR.

NOSHOWERRORS Suppresses the display of low-level errors. NOSHOWERRORS is the default. You may also enter this option in the singular form, i.e. NOSHOWERROR.

Operation Notes

- **Usage**

  You can enter this command from a session, a job, a program, or in break mode. Pressing **Break** does not affect this command.

  You must have write access to a file to delete it.

- **Purging unrecognized files**

  If the file does not exist in the specified domain, the following message appears:

  ```
  FILE filename NOT FOUND, NO PURGE DONE. (CIWARN 383)
  ```

- **Purging non-private spool files**

  You can purge a non-private spool file by entering PURGE *filename*. You must specify the fully qualified file name (including .OUT.HPSPOOL). The PURGE command deletes the specified spool file and all links to the spool file directory. The spool file does not print after you purge it.

- **Purging files with wildcards**

  You can use wildcards to remove multiple files at once. You can also use the CONFIRMALL option to prevent accidental deletion of one or more files. Examples of the wildcard feature are listed in the Examples section below:

Examples

- To delete a permanent file named PFILE, enter:

  ```
  :PURGE PFILE
  ```

- To purge multiple files using wildcards

```
:PURGE /users/jeff/bin/FILES/file@
3 FILES matched
Continue PURGE? (YES/NO) yes
3 selected.  3 succeeded.  0 failed.
```

- To purge multiple files interactively using wildcards

  To purge a number of files, one at a time, in an interactive mode so that you can skip a file or stop your purge, you can use the CONFIRMALL option.

  ```
  :PURGE /users/jeff/bin/FILES/file@; CONFIRMALL
  3 FILES matched
  /users/jeff/bin/FILES/file1 ? (NO/YES/QUIT) yes
  /users/jeff/bin/FILES/file2 ? (NO/YES/QUIT) no
  /users/jeff/bin/FILES/file3 ? (NO/YES/QUIT) yes
  2 selected.  2 succeeded.  0 failed.
  ```

  Type "q","quit", or press the **BREAK** key if you decide to stop the PURGE command completely.

- To purge log files using wildcards

  The following example shows you how to purge all log files within your current working directory that start with log, followed by any number from 0 - 9 (#), followed by any number of alphanumeric characters (@).

  ```
  :PURGE log#@
  10 FILES matched
  Continue PURGE?  (YES/NO) yes
  10 selected.  9 succeeded.  1 failed.
  ```

  Since the PURGE command does not remove the currently opened log file, the command always returns "1 failed".

Related Information

Commands	ALTSEC, BUILD, LISTFILE, LISTSPF
Manuals	None

PURGEACCT

Removes an account and its groups and users from the system directory or from the specified volume set's directory.

Syntax

PURGEACCT ***acctname***[;ONVS=*volumesetname*]

Parameters

acctname Name of the account to be deleted. This name must contain from one to eight alphanumeric characters, beginning with an alphabetic character.

volume- setname The volume set from which the account is to be purged. Volume set names consist of from 1 to 32 characters, beginning with an alphabetic character. The remaining characters may be alphabetic, numeric, the underscore, and periods.

If you specify a *volumesetname*, you must specify the full name of the volume set

The *volumesetname* specified refers to a previously defined volume set. When a *volumesetname* is specified, the volume set must be mounted, or the PURGEACCT command fails. When ONVS=*volumesetname* is specified, the account is removed from the volume set directory. When ONVS= is specified without *volumesetname*, the account is removed from the system directory.

Refer to the VSxxxxxx commands in this chapter.

Operation Notes

The system manager uses the PURGEACCT command to eliminate an entire account from the system. When PURGEACCT is executed during a session, MPE/iX 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 first, then 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 and 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 SYS account cannot be completely purged, but you can destroy critical files by attempting to do so. If you execute PURGEACCT SYS, all groups except PUB are purged; all users except the system manager are purged; and all inactive files and system files in the PUB group are purged.

NOTE If you specify volume-related commands or parameters for a volume set that is not currently mounted, or for an account that does not exist, MPE/iX returns an error message.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. System manager (SM) capability is required to execute this command.

Examples

To remove an account named ACCT1, enter:

```
PURGEACCT ACCT1
ACCT ACCT1 TO BE PURGED? YES
:
```

To purge the account FARFLE from the volume set TIME_LORD, you need to issue two commands:

```
PURGEACCT FARFLE
ACCT FARFLE TO BE PURGED? YES
:
PURGEACCT FARFLE;ONVS=TIME_LORD
ACCT FARFLE TO BE PURGED? YES
:
```

The first command informs the system volume set of the purge; the second informs the mountable volume set.

Related Information

Commands	PURGEGROUP, PURGEUSER
Manuals	None

PURGEDIR

Purges (unlinks) one or more directories.

Syntax

```
PURGEDIR[ dir=]   dir_name
 [ {   ;TREE   ;NOTREE   ;USENAME  } ]
[ {   ;CONFIRM   ;NOCONFIRM   ;CONFIRMALL  } ]
[ {   ;NOSHOW   ;SHOW  } ] [ {   ;SHOWERROR   ;NOSHOWERROR  } ]
```

Parameters

dir_name The name of the directory that is being purged (required). The *dir_name* is assumed to be an MPE name unless you begin it with a dot (.) or a slash (/) to indicate an HFS directory.

If *dir_name* is an HFS directory that ends in a slash and you don't include the NOTREE option, PURGEDIR deletes all objects at all levels under and including *dir_name*.

The use of wildcards is permitted. The *dir_name* cannot name root (/), an MPE group, or an account.

TREE Purges all objects below and including *dir_name*. The *dir_name* may or may not end in a slash (/), with no error or warning reported. Since the MPE naming convention does not support a trailing slash (/), the TREE option is the only way to delete a non-empty, MPE-named directory with a single command.

NOTREE Purges *dir_name* only if it is empty. If *dir_name* is an HFS name and ends in a slash (/), a warning tells you that NOTREE overrides the trailing slash (/).

USENAME Indicates that *dir_name* alone controls whether or not all levels of directories and files are deleted. (This is the default.) If *dir_name* is an HFS name and ends in a slash (/), then it, and all objects under it are deleted. If *dir_name* does not end in a slash (/), then only *dir_name* is purged, assuming it is empty. USENAME only applies to HFS-named directories, and is ignored for MPE-named directories.

CONFIRM Requires the user to confirm the purge of the directory. A different prompt is seen depending on whether *dir_name* is to be purged with the TREE option or with the trailing slash feature. CONFIRM is the default for sessions. CONFIRM is ignored for jobs.

NOCONFIRM Purges *dir_name* (and all objects under it for TREE purges) without user confirmation. NOCONFIRM is the default for jobs.

CONFIRMALL Requires the user to confirm each directory before the purge is executed. A proper response is one of the following:

- "Y" or "YES" to purge the directory

- "N", "NO", or **Return** to retain the directory
- "Q", "QUIT", or **Break** to stop the PURGE command

The CONFIRMALL option is ignored in jobs and when you are purging a single directory.

SHOW Displays to $STDLIST each file or directory under *dir_name* that was purged. Directory names are always displayed in an HFS syntax, even if the name was specified as an MPE name.

NOSHOW Suppresses the display of each file and directory purged. NOSHOW is the default.

SHOWERROR Displays on $STDLIST each lower-level error that prevents an object below *dir_name* from being deleted'. The object (file or directory) name is shown, followed by the error message. By default, lower-level errors are not displayed. SHOWERRORS is a synonym for SHOWERROR.

NOSHOWERROR Suppresses the display of low-level errors. NOSHOWERROR is the default. `NOSHOWERRORS" is a synonym for NOSHOWERROR.

Use

You can issue the PURGEDIR command from a job, a session, a program, or in BREAK. Pressing **Break** terminates execution of this command. You must have TD access to each component in the *dir_name* pathname, and DD permission to the parent directory of *dir_name*. (Refer to the ALTSEC command for more information on directory access.) If wildcards are specified with *dir_name*, then RD access is required to the parent directory of each wildcard component. If the purge is multilevel, then TD, RD and DD accesses are necessary to each directory below *dir_name*.

Operation

The PURGEDIR command purges the directory *dir_name*. The *dir_name* cannot name an MPE account, an MPE group, a file, or root (/). Dot (.) and dot-dot (..) can be specified but cannot be purged.

By default PURGEDIR deletes an MPE-named directory. This means that *dir_name* must follow all MPE naming rules, unless it is prefixed with a dot (.) or a slash (/). Since the MPE name syntax defines three levels, fully (or partially) qualified MPE-named directories can only be created under MPE groups. Unqualified MPE-named directories are created relative to the CWD. Directories do not support lockwords, file equations, or system defined file names (for example, $NEWPASS). If *dir_name* begins with a dot (.) or a slash (/), then HFS naming rules are enforced.

The directory referenced by *dir_name* must be empty (except for dot (.) and dot-dot (..)) in order to be purged, unless a TREE purge is requested. A TREE purge may be requested as follows:

1. Specify the TREE option. (The *dir_name* parameter does not control a multilevel purge in this case). This is the only choice available if *dir_name* is an MPE name.

2. If *dir_name* is an HFS name, ends in a slash (/), and the ;NOTREE option is *not* requested, then a TREE purge occurs.

The *dir_name* parameter cannot reference root (/) because purging root is undesirable, and most likely is not what is intended.

A file or directory is not deleted if it is being accessed (opened); however, all non-accessed objects under *dir_name* are still purged. A final "IN USE" error indicates that *dir_name* was not deleted because one or more children objects could not be removed.

If CONFIRM is specified while your session is interactive, and it is legal for you to purge *dir_name*, then you are prompted to confirm the purge of *dir_name*. If a NOTREE purge is requested, the following prompt is displayed:

```
        DIRECTORY dir_name TO BE PURGED? (YES/NO)_
```

Valid responses are YES, Y, NO, and N (case insensitive). If a TREE purge is requested, the prompt is:

```
        PURGE ALL FILES BELOW AND INCLUDING dir_name? (ALL/NO)_
```

Valid responses are ALL, NO, and N (case insensitive).

NOTE If *dir_name* is long, the prompt may wrap around. If *dir_name* is an MPE name, it is fully qualified in the prompt message. If the YES option is selected, then the purge is automatically confirmed without a prompt.

The SHOW option displays the name of each purged file and directory on $STDLIST. For example:

```
  PURGEDIR ./mydir ;TREE ;SHOW
   ./mydir/abc
   ./mydir/dir1/dir2/file1
   ./mydir/dir1/dir2/file2
   ./mydir/dir1/dir2
   ./mydir/dir1/f1
   ./mydir/dir1/f2
   ./mydir/dir1
   ./mydir/file1
   ./mydir
```

The SHOWERRORS option displays any error that prevents an object from being deleted on $STDLIST after the object name is displayed. Object names are only displayed if an error occurs.

Examples

The following examples purge dir1, which is empty.

```
             PURGEDIR /MYACCT/MYGRP/dir1
             PURGEDIR /MYACCT/MYGRP/dir1;NOTREE

     PURGEDIR /MYACCT/MYGRP/dir1/;NOTREE
  NOTREE option overrides directory name ending in a "/". (CIWARN 9041)
```

The following examples purge dir1 and all objects below dir1.

```
  PURGEDIR /MYACCT/MYGRP/dir1/
  PURGEDIR /MYACCT/MYGRP/dir1 ;TREE
```

The next example shows the command to purge MYDIR.

```
PURGEDIR mydir
```

The next example shows the command to purge MYDIR and all objects below.

```
PURGEDIR mydir;TREE
```

The next example illustrates the SHOW and TREE options.

```
PURGEDIR dir;SHOW;TREE
 ./DIR/A
 ./DIR/B
 ./DIR/dir1/A
 ./DIR/dir1/B
 ./DIR/dir1
 ./DIR/C
 ./DIR

PURGEDIR /dir1/dir2;SHOW;TREE
 /dir1/dir2/file1
 /dir1/dir2/file2
 /dir1/dir2

PURGEDIR ./foo/;show
 ./foo/dir1_below_foo/f1
 ./foo/dir1_below_foo/f2
 ./foo/dir1_below_foo
 ./foo
```

The following command purges all empty directories under the CWD with TMP in their name.

```
PURGEDIR @tmp@
```

The following command purges all directories under the CWD with names beginning with TMP, and all objects below these directories.

```
PURGEDIR tmp@;TREE
```

The following command purges all directories under the CWD with names ending with tmp, and all objects below these directories.

```
PURGEDIR ./@tmp/
```

The following command purges all empty directories rooted to /a/b.

```
PURGEDIR /a/b/@
```

The following command purges all directories rooted to CWD/a@/b@ and all objects below these directories.

```
PURGEDIR ./a@/b@/
```

You can use the PURGEDIR command to delete a directory and the files or directories it contains using wildcards. For example, to delete all directories rooted to MYACCT/MYGRP enter:

```
:purgedir /MYACCT/MYGRP/@
```

To delete all empty directories under the CWD (Current Working Directory) with TMP in their name:

```
:purgedir @TMP@
```

To delete all directories under the CWD with names beginning with TMP all objects below these directories:

```
:purgedir TMP@; TREE
```

To delete all directories under the CWD with names ending with TMP all objects below these directories:

```
:purgedir ./@TMP/
```

When wildcards are specified with *dir_name*, then RD access is required to the parent directory of each wildcard component. If the purge is multilevel, then TD, RD, and DD accesses are necessary to each directory below *dir_name*.

Related Information

Commands	CHDIR, LISTFILE, NEWDIR, PURGE, PURGEACCT, PURGEGROUP, LISTDIR (UDC), FINDDIR (UDC)
Manuals	None

PURGEGROUP

Removes a group (and all files belonging to it) from the system or from the specified volume set directory.

Syntax

PURGEGROUP ***groupname***[.*acctname*] [;ONVS=*volumesetname*]

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.

acctname Specifies the account in which the group is found. System manager (SM) capability is required to use this parameter.

volumesetname Specifies a particular volume set from which the group is to be purged. The volume set must be one already defined and recognized by the system.

Volume set names consist simply of from 1 to 32 characters, beginning with an alphabet character. The remaining characters may be alphabetic, numeric, the underscore, and periods.

If you specify a *volumesetname*, you must specify the full name of the volume set.

If *volumesetname* is specified, the volume set must be mounted or the PURGEGROUP command fails. When the *volumesetname* parameter is specified, the group is removed from the volume set directory, and not the system directory.

Operation Notes

Account managers use the PURGEGROUP command to delete a group from their account. When the command is executed during a session, MPE/iX 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 mountable, non-system volume, the command succeeds only if the group's home volume set is mounted.

Entries are purged by volume set definitions first, files second, and finally the group. If no files in the group are in use, and the group itself is not in use, the PURGEGROUP command removes the entire group. Otherwise, only inactive files are removed. To completely purge the group in this case, reenter the PURGEGROUP command when neither the group nor its files are in use.

If you specify volume-related commands or parameters for a volume set that is not currently mounted, or for an account that does not exist, MPE/iX returns an error message.

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 are purged, which seriously impairs the proper functioning of the entire system.

Use

This command may be issued from a session, a job, a program, or in BREAK. Pressing **Break** has no effect on this command.

Account manager (AM) or system manager (SM) capability is required to execute this command. Account manager (AM) capability, however, may lack the appropriate privilege to purge all files and directories below an MPE group. If you lack sufficient access to purge all directories and files, an error occurs and the MPE group is not purged.

Examples

To purge a group named GROUP1, enter:

```
PURGEGROUP GROUP1
GROUP GROUP1 TO BE PURGED? YES
:
```

To purge the group LEELA in the volume set MY_VOL, you need to issue two commands:

```
PURGEGROUP LEELA
GROUP LEELA TO BE PURGED? YES
:
PURGEGROUP LEELA;ONVS=MY_VOL
GROUP LEELA TO BE PURGED? YES
:
```

The first command informs the system volume set of the purge; the second informs the mountable volume set.

Related Information

Commands	ALTGROUP, LISTGROUP, PURGEACCT, PURGEUSER, PURGEDIR
Manuals	*Performing System Management Tasks*

PURGEJOBQ

Removes a job queue

Syntax

`PURGEJOBQ` *qname*

Parameters

qname is the name of the queue to be deleted

Operation Notes

The **PURGEJOBQ** command deletes a job queue. The queue will be deleted only if it is empty, that is, if no jobs are waiting or executing in the queue. The default system job queue can not be purged. The user must have SM or OP capability to execute the command.

This command is available in a session, job. or in BREAK. Pressing **Break** aborts the execution of this command. This command is not allowed in SYSSTART.

Example

```
:PURGEJOBQ myjobq
```

Related Information

Commands NEWJOBQ, LISTJOBQ, SHOWJOB

PURGELINK

Removes a link. (Native Mode)

Syntax

PURGELINK[LINK=] *linkname*

Parameters

linkname The name of a symbolic link file. All rules regarding file name specification apply to this parameter.

This is a required parameter. You may not use wildcards in *linkname* or specify a file equation in place of *linkname*.

Operation Notes

A symbolic link is a special file that can point to a file, group, account, or directory. Links are established through the NEWLINK command, and they are removed through the PURGELINK command.

The PURGELINK command may be issued from a session, job, program, or in BREAK. PURGELINK requires Traverse Directory (TD) and Delete Directory entry (DD) permissions.

Example

For the following examples assume that a user is currently logged on as USER1 in the group SAFE.COMPANY.

```
To remove the link /COMPANY/SAFE/PAYROLL, enter the following command:

:PUREGLINK PAYROLL

To remove the link /dira/scripts, enter the following:

:PURGELINK /dira/scripts
```

Related Information

Commands	NEWLINK, PURGE, PURGEDIR, LISTFILE
Manuals	None

PURGEUSER

Removes a user from an account.

Syntax

PURGEUSER ***user***[*.acctname*]

Parameters

user Name of the user to be deleted.

acctname Specifies the name of the account in which the user is found. Default is the logon account of the account manager.

Operation Notes

Account managers use the PURGEUSER command to delete a user from an 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 on to the system fails, and an explanatory message is displayed:

```
IN USE: CAN'T BE PURGED.
```

The user can only be purged if the user is not logged on when the PURGEUSER command is issued. An attempt to purge MANAGER.SYS always fails, since this user can never be purged if the user is logged onto the system.

If files created by a purged user remain after the user is purged from the system, the system manager can remove them with the PURGEACCT command, or the account manager can eliminate them by executing PURGEGROUP.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

To execute this command, the account must be the same as the logon account of the command issuer unless that user has system manager (SM) capability.

Example

To remove a user named USER1, enter:

```
PURGEUSER USER1
USER1 TO BE PURGED? YES
```

Related Information

Commands	`PURGEACCT`, `PURGEGROUP`, `NEWUSER`, `ALTUSER`
Manuals	*Performing System Management Tasks*

RECALL/=RECALL

Displays all pending console REPLY messages.

Syntax

RECALL=RECALL

Parameters

None.

Operation Notes

A user, the system operator, a job or a program issues the RECALL command to determine if any pending resource requests are currently awaiting a response. Pending resource requests are responded to by using the REPLY command.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console.

Any user may execute the RECALL command. However, the **CTRL A** =RECALL command may only be executed at the physical console, and cannot be executed from a job or a program.

Examples

To display all pending system console messages, which require a response, enter:

```
 RECALL
THE FOLLOWING REPLIES ARE PENDING:
10:05/#J19/15/LDEV # FOR "L00576" ON TAPE1600 (NUM)?
```

If any replies are pending, the request(s) are displayed on the console as shown above. If no replies are pending, the following message appears on the console:

```
 RECALL
NO REPLIES PENDING (CIWARN 3020)
```

Use the =RECALL command if the RECALL command is ineffective, or when a job or subsystem is being executed from the console.

```
 CTRL A
=RECALL
NO REQUESTS PENDING (SYS 15)
```

Related Information

Commands	REPLY
Manuals	*STORE and TurboSTORE/iX Manual*

Performing System Operation Tasks

REDO

Allows the user to edit and reexecute any command still retained in the command line history stack. (Native Mode)

Syntax

REDO[[CMD=] *cmdid*] [[;EDIT=] *editstring*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

Parameters

cmdid Specifies the command to execute. The command may be specified by its relative or absolute order in the command line history stack, or by name (as a string). The default is -1, the most recent command.

The following Table 6-2 illustrates the result of using various forms of the *cmdid* parameter.

Table 6-2 Re-execute Directives for the REDO Command

cmdid	Executes
(omitted)	Previous command (same as `REDO -1`).
-*n*	The *n*th command before the most recent one. The *n* represents a number in the command line stack relative to the most recent command, which is -1.
m	Command number *m* in the command line stack. The number *m* is absolute (not relative).
string	The most recent command beginning with *string*.

MPE/iX detects an error if you specify a *cmdid* that cannot be found in the history stack.

editstring A string specifying the first (of one or more) edit(s) to be performed on *cmdid* before it is displayed on the standard listing device (`$STDLIST`).

When the (edited) command line is displayed, you may edit the line interactively. `REDO` displays the command line and accepts further edits repeatedly, until you signal completion by entering a **Return** only. At this point, the CI executes the edited version of the command.

If you omit *editstring*, then you are given the opportunity to edit the command line interactively, after which the command is reexecuted.

If you specify *editstring*, it must appear, character for character, and space for space, exactly as it would if you were using the `REDO` command in interactive mode.

The edit string must be surrounded by quotation marks (" ") if it contains any scanner/parser delimiters such as: , ; " ' [ ] or = or spaces.

Operation Notes

REDO executes the command specified as *cmdid*. The user may specify an optional *editstring* that edits the command before it is reexecuted. This command is a companion to the MPE/iX DO command. Unlike the DO command, the REDO command does permit interactive editing.

If *editstring* is specified, the edit is performed on *cmdid* before the command is presented for interactive editing. If *editstring* is omitted, then editing is interactive.

In either case, the (edited) line is echoed to $STDLIST before it is reexecuted. At this point, you may edit the line interactively. The interactive (editing) mode, remains available to you until you press only **Return**.

Both *cmdid* and *editstring* must be surrounded by either single or double quotation marks if they contain any delimiters such as , ; " " [,], =, or a space.

The editing directives used in *editstring* are defined in Table 6-3.

.

Table 6-3 Editing Directives for the REDO Command

Directive	Effect
i	INSERT. If text follows the i, the text following i is inserted in the current line at the position after the i.
r	REPLACE. If text follows the r, the text following r replaces the same number of characters in the current line, beginning at the position of r.
d	DELETE. Deletes a character from the current line for each specified in the edit line. Note that "d d" does not specify a range but simply deletes one character from the position above each d. Multiple d's may be followed by an insert or replace operation.
dw	DELETE WORD. Deletes a word starting at the letter d. A word is defined as all characters except a space, comma, or semicolon. If you place the d directly beneath a word delimiter, then the word and the delimiter characters are deleted. If no word exists on the command line, no delete occurs. You may follow this directive with other edits.
d*delim*	DELETE TO DELIMITER. Deletes all characters starting at the position of the d and ending at, but not including, the specified delimiter. If *delim* is not found, no delete occurs. You may follow this directive with other edits.
d>	DELETE TO EOL. Deletes to the end of the current line from the position specified by d>. It may be followed by an INSERT or REPLACE operation.
^	UPSHIFT. Upshifts the character positioned at the ^. You may specify multiple ^ characters to upshift a series of characters. Or, you may type multiple ^ characters, followed by spaces, then followed by more ^'s to upshift some characters while skipping others. You may follow this directive with other edits.

Directive	Effect
^w	UPSHIFT WORD. Upshifts the word starting at the position specified by ^. A word is defined as all characters except a space, comma, or semicolon. If you place the ^ directly beneath a word delimiter, the delimiter is skipped and only the word is upshifted. If no word exists on the command line, no upshift occurs. You may follow this directive with other edits.
^*delim*	UPSHIFT TO DELIMITER. Upshifts all characters starting at the position specified by the ^ and ending at, but not including, the specified delimiter. If *delim* is not found, no upshift occurs. You may follow this directive with other edits.
^>	UPSHIFT TO EOL. Upshifts all characters starting from the position specified by the ^ to the end of the current line. You may follow this directive with other edits.
v	DOWNSHIFT. Downshifts the character positioned at the v. You may specify multiple v's to downshift a series of characters. Or, you may type multiple v's, followed by spaces, then followed by more v's to downshift some characters while skipping others. You may follow this directive with other edits.
vw	DOWNSHIFT WORD. Downshifts the word starting at the position specified by v. A word is defined as all characters except a space, comma, or semicolon. If you place the v directly beneath a word delimiter, the delimiter is skipped and only the word is downshifted. If no word exists on the command line, no downshift occurs. You may follow this directive with other edits.
v*delim*	DOWNSHIFT TO DELIMITER. Downshifts all characters starting at the position of the v and ending at, but not including, the specified delimiter. If *delim* is not found, no downshift occurs. You may follow this directive with other edits.
v>	DOWNSHIFT TO EOL. Downshifts all characters starting from the position specified by the v to the end of the current line. You may follow this directive with other edits.
>*text*	APPEND. The > followed by text appends the text to the end of the current line. If > is positioned beyond the end of the current line, then a replacement is performed instead.
>d	DELETE FROM EOL. Deletes from the end of the current line, right-to-left. Multiple d's may be specified after >, as well as INSERT and REPLACE strings.
>dw	DELETE WORD FROM EOL. Deletes the last word in the command line. To find the last word, trailing word delimiters are skipped. If no word exists in the command line, then none is deleted. If you follow >dw with additional editing directives, each edit is performed recursively. That is, the first edit is performed (updating the current EOL), then the next edit is performed (again updating the current EOL), and so on.
>d*delim*	DELETE TO DELIMITER FROM EOL. Starting at the end of the current line, deletes all characters right-to-left up to, but not including, *delim*. If the delimiter is not found, no delete occurs. If you follow this directive with additional editing directives, each edit is performed recursively. That is, the first edit is performed (updating the current EOL), then the next edit is performed (again updating the current EOL), and so on.

Directive	Effect
`>^`	`UPSHIFT FROM EOL`. Upshifts the character at the current EOL. You may specify multiple `^`'s to upshift a series of characters (read right-to-left) from the EOL. Also, you may follow this directive with other edits.
`>^w`	`UPSHIFT WORD FROM EOL`. Upshifts the last word in the command line. You may follow this directive with other edits.
`>^`*delim*	`UPSHIFT TO DELIMITER FROM EOL`. Starting at the end of the current line, upshifts all characters right-to-left up to, but not including, *delim*. If the delimiter is not found, no upshift occurs. You may follow this directive with other edits.
`>v`	`DOWNSHIFT FROM EOL`. Downshifts the character at the current EOL. You may specify multiple `v`'s to downshift a series of characters (read right-to-left) from the EOL, and you may follow this directive with other edits.
`>vw`	`DOWNSHIFT WORD FROM EOL`. Downshifts the last word in the command line. You may follow this directive with other edits.
`>v`*delim*	`DOWNSHIFT TO DELIMITER FROM EOL`. Starting at the end of the current line, downshifts all characters right-to-left up to, but not including, *delim*. If the delimiter is not found, no downshift occurs. You may follow this directive with other edits.
`>r`*text*	`REPLACE`. Replaces characters at the *end* of the command line. The replacement is done so that the last (rightmost) character of the replacement string is at the end of the line.
`c`	`CHANGE`. Changes all occurrences of one string to another in the current line when the search string and replace string are properly delimited. A proper delimiter is a nonalphabetic character (such as ', ", `/` or `,`). The substitution is specified as: `c`*<delim> search-string<delim>* [*replace-string* [<*delim*>]]. Omitting the *replace-string* causes occurrences of *search-string* to be deleted, with no substitution.
`u`	`UNDO`. A single `u` in column one cancels the most recent edit of the current line. Using the `UNDO` command twice in a row cancels all edits for the current line and reestablishes the original, unedited line. If `u` is placed anywhere other than column one of the current line, then a simple replacement is performed. `UNDO` makes sense only if you have a line on which you have performed some editing that can be "undone."
other	Simple replacement. Any other character (not `i`, `r`, `d`, `d>`, `>`, `>d`, `c`, or `u`) causes that character to be replaced in the current line at the position indicated by the character. In fact, simple replacement also occurs for the editing characters `i`, `r`, `c`, or `>` if they are not followed by text; or if `>` appears at or beyond the current end of line.

Editing Samples

The Table 6-4 shows examples of using the `REDO` command.

Table 6-4 REDO Editing Samples

Edit	Action
u	First occurrence undoes the previous edits. The u must be in column one.
u	Second occurrence undoes all edits on the current line. The u must be in column one.
rxyz	Replaces the current text with xyz starting at the position of r.
xyz	Replaces the current text with xyz starting at the position of x.
ixyz	Inserts xyz into the current line, starting at the position immediately before the i.
ddd	Deletes three characters, one above each d.
d xyz	Deletes a single character above the d, skips one space, then replaces the current text with xyz starting at the position of x.
ddixyz	Deletes two characters, then inserts xyz in the current line in the position before the i.
d d	Deletes one character above the first d, skips two spaces, and deletes a second character above the second d. It does not delete a range of characters.
d d>xyz	Deletes a single character above the first d, skips two spaces, and deletes to the end of the line beginning at the second d, and then appends xyz to the end of line.
>xyz	Appends xyz to the end of the current line.
>ddxyz	Deletes the last two characters from the end of the current line and then appends xyz to the end of the line.
>rxyz	Replaces the last three characters in the current line with xyz.
>ixyz	Appends xyz to the end of the line. In this case, the i command is superfluous, because > accomplishes the same result. Using >xyz would be sufficient.
c/ab/def	Changes all occurrences of ab to def, starting at c.
c"ab"	Deletes all occurrences of "ab" starting at c.
cxyz	Replaces the current text with cxyz, starting at c. Because delimiters have been specified (as they were in the previous two examples), this is a simple replacement.
^wix	Upshifts the word above the ^ and inserts an "x" at the end of the word it just upshifted.
v/abc	Starting at the position of v, downshifts all characters up to, but not including, the "/", then replaces the "/" and the next two characters with "abc".
>dw^.dw	Deletes the last word in the current line, recalculates the EOL, then upshifts all characters up to, but not including, the dot (.), then deletes the word to the left of the characters that were upshifted.

Use

This command is available in a session or in BREAK. It is not available in a job or from a program. Pressing **Break** aborts the execution of this command.

Examples

The following are examples of editing options for the `REDO` command:

`REDO PAS`	Edits the most recent command beginning with the string `PAS`.
`REDO 10`	Edits command number 10 (absolute) on the command history stack.
`REDO -2`	Edits the second-to-last command on the stack (one command before the most recent).

Related Information

Commands	`DO`, `LISTREDO`
Manuals	*Using the HP 3000 Series 900: Advanced Skills*

REFUSE

Disables jobs/sessions and/or data on a designated device.

Syntax

REFUSE[JOBS,] [DATA,] *ldev*

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.

Operation Notes

The REFUSE command prevents a device from automatically recognizing and accepting one or more of the three commands (JOB, HELLO, and DATA) users execute to introduce jobs or sessions. The JOBS parameter in the REFUSE command refers to both jobs and sessions. If neither the JOBS nor DATA parameter is supplied, both JOB (or HELLO) and DATA commands are refused. To undo the effect of the REFUSE command, use ACCEPT.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be executed only from the console unless distributed to users with the ALLOW or ASSOCIATE command.

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
```

Related Information

Commands	ACCEPT
Manuals	*Performing System Operation Tasks* (32650-90137)

RELEASE

Removes security provisions from a file. Security does not resume for a released file until you enter the SECURE command for the file.

Syntax

RELEASE *filereference*

Parameters

filereference Specifies the actual file designator of the file whose file access matrix access control you want to disable. The *filereference* can be either in MPE or HFS syntax.

MPE Syntax

If the *filereference* does not begin with a dot or a slash, it is parsed according to the MPE syntax and has the form:

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

If the file has a lockword, you must specify it; otherwise, the system prompts you for it. If you do not specify *groupname.acctname*, the system assumes the logon group and account.

HFS Syntax

If the *filename* begins with a dot (.) or a slash (/), it is parsed according to HFS syntax.

Operation Notes

- **Usage**

  You can use this command only for permanent disk files you have created. Under default system security provisions, the file must be in your logon account and must belong to your logon or home group.

- **Checking the file status**

  You can enter the LISTFILE command to determine if a file is currently released or secured. Refer to the LISTFILE command for more information.

- **Access control definition**

  An access control definition (ACD) overrides file access controls whether or not you have released or secured the file.

  For more information about ACDs, refer to the ALTSEC command in this manual.

- **Unaffected access controls**

  This command does not affect the following access controls:

Privileged files	You cannot release privileged files.
Lockwords	You cannot override lockwords.
ACDs	This command does not affect the security on files with access control definitions. However, if you remove the ACD, the file is released. Refer to the ALTSEC command in this book for more information about ACDs.

Use

You can enter this command from a session, a job, a program, or in BREAK. Pressing **Break** does not affect this command.

Example

- To release all security provisions for a file named FILE1 in your logon group and account, enter:

  ```
  :RELEASE FILE1
  ```

  If the system fails to locate the file, the following error message appears:

  ```
  UNABLE TO ACCESS FILE1.GROUP1.ACCT1.  (CIERR 356)
  ```

Related Information

Commands	ALTSEC, LISTF, LISTFILE, SECURE, ALTLOG, CHANGELOG, GETLOG, LISTLOG, LOG, OPENLOG, RESUMELOG, SHOWLOGSTATUS, SWITCHLOG
Manuals	None

RELLOG

Removes a user logging identifier from the system.

Syntax

RELLOG ***logid***

Parameters

logid	The logging identifier to be removed from the system.

Operation Notes

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. System supervisor (OP) or user logging (LG) capability is required to use this command.

After RELLOG is issued, programs containing the removed logging identifier are not allowed to access the logging system.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

To remove the logging identifier DATALOG from the system, enter:

```
RELLOG DATALOG
```

Related Information

Commands	GETLOG
Manuals	*User Logging Programmer's Guide* (32650-60012)

RENAME

Changes the file name, lockword, and/or group name of a disk file.

Syntax

RENAME ***oldfilereference,newfilereference***[;TEMP]

Parameters

oldfilereference Current name of file, written in the format:

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

To use HFS syntax, preceed the file name with a dot (.), or a slash (/).

newfilereference New name of file, in the same format as *oldfilereference*. If you omit *acctname* and/or *groupname*, the logon account and/or group are assumed.

To use HFS syntax, preceed the file name with a dot (.), or a slash (/).

TEMP Indicates that the old file was, and the new file will be, temporary files. If you do not specify TEMP, RENAME assumes that the files are permanent.

Operation Notes

The RENAME command changes the system file identification for a permanent or temporary disk file. You can use it to change the name of a file, to change the lockword of an MPE file, or to move any file to a different location.

MPE Files To rename an MPE file, you must be the file's creator and have exclusive access to the file. If you specify groupname or acctname, you must have save access to the group or account. Users with System Manager (SM) capability can rename any file to any location on the system.

You can use RENAME to move native mode MPE files to HFS directories. You cannot move compatability mode MPE files to HFS directories. For example, you can use RENAME with KSAM/XL files, but you cannot use it to rename MPE V/E KSAM files.

To successfuly rename a file across group or account boundaries, you must move it within a single volume set and that volume set must be physically mounted.

When you use RENAME to move a file that does not have an ACD to a directory or to another account, an ACD is automatically created for the file to ensure that it is protected by the appropriate file access matrix of its new location.

HFS Files To rename a file in an HFS directory, you must have delete directory entry access (DD) to the old directory and create directory entry access (CD) to the new directory.

Files in HFS directories can be renamed to files in the MPE account group structure, and they can be renamed to files in other HFS directories.

You cannot rename a directory. If either *oldfilereference* or *newfilereference* is actually a directory, you will get an error.

Spool Files If you have access to spoolfiles, you can rename them. In this case, the name of the file changes, but the contents and links to the spooler remain the same.

Use

This command may be issued from a session, a job, a program, or in BREAK. Pressing **Break** has no effect on this command.

Examples

Since temporary files exist only for the duration of your current job or session, their fully qualified file names correspond to your logon group and account. The following example shows the command entry to change the name of a temporary file from OLDFILE to NEWFILE, and reassign it to the group NEWG.

```
RENAME OLDFILE,NEWFILE.NEWG,TEMP
```

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 have SAVE access to GROUP2 and both groups must be in the system domain or reside on the same volume set. For example, to move the file MYFILE from GROUP1 to GROUP2, enter:

```
RENAME MYFILE.GROUP1,MYFILE.GROUP2
```

The following command renames the file dir2/doc/print.txt in the current working directory (CWD) to MYFILE in the group and account MYGROUP.MYACCT.

```
RENAME ./dir2/doc/print.txt, MYFILE.MYGROUP.MYACCT
```

The following command renames the file FILE1 in the PUB group to new_txt in the HFS directory dir1 under the root directory.

```
RENAME FILE1.PUB, /dir1/new_txt
```

The following command renames the KSAM XL file KSFILE in the PUB group to ksfile in the HFS directory dir1 under the root directory.

```
RENAME KSFILE.PUB, /dir1/ksfile
```

Related Information

Commands	BUILD, COPY, PURGE, PRINT
Manuals	None

REPLY/=REPLY

Replies to pending resource requests at the console.

Syntax

REPLY ***pin,reply***
=REPLY ***pin,reply***

Parameters

pin The process identification number (PIN) 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 either YES (or Y) or NO (or N).

(MAX CHARS.=*nn*``) Reply must be a string expression consisting of *nn* characters or less.

Operation Notes

User programs that have requested the use of a device and are waiting for you to reply remain suspended indefinitely and cannot be aborted until a REPLY or a **Break**/ABORT is issued. If for any reason you cannot reply as requested (for example, if the particular device is nonexistent or a special form is unavailable), then use REPLY/=REPLY with 0 if type NUM is requested, or with N if type Y/N is requested. This returns an error code to the program and the REPLY/=REPLY is aborted.

The reply usually takes the form (NUM) or (Y/N), since (MAX CHARS.=*nn*) is used only for labeled tapes and the PRINTOPREPLY intrinsic.

If your reply is not of the type specified, an error message is displayed.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the logical console, unless distributed to specific users with the ALLOW or ASSOCIATE command.

The **Ctrl A** =REPLY command can be used only from the physical console. It cannot be executed from a job or a program.

Examples

Use the REPLY command to respond to a message from the MPE/iX system, as follows:

```
10:05/#J19/15/LDEV# FOR "NAS" OF TAPE1600 (NUM)?
 REPLY 15,7
```

or

```
 CTRL A
=REPLY 15,7
```

Use the REPLY command to respond to a FORMS message from the MPE/iX system, as follows:

```
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 causes the printing to be deferred to a much lower priority. After the forms have been aligned, use the ALTSPOOLFILE command 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 causes the spoolfile to go to the printer in its assigned sequence.

When the next spoolfile becomes ACTIVE, you are 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
```

Related Information

Commands	RECALL
Manuals	*Performing System Operation Tasks*
	System Startup, Configuration, and Shutdown Reference Manual

REPORT

Displays accounting information for the logon account and group. Any user may obtain `REPORT` information about the user's logon group. (Compatibility Mode)

Syntax

`REPORT`[*groupset*] [*,listfile*] [;ONVS=[*volumesetname*]]

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 such as 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 account totals. This is the default for system managers and requires SM capability.

`ONVS=` should always be used when @.@ is used as the *groupset* parameter.

group`.@` — Reports on specified group in any account. This requires SM capability.

You may use the wildcard characters, `@`, #, and `?` to specify a set of names.

`@` — Specifies zero or more alphanumeric characters. Used by itself, it specifies all possible combinations of such characters. Used with other characters, it indicates all the possible names that include the specified characters (`@ABC@` = all names that include `ABC` anywhere in the name).

# — Specifies one numeric character (`A###@` = all names that begin with `A` followed by any three digits, followed by any combination of zero to three alphanumeric characters).

`?` Specifies one alphanumeric character (`A?#` = all the three-character names that begin with `A`, followed by an alphanumeric, followed by a digit.)

The characters may 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".

`?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
```

volume-setname Instructs MPE/iX to report account information for the specified volume set. If this parameter is omitted, the default is the MPE/iX system volume set. Refer to "Operation Notes."

Operation Notes

The `REPORT` command displays the total resource usage logged against accounts and groups, and the limits on those resources. For standard users, data is displayed for their own group(s) only; account managers may specify all groups in their account; system managers 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 session. CPU-time and connect-time contain non-zero values *only* when the MPE/iX system volume set is specified (either in the `ONVS=` parameter or by default when `ONVS=` is not used). CPU-time and connect-time are displayed as zero for non-system volume sets.

If you specify the ONVS= parameter, REPORT displays file space counts for the specified volume set(s) only. If you specify a non-system volume, all other volume names are also displayed, but their file space counts are *displayed* as zero even though they may not be zero. You should **always** specify ONVS= when @.@ is the *groupset* parameter.

If data for the MPE/iX system volume set is requested (either with or without the ONVS= parameter), file space counts are displayed for *all* volume sets (both system and *non-system*). However, the *account total* display reflects only file space in the MPE/iX system volume set.

If you specify volume-related commands or parameters for a volume set that is not currently mounted, or for an account that does not exist, MPE/iX returns an error message.

MPE/iX uses a naming convention for volume sets that differs from the MPE V/E naming convention for private volumes. As a convenience to established Hewlett-Packard users, MPE/iX does, however, accept the naming convention that was used for MPE V/E private volumes. Refer to the VSRESERVE or VSRELEASE commands in this chapter.

For information on migrating files from MPE V/E private volumes to MPE XL mountable volume sets, please refer to the chapter on DIRMIG in the *Migration Process Guide* (30367-60003).

NOTE The REPORT does not produce the same output as DISCFREE because REPORT does not account for disk space taken up by objects such as directory files and label tables. To determine how much space is taken up by other objects, issue the FSCHECK TOTALEXTENTS command.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command. Account manager (AM) capability is required to issue the command for an entire account, or system manager (SM) capability to issue the command for the entire system.

Example

To obtain the display of account information for the group, SOPRM, enter:

```
REPORT SOPRM
ACCOUNT       FILESPACE-SECTORS     CPU-SECONDS    CONNECT-MINUTES
 /GROUP       COUNT    LIMIT    COUNT    LIMIT    COUNT    LIMIT
SOPRM         13599        **    30144        **    17258        **
 /GLOSSARY     1068        **      542        **      656        **
 /PUB           182        **      123        **     1155        **
 /SECT1         180        **       85        **      429        **
 /SECT10      11779        **    25271        **     9716        **
 /SECT2         390        **     4123        **     5302        **
```

Related Information

Commands	VSCLOSE, VSOPEN, VSRELEASE, VSRESERVE, VSRESERVESYS, VSTORE, VSUSER, RESETACCT, DISKUSE, DISCFREE Utility, LISTFILE
Manuals	*Volume Management Reference Manual*

RESET

Cancels file equations.

Syntax

RESET{ *formaldesignator* @ }

Parameters

formal-designator A formal file designator name in the form *file*[.*group*[.*account*]] [:*nodespec*], for which a FILE command has 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.

Operation Notes

The RESET command resets a formal file designator to its original meaning, canceling 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 Series 900 Computer System Fundamental Operating System. The NS3000/XL AdvanceNet subsystem must be purchased separately. The *nodespec* parameter is optional. If you do not have NS3000/XL AdvanceNet, omitting the *nodespec* parameter makes no difference in the performance of the RESET command, however, specifying it does produce an error message.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

To cancel the effects of a previous FILE command that specified characteristics for a file programmatically referred to as ALPHA enter:

```
RESET ALPHA
```

Related Information

Commands FILE, LISTEQ

Manuals None

RESETACCT

Resets the running counts of CPU-time or connect-time accumulated by an account and by all groups within that account to zero.

Syntax

RESETACCT[{ @ *acct* } [,{ CPU 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 are to be reset.

CPU Specifies that only the CPU usage counter is to be reset. Default is that both the CPU-time 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-time and connect-time counters are reset.

Operation Notes

This command resets the running counts of CPU-time or connect-time accumulated by an account and by all groups within that account to zero. If all parameters are omitted when you execute RESETACCT, all counters (except file space) for all groups in all accounts are reset.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. System manager (SM) capability is required to execute this command.

Example

To reset the CPU counter for all accounts in the system, enter:

```
  RESETACCT @,CPU
```

Related Information

Commands	REPORT
Manuals	None

RESETDUMP

Disarms the debug facility call that is made during abnormal process termination. (Native Mode)

Syntax

```
RESETDUMP
```

Parameters

None

Operation Notes

This command disarms the debug facility (armed by using the `SETDUMP` command) after a process abort. It affects all processes created later under the current session or job.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. Issuing this command in BREAK does not affect existing processes.

Example

To disarm the stackdump/debug facility enter:

```
RESETDUMP
```

Related Information

Commands	`DEBUG`, `SETDUMP`
Manuals	*System Debug Reference Manual*

RESTORE

Returns files that have been stored on backup media to the system.

Syntax

RESTORE[*restorefile*] [;*filesetlist*] [;*option*[;...]]

where *option* is:

[;SHOW [=*showparmlist*]] [;ONERROR= { QUIT | SKIP | FULL}]

[;{ LOCAL GROUP=*groupname* ACC[OUN]T=*accountname*}]

[;CREATE= { ACCT | GROUP | CREATOR | PATH }]

[;CREATOR[=*username*]] [;GID[=*filegroupname*]]

[;KEEP NOKEEP] [;OLDDATE NEWDATE] [;DIRECT0RY] [;LISTDIR] [;PROGRESS[=*minutes*]]

[;FCRANGE=*filecode/filecode*[,...] ;FILES=*maxfiles*]

[;DEV=*device*] [;VOL=*volumename*] [;VOLCLASS=*volumeclassname*] [;VOLSET=*volumesetname*]

[;COPYACD] [;NOACD] [;TREE] [;STOREDIR[ECTORY] =*directoryname*]

[;PARTI[IAL] DB] [;RESTORESET=(*device*[,...])]

The following parameters are available with TurboStore/iX and TurboSTORE/iX True-Online Backup products only:

[;RESTORESET=(*device*[,...])[,(*device*[,...])[,...]]]
[;MOSET=(ldev[,...])[,(ldev[,...])[,...]]]
[;NAME=*backupname*]

Parameters

restorefile The name of the device that contains the files you want restored to the system. This file must be backreferenced, using an asterisk (*). A File equation for *restorefile* should be set up before invoking RESTORE. If you want to restore files from a file called SOURCE enter this file equation before running RESTORE:

```
FILE SOURCE;DEV=TAPE
```

The *restorefile* can now reference a remote device. For example,

```
FILE REMOTE;DEV=REMSYS#TAPE
RESTORE *REMOTE;@;SHOW
```

NM RESTORE will restore all files from the specified remote device. Although the initial tape mount request will appear on the remote console, all of RESTORE's console messages will be displayed on the local console. Currently, labeled tapes and Magneto-optical devices cannot be used for remote backup.

A message is displayed on the system console requesting the operator to mount the tape identified by the *restorefile* parameter and to allocate the device.

If *restorefile* is not supplied and the RESTORESET option is not used, then RESTORE creates a default file name. The default file name is the user's logon username. No file equation is used.

Sequential and parallel devices are specified with the RESTORESET option. Similarly, magneto-optical devices are specified using the MOSET option. You should not specify *restorefile* when using RESTORESET or MOSET.

A disk file can also be specified with a file equation for *restorefile*. An example of such a file equation would be:

```
:FILE MYDISC=DISCBACK.DAILY.BACKUP;DEV=DISC
```

Note that DEV=DISC must be specified for RESTORE to recover files from disk backups. All other information in the file equation will be ignored by RESTORE.

NOTE TurboSTORE/iX 7x24 True-Online Backup must be used to create disk backups.

filesetlist Specifies the set of files to be restored. The default depends on the user's capability, as shown below:

Default	Capability
@	None
@.@	Account manager (AM)
@.@.@	and/or system supervisor (OP)

The *filesetlist* parameter has the form shown below:

filesetitem[,*filesetitem*[...]]

where *filesetitem* can be ^*indirectfile* or *fileset*.

indirectfile A file name that backreferences a disk file. The syntax is

^*indirectfile* or !*indirectfile*

This file may consist of *fileset(s)* and *option(s)*, but only options can appear after the first semicolon (:) on each line. An option specified on one line will operate on all files in the *filesetlist*.

^*indirectfile* is the preferred format. If you use !*indirectfile*, the CI will interpret this as a variable reference, so you will have to specify *indirectfile* instead.

fileset Specifies a set of files to be restored, and optionally those files to be excluded from the RESTORE operation. The *fileset* parameter has the form:

filestorestore[-*filestoexclude*[..]]

The system restores any file that matches *filestorestore* unless the file also matches *filestoexclude*, which specifies files to be excluded from the RESTORE operation. You may specify an unlimited number of *filestoexclude*.

Since "-" is a valid character for HFS syntax file names, a blank character must separate it from HFS file sets to obtain the special negative file set meaning.

filestorestore
filestoexclude Both *filestorestore* and *filestoexclude* may be entered in MPE or HFS syntax. Wildcards are permitted for both MPE and HFS syntax.

The MPE syntax is as follows:

filename[.*groupname*[.*accountname*]]

A lockword may be specified for files to be restored, in the form:

filename/lockword.group.account

The HFS syntax is as follows:

/dir_lev_1/dir_lev_2/.../dir_lev_i/.../filedesig

or

./dir_lev_i/dir_lev_j/.../dir_lev_k/.../filedesig

If the name begins with a dot (.), then it is fully qualified by replacing the dot with the current working directory (CWD).

Each of the components *dir_lev_i* and *filedesig* can have a maximum of 255 characters with the full path name being restricted to 1023 characters. Each of the components *dir_lev_i* and *filedesig* can use the following characters:

Letters a to z

Letters A to Z

Digits 0 to 9

Special characters - _ .

For HFS syntax, the lowercase letters are treated distinctly from the uppercase letters (no upshifting). Names in MPE syntax are upshifted.

Both MPE and HFS name components can use the characters @, #, and ? as wildcard characters. These wildcard characters have the following meaning:

@	specifies zero or more alphanumeric characters.
#	specifies one numeric character.
?	specifies one alphanumeric character.

These wildcard characters can be used as follows

n@	Restore all files starting with the character n.

`@n` Restore all files ending with the character `n`.

`n##...#` Restore all files starting with character `n` followed by up to seven digits (useful for storing all EDIT/3000 temporary files).

`n@x` Restore all files starting with the character `n` and ending with the character `x`.

`?n@` Restore all files whose second character is `n`.

`n?` store all two-character files starting with the character `n`.

`?n` Restore all two-character files ending with the character `n`.

Also, character sets may be specified in the following syntax:

`[ct]` specifies letter `c` or `t`.

`[c-t]` specifies any letter from range `c` to `t`.

`[e-g1]` specifies any letter range `e` to `g` or digit `1`.

Examples of using character sets are:

`[A-C]@` Restore all files that begin with the letters `A, B,` or `C`.

`myset[e-g1]` Restore all files that begin with the name `myset` and end in `e, f,` or `g`, or `1`.

`myset`
`[d-e1-6]` Restore all files that begin with the name `myset` and end in `d` or `e`, or `1, 2, 3, 4, 5,` or `6`.

You may specify up to a maximum of sixteen characters for each character set and you may not nest brackets.

A character set specifies a range for only one (1) ASCII character. The range `[a-d]@` gets all files that begin with the letter `a` through the letter `d`. The ranged `[ad-de]` may cause unpredictable results.

Since the hyphen (-) is a valid character for HFS syntax file names, it is allowed inside a character set, immediately following a left bracket ([) or preceding a right bracket (]). When specified between two characters, the hyphen implies a range of characters.

Specifying Database Files

When specifying TurboIMAGE and ALLBASE/SQL databases to be restored, only the root file or DBCon file needs to be specified. `RESTORE` will determine which other files belong to that database, and will restore all of them. If dataset file(s) are specified without specifying a root file, then a warning will be printed for each file, and they will not be restored. Individual database files can be restored without the root file by specifying the `;PARTIALDB` option on the `RESTORE` command line.

Database corruption may result if not all database files are restored from a backup. Be sure that you only want to restore certain database files before overriding the default behavior with `;PARTIALDB`.

MPE and HFS Naming Equivalences

When an MPE name component is a single @ wildcard, the @ will be "folded" to include all MPE and HFS named files at that level and below. To specifiy only MPE-named files, use `?@` instead.

MPE wildcards are not expanded in *filestoexclude*. This means that `@.@.@-@.@.@` is NOT an empty fileset. It contains all of the HFS named files on the system.

A fileset may be entered in any of the following formats and may use wildcard characters. Equivalent MPE and HFS formats are grouped together as follows.

file.group.acct/ACCT/
GROUP/FILE One particular file in one particular group in one particular account.

file.group/LOGON-
ACCT/GROUP/FILE One particular file in one particular group in the logon account.

file
./FILE One particular file in the logon group and account.

@.group.acct
/ACCT/GROUP/ All files (MPE and HFS) in one particular group in one particular account.

?@.group.acct All MPE name files in one particular group in one particular account.

@.group/LOGON-
ACCT/GROUP/ All the files (MPE and HFS) in one particular group in the logon account.

?@.group All MPE named files in one particular group in the logon account.

@.@.acct
/ACCT/ All the files (MPE and HFS) in all the groups in one particular account, plus all the files and directories under the specified account.

thisisit.@.account Any MPE file named `thisisit` in all groups in one particular account.

?@.@.acct All MPE named files in all the groups in one particular account.

@ All (MPE and HFS) files in the CWD. This is the default for everyone, regardless of permissions.

	@.@	All (MPE and HFS) files in the logon account.
	@.@.@	All the files and directories (MPE and HFS) on the system.
	?@.@.@	All MPE named files on the system.

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. For instance, the following file equation entered before the `RESTORE` command would send the listing to a line printer:

```
FILE SYSLIST; DEV=LP
```

showparmlist — Tells `RESTORE` what information to display for the files that are restored. If you specify `;SHOW` and omit *showparmlist*, then the default is `SHORT` if the recordsize of `SYSLIST` is less than 132 characters, or `LONG` if the recordsize is equal to or greater than 132 characters. The format for showparmlist is:

showparm `[,`*showparm*`[,`*showparm*`[,...]]]`

where showparm may be one of the options described below. If you do not specify `SHORT` or `LONG`, then the base information is `SHORT` if SYSLIST is less than 132 characters, or `LONG` if `SYSLIST` is 132 or more characters.

NOTE If an HFS-named file is specified in the *filesetlist*, or the expansion of a wildcard includes a HFS-named file, then a HFS-style output listing will be used. This listing shows the same information as the MPE format, but puts the name of the file at the right end of the listing, to allow for longer HFS names. If a HFS name is too long to fit in the record size of the output file, it will be wrapped onto the next line. Wrapping is signified by a "*" as the last character on the line.

showparm	`SHORT`	Overrides the `LONG` display to show file, group, and account name or the fully qualified path name, volume restrictions, file size (in sectors), file code, and media number.
	`LONG`	Overrides the `SHORT` display to show all the information that `SHORT` does plus the ending reel number, record size, blocking factor, number of extents, EOF, and file starting and ending media number. For spoolfiles, the old spoolfile name is also displayed.
	`NAMESONLY`	Displays only the filename and the starting and ending media number. You cannot use `NAMESONLY` with `SHORT` or `LONG`.
	`DATES`	Displays the creation date, the last date of access, and the last date of modification.

`SECURITY` For MPE format listing, causes `SHOW` to display the creator and the file access matrix for all the files which do not have an active ACD. For files with active ACDs only, the phrase `*ACD EXISTS*` is displayed.

For HFS format listing, the phrase `*ACD EXISTS*` or `*ACD ABSENT*` is displayed, depending on whether the file has an ACD.

`PATH` Forces all file listings to be in HFS format. Full HFS pathnames are displayed instead of MPE style names.

`OFFLINE` Sends another copy of the `SHOW` output to the formal file designator OFFLINE, which defaults to device LP.

`ONERROR` Tells `RESTORE` what to do if there is a tape read error. If you omit this parameter, then the default option is `QUIT` for labeled and unlabeled tapes. `ONERR` is a synonym for `ONERROR`.

`QUIT` Tells `RESTORE` to abort after a tape read error.

`SKIP` Tells `RESTORE` to perform a file-skip-forward past a tape error, resynchronize, and resume reading from the tape.

`FULL` Tells `RESTORE` to restore a file even if a media error occured while reading the file's data. SM or OP capability is required to specify this option. A file can be partially restored, with "holes" where missing data would be. Warnings are issued on the RESTORE listings for all files that are partially restored. In the summary of files restored at the end of the listing, there is a total count of all partially restored files.

The use of this option could lead to corrupted copies of files. You should only use it as a last resort, when there is no other way to recover file data. It should NEVER be used as the default `ONERROR` option.

`LOCAL` Specifies that files will be restored regardless of the system's directory structure. The files will be restored in the user's current working directory. The creator will be changed to the current user.

`GROUP=` *groupname* Specifies that the files being restored will be restored to an existing group identified as *groupname*. If you specify `LOCAL`, you cannot specify *groupname*.

`ACCOUNT=` *accountname* Specifies that the files being restored will be restored to an existing account identified as *accountname*. If you specify `LOCAL`, you cannot specify *accountname*.

`CREATE` Allows you to restore files whose group, account, or creator does not yet exist in the system's directory. The account and groups will be created with default capabilities.

If no suboptions are specified, then CREATE defaults to ACCOUNT,GROUP,CREATOR,PATH for SM or OP, to GROUP,CREATOR,PATH for AM, and to PATH for everything else.

If CREATE is specified, the necessary directory structures are created, provided the user has the appropriate capabilities. System Manager (SM) or System Supervisor (OP) capability is needed for account, group, and user creation. Account Manager (AM) capability is needed for group and user creation.

GROUP Instructs MPE/iX to examine the file label of the file being restored and create the group that it finds named in the file label. The user must have Account Manager (AM), System Manager (SM), or System Supervisor (OP) capability.

ACCOUNT Instructs MPE/iX to examine the file label of the file being restored and create the account that it finds named in the file label. The user must have system manager (SM) or system supervisor (OP) capability .

CREATOR Instructs MPE/iX to examine the file label of the file being restored and create the creator that it finds named in the file label. The user must have the appropriate capabilities: AM, SM, or OP if the user is in the logon account; SM or OP for users outside the logon account. If the CREATOR=*username* parameter is specified, that creator identification will be used, instead of the user in the file label.

If CREATE=CREATOR is not used, the default behavior is: If the creator of the file is not found in the system directory, the file will not be restored. You will get an error message telling you that the creator does not exist In order to restore this "orphan" file, you must use the CREATOR option or the CREATE option.

Refer to the "EXAMPLES" section for this command.

PATH Instructs RESTORE to create the hierarchical portion of the path necessary to restore the files. The user must have the appropriate access capabilities. Read and traverse access is required over the path and insert entry access is required for the node where the next entry is being created. If the path information information exists on the media then the path is created using the information. Otherwise, a default ACD and the restoring process' uid/gid are used. Note that the suboptions ACCOUNT and GROUP are required to get the accounts and groups created, respectively.

CREATOR= *username* All files will have their creator identifications changed to the specified user name. If username does not exist, then the file is not restored, unless CREATE is specified.

If CREATOR=*username* is not specified, the creator in is determined from the file label as it appears on the tape.

GID Changes the file gid to the supplied file group name. If filegroupname is omitted, then the gid present on the media is preserved. This option overrides the account and local options with respect to the gid changes.

filegroupname The file sharing group name which will be the new gid for all files being restored. If this parameter is not specified then the gid on the media is preserved.

KEEP If a file on the RESTORE media has the same name as a file already residing on the disk, KEEP instructs the system to preserve the file on the disk and to skip over the file on the RESTORE media. The file on tape is not restored and the file on the disk remains as it was.

If you do not specify KEEP, then the file on the RESTORE media replaces the identically named file on the disk. The only exception is if the file on the disk is being accessed when RESTORE attempts to replace it. In that case, RESTORE preserves the file on the disk (as if you had specified KEEP) and skips over the file on the backup.

NOKEEP Instructs the system to restore every file on the tape, even if it has the same name as a file already residing on the disk. This is the default.

NEWDATE or OLDDATE STORE and RESTORE maintain four times and dates for each file: the creation date, modification date, last access date, and the state change date. NEWDATE changes all dates and times to the date and time that RESTORE was executed. OLDDATE retains all dates and times from the date of the store procedure. The default is NEWDATE.

DIRECTORY Instructs RESTORE to restore all the volume set directories on the media. You must have system supervisor (OP) or system manager (SM) capability to use this parameter. All HFS directories on the media will also be restored.

PROGRESS Instructs RESTORE to report its progress at regular intervals by displaying the message RESTORE OPERATION IS nnn% COMPLETE. For interactive users, this message is displayed on $STDLIST. For jobs, this message is sent to the system console.

minutes A positive number specifying the number of minutes between progress messages. The maximum is 60. The default is 1 (one) minute.

LISTDIR This option may not be specified with any other option, other than DIRECTORY. It displays information from the tape directory and tape label, but does not restore any files. The type of tape created, the record size, and any files that match your filesetlist are displayed. If specified with DIRECTORY, the names of the all volume set directories and all HFS directories on the media are also displayed. The security restrictions that apply to filesetlist also apply here. The output goes to SYSLIST.

The LISTDIR option applies only to NMSTORE tapes. It cannot be used for MPEv format tapes.

FCRANGE The set of file code ranges that are to be restored.

filecode/filecode A file code range. A filecode is an integer between -32768 and 32767. FCRANGE=1000/1040 would restore only those files having file codes between 1000 and 1040. You may specify a maximum of eight file code ranges.

`FILES=` *maxfiles* If you are restoring a large number of files from an MPE V/E (transport) tape, specify a number at least as large as the number of files to be restored. The default is 4000.

This parameter is ignored when you are restoring MPE XL format store tapes. No limit is imposed.

When a `FILES=` option is put in an indirect file, it is ignored.

`DEV=` *device* Specifies the device on which the restored files are to reside. It takes one of two forms:

devclass Specifies the type of device. The file is allocated to the home volume set (within the specified device class) of the group into which it is being restored.

ldn Specifies a particular logical device number (ldn) corresponding to a particular device. The file will be allocated to that device only if one of the volumes in the home volume set (of the group into which a file is being restored) currently occupies the device.

By default, MPE/iX attempts to restore the file on a logical device compatible with the type and subtype specified in the file's file label and with the type and subtype of the mounted home volume set (of the group into which a file is being restored). If this fails, an attempt is made to restore the file on the same device class as specified in the file's file label and that of the mounted home volume set (of the group into which a file is being restored). If this fails, an attempt is made to restore the file on any member of the home volume set (of the group into which a file is being restored). If this fails, the file is not restored.

You cannot use `DEV` with the `VOLSET`, `VOLCLASS`, or `VOL` options.

`VOL` The volume on which the restored files are to reside. If there is no room on this volume, the device restrictions will default to the volume's class; if this fails, it will default to the volume's set; if both fail, the files will not be restored.

volumename A volume name. If no `VOLCLASS` or `VOLSET` options are specified, volumename must reside on the system volume set.

`VOLCLASS` The volume class on which the files are to reside. If there is no room on this volume class, the device restrictions will default to the volume class's volume set; if this fails, the files will not be restored.

volumeclassname A volume class name. If no `VOLSET` options are specified, volumeclassname must reside on the system volume set.

`VOLSET` Specifies the volume set on which the files are to reside. If the specified directories do not exist on that volume set, the file(s) will be restored to the specified group and account.

volumesetname A volume set name. If you specify the VOL or VOLCLASS options, the corresponding volume/volume class name must reside within this volume set.

Volume Set Notes

VOLSET, VOLCLASS and VOL may not be used with the DEV option.

You can inadvertently restore files to groups or accounts that you did not intend. This can happen if the accounting structure of the files you are restoring does not match the accounting structure of the target volume, volume class, or volume set. For instance, if you restore files to VOLSET=joes_vs (assume that joes_vs exists) but the accounting structure of those files does not exist on joes_vs, the files will be restored to the volume set where the group and account exist. This may not be where you intended them to go. The system does not prevent this, so you must use caution.

MPE/iX volume sets are not compatible with MPE V/E private volumes, and MPE XL introduces a new naming convention for volume sets. Refer to the VSRESERVE and VSRELEASE commands.

COPYACD Directs RESTORE to copy the ACD associated with the files or directories from the media. This option is on by default.

NOACD Directs RESTORE not to copy the ACD associated with the files or directories from the media. This option overrides the default COPYACD option.

TREE Forces every HFS syntax file set to be scanned recursively, irrespective of the slash specified or not at the end of the file set.

NOTREE Forces every HFS syntax file set not to be scanned recursively irrespective of the slash specified or not at the end of the file set. NOTREE yields a horizontal cut in the hierarchical directory.

STOREDIRECTORY Specifies that RESTORE should use the supplied *directoryname* when looking for the disk store directory file. This option should be specified if the disk directory file for this backup resides in a directory other than the default path of /SYS/HPSTORE/store_dirs/. If a disk directory file exists in the default directory for this backup, the STOREDIRECTORY option does not need to be specified. The user needs to have access permissions to the STOREDIRECTORY path and the STORE directory file.

directoryname The name of the disk directory file to be used by RESTORE. It can be in either MPE or HFS format. If it is not a fully qualified filename, it will be qualified by the CWD. This file should either be a disk directory file created by STORE or a symbolic link pointing to one.

PART[IAL]DB Allows RESTORE to restore individual database dataset files without specifying the database's root or DBCon file.

Database corruption may result if not all database files are restored from a backup. Be sure that you only want to restore certain database files before overriding the default behavior with ;PARTIALDB.

THE FOLLOWING OPTIONS ARE AVAILABLE ONLY IF TURBOSTORE XL OR TURBOSTORE XL II IS INSTALLED ON YOUR SYSTEM. TURBOSTORE IS NOT PART OF THE FUNDAMENTAL OPERATING SYSTEM, BUT MAY BE PURCHASED SEPARATELY.

For additional information on TURBOSTORE XL, refer to the *Store and Turbostore/iX Manual* (30319-90001).

RESTORESET

Specifies parallel and sequential backup devices. This option cannot be use if the *restorefile* parameter is specified.

Consecutive tapes are specified in the following way:

```
;RESTORESET = (*tape1,*tape2,*tape3,...)
```

This instructs MPE/iX to use only one drive at a time for the restore. When the first reel of tape is exhausted, RESTORE will shift to the next available drive, leaving the first free for rewinding and changing reels. Thus, at any given time, only one drive is restoring files and the effect is to accelerate the restore process.

In the following example, all three tapes will be used in parallel during the restore:

```
;RESTORESET=(*tape1),(*tape2),(*tape3)...
```

In the following example, sets of tapes are used sequentially for the restore. Two tapes would be restoring at any particular moment, while the other two are rewinding so that the operator may switch reels.

```
;RESTORESET=(*tape1,*tape2),(*tape3,*tape4)
```

This option cannot be used if the *restorefile* parameter is specified.

device

Specifies the device from which the file is to be restored. It must be a magnetic tape or DDS. This device should be specified in a file equation before you invoke the RESTORE command, ie:

```
FILE DEVICE;DEV=TAPE
```

This file equation can also specify a remote device or a disk file.

MOSET

Specifies parallel Magneto Optical (MO) backup devices. This option is not available if the *storefile* option is specified.

Parallel devices are specified by either of the two following commands:

```
;MOSET = (12),(13),(15)
```

```
;MOSET = (MO),(MO),(MO)
```

All MO devices are used in parallel during the restore. The preferred format is specifying just "MO", since RESTORE will use the the NAME parameter to locate the correct media.

This option is not available if the *restorefile* option is specified.

`NAME` This parameter must be specified with the `MOSET` option, and cannot be specified without it. If specifies the logical name to be used for the backup. For example:

```
RESTORE @.@.@;;MOSET=(12);NAME=DAILY.D23OCT90.BOZO
```

This name could indicate that the restore should be taken from the daily backup done on 23 Oct 1990 on the system called BOZO.

backupname A three field name of a total maximum length of 26 characters. The format is *fname.gname.aname*. The name represents the "handle" to this particular backup and can is used to retrieve files from this backup. The *fname*, *gname* and *aname* can be up to 8 alphanumeric characters. For example `DAILY.D24OCT90.SYSTEM`.

Operation

This command restores data into the system (on disk), from a file or files previously stored by the `STORE` command. A message is shown on the system console requesting the system operator to mount the device(s) identified by the restorefile parameter or the `RESTORESET` option, and to allocate the device(s).

No message is displayed if AUTOREPLY is configured through SYSGEN.

- **Command process**

  The output generated by `RESTORE` is sent to a file whose formal designator is SYSLIST. Any errors encountered during the restore will be reported to SYSLIST (and optionally OFFLINE). The `ONERR` option determines if `RESTORE` will continue after encountering an error restoring a file. Any file belonging to a group whose home volume set has not been mounted will not be restored.

  If you are restoring files that were stored on a large MPE V/E tape or disk, such as a SYSGEN tape, you must include the maxfiles parameter. Specify a number at least as large as the number of files to be restored. The default is 4000.

- **Required capabilities for restoring files**

  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, 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 (PM), system supervisor (OP), or system Manager (SM) capability. If you have standard user capability, you can restore only those files in your logon account.

  With the `;CREATE` option, you may 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. However, these structures will be created with default capabilities.

- **Lockword requirements**

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. The exception is AM. If you have AM and you are working in your own account, you do not have to supply the lockword. If RESTORE is executed as a job, however, all users lacking SM, OP, or AM capability must supply file lockwords.

- **Disk space requirements**

  RESTORE determines whether sufficient disk space remains to restore a file that already exists on the disk. If sufficient space remains, RESTORE writes a new copy of the file to the disk before purging the old copy of the file. The old copy of the file is purged only if the restore operation is successful.

- **Restoring True-Online Backups**

  When restoring backups created with TurboSTORE/iX 7x24 True-Online Backup, when the sync point occurred at the end of the backup, RESTORE must read the complete store directory information before restoring any files. If a store disk directory file exists for this backup, or one is specified with the STOREDIRECTORY option, then RESTORE can read the directory information from this file before starting to restore files. However, if a disk directory file does not exist, or is not specified, then RESTORE may prompt the user to mount the last media from the backup. RESTORE will skip to the final media directory information, and then will prompt the user to mount the first needed media for the backup. If you know that you are restoring from a sync at end True-Online backup and do not have a disk directory file, then you can speed up the restore process by mounting the last piece of media first.

  Files that have after image data from a sync-at-end True-Online backup will be inaccessible between the time that the normal file data is restored, and the after image log data is read in from the end of the backup and restored. You will not be able to read or modify these files until the after image log data has been applied.

Use

This command may be issued from a session, job, or program. If you press [Break] during a restore, the operation continues while you interact with the Command Interpreter. Both ABORT and RESUME can be used within BREAK.

The user must have System Manager (SM), System Supervisor (OP), or Privileged Mode (PM) capability to use this command for privileged files.

Examples

To restore all files belonging to your logon group from the *restorefile* 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 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*, no file equation need be used. For example:

```
:RESTORE ;ABC.PUB.SYS;SHOW

TURBO-STORE/RESTORE     VERSION      A.50.11     HP36398A
               (C) 1986 HEWLETT-PACKARD CO.
WED, NOV 23  1994  11:22 AM
WILL RESTORE           1 FILES          ;  NUMBER OF FILES ON MEDIA          1

FILENAME GROUP ACCOUNT VOLUME RESTRICTIONS     SECTORS CODE   REEL
ABC      .PUB .SYS     DISC     :C     0     1

FILES RESTORED:      1
:
```

If you restore all files without specifying a fileset, a warning will appear, alerting you that all files, based on your capabilities, will be restored.

```
:RESTORE

        TURBO-STORE/RESTORE  VERSION  A.50.03  HP36398A
(C) 1986 HEWLETT-PACKARD CO.
THU, JAN  6, 1994,  8:10 PM
WARNING: YOUR DEFAULT FILESET BECOMES '@' SINCE YOU HAVE NONE OF
   OP, AM, OR SM CAPABILITY  (S/R 1913)
```

To have the list of restored files printed on a line printer, enter:

```
:FILE T;DEV=TAPE
:RESTORE *T;@;SHOW=OFFLINE
```

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
```

This changes the creator of FILEA to USERB. USERB must exist on the system.

```
:RESTORE *TAPEFILE; FILEA.GROUPA.ACCOUNTA; CREATE=CREATOR
```

This creates USERA on the system.

```
:RESTORE *TAPEFILE; FILEA.GROUPA.ACCOUNTA; CREATE
```

Creates USERA on the system, and GROUPA and ACCOUNTA, if necessary, and if you have the require capabilities.

To restore only a subset of the fileset, enter

```
:RESTORE *T;@.@.@-@.PUB.SYS
```

This restores all files except those in PUB.SYS.

Related Information

Commands	STORE, VSTORE, REPLY, RECALL
Manuals	*STORE and TURBOSTORE/iX Manual*
	Magneto-Optical Media Management User's Guide
	Volume Management

RESUME

Resumes execution of a suspended operation. (Native Mode)

Syntax

RESUME

Parameters

None.

Operation Notes

After a program or MPE/iX 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/iX commands are aborted rather than suspended by a BREAK, and thus cannot be resumed.

If, instead of RESUME, you enter another program command (such as EDITOR, FTNXL, or RUN) or one of the nonprogram commands (HELLO or BYE), 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 (suspended) 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 the prompt, the suspended program continues at the point where it was interrupted. If you had logged on using the PARM= option of HELLO to create a process with PARM=1 (or 3), and then have the occasion to respond YES to an ABORT? message, MPE/iX aborts the command process and logs you off immediately.

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. Pressing **Break** has no effect on this command.

Example

To continue a suspended program at the point of interruption, enter:

```
  RESUME
READ PENDING
Return
```

Related Information

Commands ABORT

Manuals None

RESUMEJOB

Resumes a suspended job. (Native Mode)

Syntax

```
RESUMEJOB #Jnnn
```

Parameters

#J*nnn*	A job number.

Operation Notes

The system operator uses the RESUMEJOB command to resume processing a job suspended with the BREAKJOB command. The job continues execution from the point at which it was suspended; no message is issued.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be executed only from the console unless distributed with the ALLOW command, or if the JOBSECURITY is set LOW.

Example

To resume the processing of job 68, enter:

```
RESUMEJOB #J68
```

Related Information

Commands	BREAKJOB
Manuals	*Performing System Operation Tasks*

RESUMELOG

Resumes system logging following suspension caused by an error. (Native Mode)

Syntax

RESUMELOG

Parameters

None.

Operation Notes

When the operator resumes logging with the RESUMELOG command, 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.

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) capability.

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 RESUMELOG. A confirmation message then appears at the system console, as follows:

```
ST/10:45/LOG FILE NUMBER 104. LOGGING RESUMED.
ST/10:45/LOG FILE NUMBER 104 ON.
```

Related Information

Commands	ALTLOG, CHANGELOG, GETLOG, LISTLOG, LOG, OPENLOG, RELLOG, SHOWLOGSTATUS, SWITCHLOG,
Manuals	*System Startup, Configuration, and Shutdown Reference Manual*

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 back up *nnn* files or *nnn* pages and resume printing at that point. (Refer to "Operation Notes.")

FORWARD Instructs the spooler to step forward *nnn* files or *nnn* pages and resume printing at that point. (Refer to "Operation Notes.")

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 a RESUME. (Must be an integer between 1 and 256, inclusive.)

FILES or PAGES Informs the spooler process which unit of measure to use when printing a RESUME. For the purposes of this command, FILE is defined as the text appearing between FOPEN intrinsic statements within the spoolfile. (Refer to "Operation Notes.") Using 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.

Operation Notes

If you specify only the *ldev* parameter, the printer resumes printing at the beginning of the highest-priority spoolfile. Otherwise, the printer resumes 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 getting 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, an error message is displayed on the system console and printing resumes at the beginning of the file. Similarly, an error message is displayed if you instruct the spooler to advance FORWARD beyond the point where files exist. In this case, printing does 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.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be executed only at the console unless distributed to users with the `ALLOW` or `ASSOCIATE` command.

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
```

Related Information

Commands	`SUSPENDSPOOL`
Manuals	*Performing System Operation Tasks*

RETURN

Causes execution to return from the current user command (UDC or command file) to the calling environment. (Native Mode)

Syntax

RETURN

Parameters

None

Operation Notes

This command terminates the execution of the currently executing user command. Control resumes in the calling environment at the command line following the user command in which RETURN was embedded. Invoking RETURN at the CI colon (:) prompt has no effect.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

The following example uses the RETURN command to exit from a command file prematurely based on a parameter error condition.

```
PARM ERROR_NUM
COMMENT DISPLAY CIERR MESSAGE ASSOCIATED WITH "ERROR_NUM".
IF NOT NUMERIC (!ERROR_NUM) THEN
   ECHO EXPECTED A NUMBER.
   RETURN
ENDIF
SETVAR CIERROR ABS (!ERROR_NUM)
ECHO !HPCIERRMSG
```

The last two lines above can be combined as:

```
ECHO ![SETVAR(CIERROR,ABS(!ERROR_NUM))] ![HPCIERRMSG]
```

This line causes a slightly different output because the error number precedes the message.

Related Information

Commands	ESCAPE
Manuals	None

RPG

Compiles an RPG/V program in compatibility mode. RPG/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately.

Syntax

RPG[*textfile*] [,[*uslfile*] [,[*listfile*] [,[*masterfile*] [,[*newfile*]]]]]

Parameters

textfile — The 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 `RPGTEXT`. Default is `$STDIN`.

uslfile — The 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 using the MPE/iX `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/iX `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 — The actual file designator of the file on which the program listing is written. This can be any ASCII output file. The formal file designator is `RPGLIST`. Default is `$STDLIST`.

masterfile — The actual file designator of the master file to be merged against *textfile* to produce a composite source. This can be any ASCII input file. The 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` overwrite those in *masterfile*.

newfile — The actual file designator of the file produced by merging *textfile* and *masterfile*. This can be any ASCII output file. The 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 the "Implicit FILE Commands for Subsystems" discussion of the FILE command.

Operation Notes

This command compiles an RPG program onto a user subprogram library (USL) file on disk. If you do not specify *textfile*, MPE/iX 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.

Use

This command may be issued from a session, job, or program. It may not be issued in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles an RPG program entered from your standard input device, stores the object code in the default USL file $OLDPASS, and sends the listing to the standard list device:

```
RPG
```

The next example compiles an RPG program contained in the disk file SOURCE. The object code is stored in the USL file OBJECT, which is a permanent disk file created with the BUILD command. The program listing is sent to the disk file LISTFL:

```
BUILD OBJECT;CODE=USL

RPG SOURCE,OBJECT,LISTFL
```

To compile an RPG program and store the object code in the USL file OBJECT (created during the compilation process), enter:

```
RPG SOURCE,OBJECT,LISTFL
```

Related Information

Commands	RPGGO, RPGPREP, PREP, RUN
Manuals	*MPE Segmenter Reference Manual*
	RPG/3000 Compiler Reference Manual

RPGGO

Compiles, prepares, and executes an RPG/V program in compatibility mode. RPG/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately.

Syntax

RPGGO[*textfile*] [,[*listfile*] [,[*masterfile*] [*,newfile*]]]

Parameters

textfile — The actual file designator of the input file from which source program is read. This can be any ASCII input file. The formal file designator is `RPGTEXT`. Default is `$STDIN`.

listfile — The actual file designator of the file on which the program listing is written. This can be any ASCII output file. The formal file designator is `RPGLIST`. Default is `$STDLIST`.

masterfile — The actual file designator of a file which is merged against *textfile* to produce a composite source. This can be any ASCII input file. The 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` overwrite those in *masterfile*.

newfile — The actual file designator for the file produced by merging the *textfile* and the *masterfile*. This can be any ASCII output file. The 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 the "Implicit FILE Commands for Subsystems" discussion of the `FILE` command.

Operation Notes

This command compiles, prepares, and executes an RPG program. If you do not specify *textfile*, MPE/iX expects the source code to be entered from your standard input device.

The USL file created during compilation is a system-defined temporary file `$OLDPASS`, which is passed directly to the MPE segmenter. It cannot be accessed, since the segmenter also uses `$OLDPASS` to store the prepared program segments and overwrites the USL file of the same name.

Use

This command may be issued from a session, job, or program. It may not be issued in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

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

```
RPGGO
```

To compile, prepare, and execute an RPG program read from the disk file SOURCE and send the program listing to the disk file LISTFL, enter:

```
RPGGO SOURCE,LISTFL
```

Related Information

Commands	RPG, RPGPREP, PREP, RUN
Manuals	*MPE Segmenter Reference Manual*
	RPG/3000 Compiler Reference Manual

RPGPREP

Compiles and prepares an RPG/V program in compatibility mode. RPG/V is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately.

Syntax

RPGPREP[*textfile*] [,[*progfile*] [,[*listfile*] [,*masterfile*] [,[*newfile*]]]]

Parameters

textfile The 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 `RPGTEXT`. Default is `$STDIN`.

progfile The actual file designator of the program to which the prepared program segments are written. When you omit *progfile*, the MPE segmenter creates the program file, which resides 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/iX `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 The actual file designator of the file on which the program listing is written. This can be any ASCII output file. The formal file designator is `RPGLIST`. Default is `$STDLIST`.

masterfile The actual file designator of the master file that is merged against *textfile* to produce a composite *sourcefile*. This can be any ASCII input file. The 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` overwrite those in *masterfile*.

newfile The actual file designator of the file produced by merging the *textfile* and the *masterfile*. This can be any ASCII output file. The formal file designator is `RPGNEW`. Default is that no file is written.

NOTE The formal file designators used in the command (`RPGTEXT`, `RPGLIST`, `RPGMAST`, and `RPGNEW`) 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.

Operation Notes

This command compiles and prepares an RPG program to a program file on disk. If you do not specify *textfile*, MPE/iX 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 MPE 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.

Use

This command may be issued from a session, job, or program. It may not be issued in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To compile and prepare an RPG program entered from your standard input device, and send the listing to your standard list device, enter:

```
RPGPREP
```

The USL file created during compilation is a temporary file passed directly to the MPE segmenter. You can access it under the name $OLDPASS only if the prepared program segments are not also stored in $OLDPASS (which overwrites the USL file). Therefore, to save the compiled USL and the prepared program file, specify a nonexistent 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 is 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 file name 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
```

Related Information

Commands	`RPG`, `RPGGO`, `PREP`, `RUN`
Manuals	*MPE Segmenter Reference Manual*
	RPG/3000 Compiler Reference Manual

RPGXL

Compiles an RPG/XL program. RPG/XL is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if RPG/XL is installed on your system. (Native Mode)

Syntax

RPGXL[*textfile*] [,[*objectfile*] [,[*listfile*]]] [;INFO=*quotedstring*]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

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`.

objectfile Actual file designator of the object file to which the object code is stored. This file is stored in binary form and has a file code of `1461` or `NMOBJ`. Its formal file designator is `RPGOBJ`. If the *objectfile* parameter is omitted, the object code is saved to the temporary file `$OLDPASS`.

If you specify *objectfile*, the compiler stores the object file in a permanent file of the correct size and type, and with the name you specified. If a file of the same name already exists, the object code overwrites that file. If the compiler issues an error message telling you that a new or existing object file you are trying to compile to is too small, build the object file with a larger size and recompile to it. You may use the MPE/iX `SAVE` command to store `$OLDPASS` as a permanent file under another name.

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 `RPGLIST`. Default is `$STDLIST`.

quotedstring A run-time parameter for the compiler. It is a quoted string that may contain either the word "VERSION" or "version" and is used to display the compiler and library VUF number.

NOTE The formal file designators used in this command (`RPGTEXT`, `RPGOBJ`, and `RPGLIST`) 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.

Operation Notes

The RPGXL command compiles an RPG/XL program and stores the object code in a file on disk. If *textfile* is not specified, RPG/XL expects the source program to be entered from your standard input ($STDIN). If you do not specify *listfile*, RPG/XL sends the listing to your standard list device ($STDLIST). If you omit the *objectfile* parameter, the object code is saved in the temporary file domain as $OLDPASS. To keep it as a permanent file, you save $OLDPASS under another name.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles an RPG/XL program entered from your standard input device and stores the object code in the object file $OLDPASS. The listing is then sent to your standard list device.

```
RPGXL
```

The next example compiles an RPG/XL program contained in the disk file RPGSRC, and stores the object code in the object file MYRPGOBJ. The program listing is stored in the disk file LISTFILE.

```
RPGXL RPGSRC,MYRPGOBJ,LISTFILE
```

NOTE Program development in native mode uses the MPE/iX LINK command not the MPE V/E PREP command. This produces a significant difference in the method of linking code.

If you have created an RPG program called MAIN and a FORTRAN subprogram, for example, called SUB (each contained in a separate file) you might choose to handle it this way in MPE V/E:

```
:RPG MAIN, SOMEUSL
 FTN SUB, SOMEUSL
:
 PREP SOMEUSL, SOMEPROG
:
 RUN SOMEPROG
```

The second command appends the code from SUB to SOMEUSL.

However, LINK (in MPE/iX native mode) does not append SUB. On MPE/iX, you must compile the source files into separate object files and then use the Link Editor to link the two object files into the program file, as in this example:

```
 RPGXL MAIN, OBJMAIN
 FTNXL SUB, OBJSUB
:
 LINK FROM=OBJMAIN,OBJSUB;TO=SOMEPROG
:
 RUN SOMEPROG
```

However, if an NMRL is used instead of an NMOBJ, the above can be simplified to the following:

```
BUILD RLFILE;DISC=10000;CODE=NMRL
RPGXL MAIN, RLFILE
FTNXL SUB, RLFILE
LINK RLFILE,SOMEPROG
RUN SOMEPROG
```

Related Information

Commands	RPGXLGO, RPGXLLK
Manuals	*HP RPG/XL Programmer's Guide*
	HP RPG/XL Reference Manual
	HP RPG Utilities Reference Manual

RPGXLGO

Compiles, links, and executes an RPG/XL program. RPG/XL is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if RPG/XL is installed on your system. (Native Mode)

Syntax

RPGXLGO[*textfile*] [,[*listfile*]]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

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 $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 RPGLIST. Default is $STDLIST.

NOTE The formal file designators used in this command (RPGTEXT and RPGLIST) 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.

Operation Notes

The RPGXLGO command compiles, links, and executes an RPG/XL program. If *textfile* is omitted, RPG/XL expects input from your standard input device. If you do not specify *listfile*, RPG/XL sends the program listing to the formal file designator RPGLIST (default is $STDLIST).

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. The Link Editor purges the object file and writes the linked program to $OLDPASS, which is then executed and may be executed repeatedly.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Example

To compile, link, and execute an RPG/XL program entered from your standard input device, with the program listing sent to your standard list device, enter:

```
RPGXLGO
```

To compile, link, and execute an RPG/XL program from the disk file RPGSRC and send the program listing to the file LISTFILE, enter:

```
RPGXLGO RPGSRC,LISTFILE
```

Related Information

Commands	RPGXL, RPGXLLK
Manuals	*HP RPG/XL Programmer's Guide*
	HP RPG/XL Reference Manual
	HP RPG Utilities Reference Manual

RPGXLLK

Compiles and links an RPG/XL program. RPG/XL is not part of the HP 3000 Series 900 Computer System Fundamental Operating Software and must be purchased separately. This command is recognized only if RPG/XL is installed on your system. (Native Mode)

Syntax

RPGXLLK[*textfile*] [,[*progfile*] [,[*listfile*]]]

NOTE This command follows the optional MPE/iX command line syntax. Refer to "Optional Format for MPE/iX Commands" at the beginning of this chapter.

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 file to which the linked program is written. When you omit *progfile*, the MPE/iX Link Editor creates the program file, which is stored in the temporary file domain as $OLDPASS. If you do create your own program file, you do so by specifying a nonexistent file in the *progfile* parameter, in which case a job/session permanent file of the correct size and type is created.

If you name an existing program file (file code = NMPROG), that file is purged before the new one of the same name 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 RPGLIST. Default is $STDLIST.

NOTE The formal file designators used in this command (RPGTEXT and RPGLIST) 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.

Operation Notes

The RPGXLLK command compiles and links an RPG/XL program into a disk file. If you do not specify *textfile*, RPG/XL expects your input from your standard input device. If you do not specify *listfile*, RPG/XL sends the listing output to your current list device.

The object file created during compilation is a system-defined temporary file, $NEWPASS, which is passed directly to the Link Editor as $OLDPASS. The Link Editor overwrites *progfile* and writes the linked program to $OLDPASS, if *progfile* is omitted, which can then be executed.

NOTE This command is implemented as a command file. If you set the HPPATH variable to null (SETVAR HPPATH ""), the command file is not executed, and the command fails.

Use

This command may be issued from a session, job, or program. It may not be used in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following example compiles and links an RPG/XL program entered through your standard input device and stores the linked program in the file $OLDPASS. The listing is printed on your standard list device.

```
RPGXLLK
```

To compile and link an RPG/XL source program from the source file RPGSRC, store it in RPGPROG, and send the listing to your standard list device, enter:

```
RPGXLLK RPGSRC,RPGPROG
```

Related Information

Commands	RPGXL, RPGXLGO
Manuals	*HP RPG/XL Programmer's Guide*
	HP RPG/XL Reference Manual
	HP RPG Utilities Reference Manual

RUN

Executes a prepared or linked program. (Native Mode)

Syntax

The only required parameter is *progfile*. If you specify any other parameters, they will override the default parameters that the creator of the program established, but only for that particular execution of the program. If run is *implied*, see operation note below.

```
RUN progfile[ ,[ "] entrypoint[ "] ]
[;NOPRIV] [ ;LMAP] [ ;DEBUG] [ ;MAXDATA=maxstack] [ ;PARM=parameternum]
[;STACK=stacksize] [ ;DL=dlsize] [ ;NMSTACK=nmstacksize] [ ;NMHEAP=nmheapsize]
[;LIB= {  G  P  S } ] [ ;XL="library[ , ...] "] [ ;NOCB]
[;INFO="quotedstring"] [ ;UNSAT=[ "] unsatproc[ "] ]
[;STDIN=[ {  *formaldesig  fileref  $NULL } ] ]
[;STDLIST=[ {  *formaldesig  fileref[ ,NEW]  [ $NULL] } ] ]
[;PRI={  BS  CS  DS  ES } { #} ]
```

Parameters

progfile — The name of the program file to be executed. If the name is not fully qualified, it is given a full qualification consistent with the current job domain. The file may be redirected with a file equation.

entrypoint — Program entry point where execution is to begin. It contains a character string 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 the program, or any secondary entry point in the program's outer block. Default is the primary entry point.

By default, MPE/iX shifts all alphabetic characters in *entrypoint* to uppercase; surrounding the parameter with quotation marks (" or ') prevents MPE/iX from performing the upshift and permits you to enter strings for case-sensitive applications.

`NOPRIV` — Specifies that the pages of the code space of the program are to be assigned execution level 3 (the least-privileged execution level), regardless of the declared execution level. The execution level of pages in a library are not affected by the `NOPRIV` parameter. The default is that code in the program executes at its declared execution level.

`LMAP` — Indicates that the user wants a listing of the process describing the spaces occupied by the process and by the links created to bind the external references of the process. The load map is written to the loader list device. The default is not to print a load map. Load maps for compatibility mode and native mode are significantly different from each other.

Native Mode

The load map for a native mode program or library is a listing that describes the spaces loaded for a process and the linkages used to connect the external references of the process. When the lmap option is selected at run time, the listing is produced for the program and for each library specified by the user.

The load map is organized into two major areas: the SOM's Description area, with one per loaded SOM, and the Process Data Dictionary area.

Each SOM Description Area has six sections:

- The name section.
- The locality name section.
- The export code symbols section.
- The import code symbols section.
- The export data symbols section.
- The import data symbols section.

The above description is true for the program file and all user-supplied library files, but not for the subsystem library XL.PUB.SYS. The SOMs loaded from the subsystem library are now displayed in the load map. However, only the name section is written except for subsystem SOMs that have Shared Globals, in which case the export and import data symbols sections are written to the load map.

SOM Description area

```
Name Section.
NM Program File : REALP.CMARTCLE.CICSNM
Module Name     : REALS
FSN             :    0
SOM             :    0

LP              :      240.40100000
DP              :      240.41635000

Shared Data     :  YES
```

The first line of the load map from the name section displays the type of the file (program or library) and the full name of the file. The title is followed by the module name of the loaded SOM. The next grouping of items is the File Sequence Number (FSN) and the SOM number. The FSN is the number given the file according to its location in the ordered list of files presented to the loader. Starting with the number zero, which is assigned the program file, each user library is given the next number as it is encountered in the binding sequence. SOMs are numbered according to their position in the library file. This value is given by the Link Editor and read by the Loader.

The FSN and SOM number are useful when using the Process Data Dictionary area of the load map. They identify the file and SOM to which the data export belongs.

The next grouping is the LP and DP. The LP is the pointer to the Cross Reference Table (XRT), which contains the plabels for external procedure calls for this module. The DP is the pointer to the Static Global Data area for this module. The notation used for an address has the form: sid.offset.

The sid (space ID) is the 32-bit virtual space number that was assigned for that space when it was loaded. The offset is the byte offset within the space relative to its beginning.

The next grouping shows the condition of the shared global flag for this module. This information is only shown if the flag is set true.

Locality Name Section

```
Locality Name        Type       Address        Length  XL|R/W
-               -
$LIT$                Code       2C5.5000          348   3
$UNWIND_START$       Code       2C5.5348           74   3
$DXRT$               Data       240.41634000     1000   3/0
$GLOBAL$             Data       240.41635000       E8   3/3
```

The name section is followed by the section that describes the spaces declared with the module.

The new subspace is the $DXRT$, which is the Data Cross Reference Table. Its address points to the bottom of the DXRT. Entry into the DXRT is negatively offset from the beginning of the Static Global area, which is the address of the $GLOBAL$ subspace.

The valid types for subspace are: Code, Data and Common. The length column is the number of bytes in hexadecimal format. The last column is read in two ways: for Code subspaces, it is the execution level; for Data subspace, it is R-read access, W-write access.

Export Code Symbols Section

```
Entry Name           Type   Proc Addr     Stub Addr     XL/EL
              -     -     -
$START$              PProg                2C5.5014       3/3
main                 Entry  2C5.50BC      2C5.5050       3/3
```

The valid types for export code symbols are:

Entry Any code entry point. Includes both primary and secondary entry points that may be used as targets of r-space calls.

PProg Primary program entry point.

SProg Secondary Program entry point.

The procedure address (Proc Addr) column gives the starting address of the procedure. The stub address (Stub Addr) column gives the (inbound) external call stub. The last column is interpreted as follows: XL-execution level and/or EL-the call execution level.

Import Code Symbols Section

```
External Name         Type          XRT           Stub Addr
-                   -           -
printf                Stub            4           2C5.506C
proca                 Stub            3           2C5.509C
.
.

exit                Stub            A           2C5.5294
```

The valid types for import code symbols are:

Stub This symbol marks an import (outbound). The Link Editor creates an import stub for the unsatisfied code symbols, and the Loader satisfies the reference by filling in the XRT entry allocated for this stub.

Plabl This symbol defines an export stub for a procedure for which a procedure label has been generated. The Loader builds an XRT entry for the procedure at the offset allocated by the Link Editor.

The XRT column specifies the entry in the XRT through which the contents of a plabel can be located. Each entry is 32 bytes. The stub address (Stub Addr) column is the outbound stub address. This stub accesses the XRT for the targeted export.

Export Data Symbols Section

```
  Symbol Name         Select Type Scope   Size     DP Addr       R/W
  -           -        -       -
  a                      YES Stor  Univ      8     240.416350E0  3/3
  b                      YES Data  Univ    n/a      240.41635000  3/3
```

The Select column indicates whether this particular export was the one chosen by the Loader to place in the PDD.

The valid types for export data symbols are:

Data Normal initialized data. Example (a C construct): `double b = 3.3;`

Stor Storage. This symbol requests a data storage location of a certain size.

The scope column is always Univ-universal.

The Size column shows the number of bytes in decimal format required for the export symbol. Space is allocated for four (4) characters only. To accommodate numbers greater than 9999 bytes, the format changes to 10k up to 999k (999 kilobytes). The next range is 1.0m up to 9.9m (9.9 megabytes), followed by 10m to 999m (999 megabytes), and finally, 1.0g to 4.2g (4.2 gigabytes). Size information is only available for storage request types. There is no size information available for initialized data, that is, data universals.

The DP Addr column contains the actual virtual address of the symbol, provided the Select column is YES.

The last column gives the access rights for the symbol.

Import Data Symbols Section

```
  IMPORTED DATA SYMBOLS .....
  Symbol Name             Type Scope   DXRT      DXRT Addr     R/W
  -              -         -     -
  c                       Data Unsat     -C      240.41634FE4  3/3
  d                       Data Unsat    -14      240.41634FEC  3/3
  ANSI_MODE               Data Unsat    -18      240.41634FE8  3/3
  a                       Data Unsat    -10      240.41634FF0  3/3
```

The valid type for import data symbols is:

Data Requested import data item. Example (a C construct): `extern double c`

The scope column is always UnsatImport request has not been satisfied.

A DXRT entry is indexed negatively from the DP of the SOM. The DXRT column gives this offset, which is in bytes. The value is in hexadecimal format. The DXRT Addr column gives the indirect address for the import symbol.

The last column gives the access rights for the symbol.

Process Data Dictionary Area

```
|||||||||||||||||||||||||||||||||||||||||||||||||||||
|||                                               |||
|||            PROCESS DATA DICTIONARY            |||
|||          SHARED GLOBALS DATA EXPORTS          |||
|||                                               |||
|||||||||||||||||||||||||||||||||||||||||||||||||||||

Symbol Name   FSN  SOM Type Scope Size   DP Addr       R/W
-   -  -       -       -
a               0    0 Stor  Univ    8   240.416350E0  3/3
b               0    0 Data  Univ  n/a   240.41635000  3/3
d               1    0 Stor  Univ    8   240.416370A8  3/3
c               1    0 Data  Univ  n/a   240.41637000  3/3
.

.


__ANSI_MODE     2    0 Stor  Univ    4   240.41641894  3/3
.

.
```

The FSN (File Sequence Number) and the SOM columns can lead you to the file and SOM, which supplied the export data symbol. For example, the `_ANSI_MODE` symbol comes from the subsystem library in the binding sequence, which would be `XL.PUB.SYS`, and the first SOM (SOM 0) with module name hp30026_01. Shown below are some lines from the SOM Description Area of the load map for the subsystem library.

```
NM Library File : XL.PUB.SYS
Module Name     : hp30026_01

FSN             :    2
SOM             :    0

LP              :      240.401001A0
DP              :      240.41639000

Shared Data     :  YES

EXPORTED DATA SYMBOLS .....
Symbol Name         Select Type Scope   Size     DP Addr       R/W
-           -        -       -
.

.


__ANSI_MODE            YES Stor  Univ      4     240.41641894  3/3

.

.
```

Continuing with the PDD area, the remaining columns starting with Type through R/W are interpreted in the same manner as explained in the Export Data Section.

Compatibility Mode

A compatibility mode loader map shows information on the origin and destination of the reference. The exact origin or destination is identified by the file type, the segment within the file, and by the STT entry of the segment. The level of parameter checking is also listed. For example:

```
  PROGRAM FILE SAMPLE.LOADER.MPEXL
  TERMINATE      PROG  0    4   0 SSL  0    2   41
  GETUSERMODE    PROG  0    3   0 SSL  0   13   44
  GETPRIVMODE    PROG  0    2   0 SSL  0   14   44

  301
```

The first entry reading across lists the name. The next four entries show the information for the reference origin. The last four show the information for the reference destination:

```
                 Reference Origin Reference Destination
                 F T   L    S   S F T  L    S   S
                 i y   C    T   e i y  C    T   e
                 l p        T   g l p       T   g
                 e e              e e
  TERMINATE      PROG  0    4   0 SSL  0    2   41
  GETUSERMODE    PROG  0    3   0 SSL  0   13   44
  GETPRIVMODE    PROG  0    2   0 SSL  0   14   44

  301
```

The file types are:

PROG	Compatibility mode program file	
SSL	SL.PUB.SYS	
PSL	SL.PUB.*account*	
GSL	SL.*group.account*	
LC	(Level of file checking):	
	0	No checking
	1	Check procedure type
	2	Check number of parameters
	3	Check parameter type

STT is the segment transfer table entry within the segment.

Seg is the logical segment number of the segment.

A list of the CSTX numbers (the single number 301 in this example) assigned to the segments of the program follows the load map. The first number in the list corresponds to logical Seg 0, the second to logical Seg 1, and so on.

DEBUG Instructs the process to enter the system debugger just before executing the first instruction of the program. Once the debugger has been invoked, the commands available to the user depend upon the user's assigned capability. The default is not to enter the system debugger. This parameter is ignored in a job.

maxstack The maximum CM stack area (Z-DL) size permitted, in 16-bit words. This parameter is included if you expect the size of the DL or the Z-DB areas to be changed during the program execution. But no matter what you specify, MPE/iX may change *maxstack* to accommodate table overflow conditions.

A value of -1 or a + sign (interpreted as a zero) causes the default value to be used.

The *maxstack* is always equal to the compatibility mode maximum default size if *progfile* is a native mode program.

parameternum A value that can be passed to the program as a general parameter for control or other purposes. If the parameter is not specified, the default value is zero (0). If the executing program is a compatibility mode program, Q(initial)-4 contains the parameter value. Note: Q relative addresses are 16-bit word addresses. Q(initial) is the Q address for the outer block of the program.

MPE/iX provides an intrinsic (`GETINFO`) for retrieving the `PARM` parameter for a native mode process.

stacksize The size of the CM local area, Z-Q, in 16-bit words. This value, if specified, must be in the range 512 to 32,767. It overrides the default stack size estimated by the MPE segmenter.

The *stacksize* is always equal to the compatibility mode maximum default size if *progfile* is a native mode program.

dlsize The DL-DB area to be assigned initially to the CM stack. To accommodate system logging requirements, this 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 16-bit words.

This value must be in the range -1 to 32,767. The default (which is used when no value or an invalid value is specified) is estimated by the MPE segmenter. A + sign for this parameter is interpreted as a zero.

The *dlsize* is always equal to the compatibility mode maximum default size if *progfile* is a native mode program.

nmstacksize The maximum size in bytes to which the NM stack may grow. This must be a decimal number. If a value is specified which is less than the system-defined minimum (including values <= 0), the system-defined value will be used. If a value is specified which is greater than the system-defined maximum value, the system-defined maximum value will be used. A + sign for this parameter is interpreted as a zero.

The default is -1, which currently instructs MPE/iX to assign a system-defined constant as the value of *nmstacksize*.

nmheapsize The maximum size, in bytes, to which the NM heap may grow. This must be a decimal number. If a value is specified which is less than the system-defined minimum (including values <= 0), the system-defined value will be used. If a value is specified which is greater than the system-defined maximum value, the system-defined maximum value will be used.

The default is -1, which currently instructs the command to assign a system-defined constant as the value of *nmheapsize*. A + sign for this parameter is interpreted as a zero.

`G`, `P`, or `S` These parameters provide an efficient way to specify the executable libraries that may be used to load the program.

- `G` The program's group library is searched first, then its public account library is searched, and finally the system library is searched to resolve the program's external references.
- `P` The program's public account library is searched before the system library is searched to resolve the program external references.
- `S` Only the system library is used to bind the external references of the program. This is the default.

These parameters will result in a fail load if *progfile* contains a program name which cannot be expressed using the MPE syntax.

The group and account libraries referenced by this parameter must be named `SL`.*group.account* for compatibility mode programs and `XL`.*group.account* for native mode programs. Group and account are the group and account of the program, where the program resides.

If the `LIB` and `XL` parameters are missing, this parameter defaults to `S`. This parameter may not be used at the same time as the `XL` parameter.

"*library*" Specifies the library or libraries to be searched, and the order in which they are searched to resolve any external references. This parameter is available only for native mode load operations. It may not be used at the same time as the `LIB` parameter. It must be delimited by a matching pair of quotation marks (either `"` or '). Compatibility mode ignores this parameter if it is specified. In native mode, this parameter overrides `LIB=` if both are specified.

If any library name in the list is not fully qualified, it will be qualified with a name consistent with the program file being loaded. Library names, except those in the system library, may be redirected with a file equation.

A default value for this parameter may be stored in the program file. The default is used only if the `LIB` and `XL` parameters are both omitted.

In a list of libraries, each library must have a privilege level equal to or greater than the privilege level of the library that precedes it in the list. The privilege level of any file is governed by the privilege level of the group in which it resides. For example,

RUN PROGA.*grp.acct*;XL='LIB1.PUB.TOOLS,LIB2.DIAG.SYS'

Suppose the group *grp* does not have privileged mode (PM) capability. We assume for this example that the user is able to execute PROGA.*grp.acct*. Suppose also that PUB.TOOLS does have PM capability, but that DIAG.SYS does not.

The program PROGA.*grp.acct* is able to load PUB.TOOLS. But PUB.TOOLS has PM capability. Therefore everything following it in the list must have PM capability, too. Since DIAG.SYS does not, the library search ends without loading LIB2.DIAG.SYS.

This prevents non-PM processes from "piggybacking" on legitimate PM processes.

NOTE XL.PUB.SYS and NL.PUB.SYS, which are two of the three system libraries for MPE/iX, are searched automatically. The user does not need to specify them.

If you do specify one or both, place them at the end of your list of libraries. Otherwise, MPE/iX detects an error.

If you specify NL.PUB.SYS but not XL.PUB.SYS, only NL.PUB.SYS is searched. XL.PUB.SYS is ignored in this particular case. However, if you specify XL.PUB.SYS but not NL.PUB.SYS, both are searched despite the omission of NL.PUB.SYS.

An absolute pathname must be used when a library name is specified in HFS syntax. In addition, if *progfile* contains a name which can only be expressed in the HFS syntax, the file names specified in this item must be fully qualified.

To have an XL in the HFS, you must copy it from the MPE group to the HFS directory.

NOCB Instructs the file system not to use the stack segment, PCBX, for its control blocks, even if sufficient space is available. This allows for expansion of the stack, using the DLSIZE and ZSIZE intrinsics, to the maximum possible limit at a later time.

NOCB affects only those programs that use the following types of file: MSG, RIO, and CIR. Programs using other types of files ignore the NOCB parameter.

Be aware, that NOCB causes the file management system to operate more slowly.

quotedstring Allows the user to pass an ASCII string to the program that is to be run. The string must be delimited by a matching pair of quotation marks (either " or '). If you want a quotation mark to appear within the string, you may double it, as with most programming languages: can't must appear as cant", " and " must appear as ""and"", 'but' must appear as but''". The maximum length of the string, including delimiters, is 255 characters. Refer to "Examples."

If the executing program is a compatibility mode program, Q(initial)-5 contains a byte pointer to the string, and Q(initial)-6 contains the number of characters in the string. The Q-relative addresses are 16-bit addresses. Q(initial) is the Q address for the outer block of the program. Default is that no string is passed, and the length of the string is set to zero.

MPE/iX provides an intrinsic (GETINFO) for retrieving the *quotedstring* for a native mode or compatibility mode process.

unsatproc Specifies the (fall-through) procedure that is linked in the event that any of the external references cannot be resolved to one of the libraries available to the process. This is available only when loading a native mode program. It is ignored when loading a compatibility mode program. By default, MPE/iX shifts all alphabetic characters in *unsatproc* to uppercase; surrounding the parameter with quotation marks (" or ') prevents MPE/iX from performing the upshift and permits you to enter strings for case sensitive applications.

For instance:

```
;UNSAT = terminate
```

The procedure TERMINATE is linked if one of the external references cannot be resolved to one of the available libraries. Because the value terminate is not delimited by quotation marks (" or '), the value is upshifted to TERMINATE.

```
;UNSAT = "foo"
```

```
;UNSAT = 'foo'
```

Here the procedure foo is linked if one of the external references cannot be resolved to one of the libraries. In both cases, delimiting the value foo with quotation marks (" or ') causes MPE/iX to use the value as given, in lowercase.

If the user does not supply an UNSAT procedure and a process cannot be fully bound, the load fails.

An UNSAT procedure *must* reside in an XL. The UNSAT procedure *cannot* be placed in an NMOBJ file and linked with the rest of the program.

STDIN Specifies 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=**Return**, STDIN defaults to the standard input device for the job or session.

formaldesig The formal file designator for a file previously specified in a file equation.

fileref The name of an existing permanent or temporary disk file.

`$NULL` The actual file designator of a system-defined file that is always treated as an empty file. When referenced by another program, a program receives only an end-of-file indication when accessed. When referenced by a program as `$STDLIST`, the associated write request is accepted by MPE/iX, but no physical output is actually performed. Thus, `$NULL` can be used to discard unneeded output from an executing program.

`PRI` The execution priority that the command interpreter uses for your program. BS has the highest priority; ES has the lowest priority.

DS and ES are intended for batch jobs and are not well-suited for interactive applications. Specifying a positive integer (#) permits you to set priority at points that lie between the preset priority levels BS, CS, DS, and ES. Accepted values for # are in the range 100 to 255, inclusive. Refer to the `CREATEPROCESS` intrinsic in the *MPE/iX Intrinsics Reference Manual*.

If you are in user mode (that is, nonprivileged) you may specify BS, CS, DS, or ES.

If you attempt to specify a priority higher than the priority permitted for your account or user name, MPE/iX sets the highest priority below BS. The default is CS. If you do not specify a value the default (the parent process's dispatching subqueue priority) is used.

CAUTION Use care in assigning the BS queue. Processes at the BS priority can lock out other processes.

`STDLIST` 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=`**Return**, 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` (for instance, `";STDLIST=`*filename*` ``,NEW"`).

`NEW` The name to be assigned to a job/session temporary disk file consisting of 132-byte fixed ASCII records.

Operation Notes

This command executes a program prepared in a program file. It permits searching libraries (SLs for compatibility mode, XLs for native mode) to satisfy external references. Relocatable libraries (RL) 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. The volume set has to be opened with a `VSOPEN` command.

If the program file is a temporary CM file, the logon group and account libraries for the current session along with SL.PUB.SYS are searched. If a program file is a temporary NM file XL.PUB.SYS and NL.PUB.SYS are also searched. Refer to the *Accessing Files Programmer's Guide* (32650-60010) for more information on file domains.

NOTE NM and CM loader error messages are reported differently, allowing you to determine the system in which the error occurred.

NM Loader Error: ErrMessage (``LDRERR *nnnn*)"

CM Loader Error: ErrMessage (``LOAD ERR *nnnn*)"

The RUN command is parsed by the Compatibility Mode parser unless it is *implied*, in which case the Native Mode parser is used. To use the *implied* version of RUN simply omit the word *run* and enter the name of the program along with either the INFO or PARM parameters.

Because the Native Mode parser is used with *implied* run you can use quotes (" or ') with the program file name and/or the ;INFO= parameter. Also, quotes are not required if the parameter contains no delimiter characters such as a blank, comma, semicolon, quotemarks or equal sign. In addition, the ;INFO string can be up to 280 characters long and the ;PARM= value can be any signed 31 bit number. Without *implied* RUN the ;INFO limit is 255 characters and the ;PARM= value is limited to a signed 15 bit decimal or unsigned 16 bit octal or hex value.

NOTE Programs whose name cannot be expressed using MPE syntax are not allowed to have PM, MR or DS capability. Programs linked with these capabilities cannot be loaded.

Users must have PM capability to load programs whose name cannot be expressed using MPE syntax, with PM capability.

CM programs cannot be loaded from the HFS directory.

Use

This command may be issued from a session or a job. It may not be issued in BREAK or from a program, unless the user or the calling program has PH capability. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

To list the references 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 disk file named INPUT and $STDLIST set to the line printer:

```
FILE LPFILE;DEV=LP
RUN TESTPROG; STDIN=INPUT;&
STDLIST=*LPFILE
```

The next example runs a program using the STDIN parameter, setting $STDIN to an existing disk file named INPUT, this time referenced through a file equation. To set $STDLIST to a temporary disk 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;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
```

Related Information

Commands	LINK, PREP, XEQ, VERSION Utility
Manuals	CREATEPROCESS intrinsic in the *MPE/iX Intrinsics Reference Manual*

7 Command Definitions S-SO

SAVE

Saves a file in the 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.

newfile-reference New actual file designator assigned to $OLDPASS when it is made permanent. 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.

tempfile-reference 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.

Operation Notes

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.

You must specify a new *filename* for $OLDPASS, because MPE/iX does not allow $OLDPASS as a permanent file name. If there is a file in the temporary domain with the same name specified by *newfilereference,* MPE/iX attempts to save $OLDPASS by creating a new temporary file. This temporary file name, created by SAVE, starts with S and is 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 is then saved under the file name specified by *newfilereference,* and is deleted from the temporary domain. If both temporary and permanent files exist under the same name specified by *newfilereference,* the temporary SAVE file is saved as a permanent file. In this case, a printed error message states the file name for the new SAVE file. It can be renamed later using the RENAME command.

This command applies only to temporary files on disk. It is similar to opening a file with the FOPEN intrinsic, and then closing it with the FCLOSE intrinsic, using a permanent file disposition.

Use the SAVE command to save KSAM XL files. Since the KSAMUTIL utility is not supported for KSAM XL, the SAVE command is the only method of doing so.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

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 are searched, for example:

```
SAVE DATAFILE
RENAME DATAFILE,DATABASE
```

Related Information

Commands	PURGE, LISTFILE, LISTFTEMP, RENAME
Manuals	None

SECURE

Reinstates all file security provisions that you previously suspended with the `RELEASE` command.

Syntax

SECURE *filereference*

Parameters

filereference Specifies the actual file designator for which you want to reinstate file access control. The *filereference* can be either in MPE or HFS syntax.

MPE Syntax

If the *filereference* does not begin with a dot or a slash, it is parsed according to the MPE syntax and has the form:

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

If the file has a lockword, you must specify it; otherwise, the system prompts you for it. If you do not specify *groupname.acctname*, the system assumes the logon group and account.

HFS Syntax

If the *filename* begins with a dot (.) or a slash (/), it is parsed according to HFS syntax.

Operation Notes

- **Usage**

  You can use this command only for permanent disk files you created. Under default system security provisions, the file must be in your logon account and must belong to your logon or home group.

- **Checking the file status**

  You can enter the `LISTFILE` command to determine if a file is currently released or secured. Refer to the `LISTFILE` command in this book for more information.

- **Access control definition**

  An access control definition (ACD) overrides file access controls whether or not you have released or secured the file.

Use

You can enter this command from a session, a job, a program, or in break mode. Pressing **Break** does not affect this command.

Example

- To reinstate file access control previously in effect for the file named FILE1, enter:

```
:SECURE FILE1
```

Related Information

Commands	ALTSECT, LISTF, LISTFILE, RELEASE'
Manuals	None

SEGMENTER

Starts the MPE segmenter.

Syntax

SEGMENTER [*listfile*]

Parameters

listfile	Actual file designator of an 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. This must be defined in a FILE command, and then backreferenced (see "Example").

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 the "Implicit FILE Commands for Subsystems" discussion of the FILE command.

Operation Notes

This command starts the segmenter subsystem from MPE/iX. The segmenter subsystem performs the intermediate functions between source code compilation and program execution.

The segmenter employs temporary files named T999SYM, SEGTMP01, and SEGTMP00. If you create temporary files with these names, the segmenter attempts to purge them.

You must have READ and LOCK access to use a relocatable library with the SEGMENTER command.

Use

This command may be issued from a session or a job. It may not be issued in BREAK or from a program, unless the user or the MPE segmenter has process handling (PH) capability. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the 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
```

Related Information

Commands	`FILE`
Manuals	*MPE Segmenter Reference Manual*

SET

Defines elements of the command interpreter. It also allows a job using a spooled $STDLIST to mark its standard list device for deletion when the job terminates. (Native Mode)

Syntax

```
SET[ STDLIST={ DELETE | SAVE } ] [;MSG={ON | OFF}]
[ECHO={ ON| OFF}][ ;SPEED={ 300 | 1200 | 2400 | 4800 | 9600 | 19200 | 19.2K}]
```

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.

ECHO — Turns terminal echoing ON or OFF.

MSG — Specifies whether or not TELL messages are displayed on the user's terminal. MSG=OFF prevents TELL messages from appearing on the terminal. WARN messages override MSG=OFF and will appear on the terminal. (This parameter provides the same function as the SETMSG command.)

SPEED — Specifies the terminal's data transmission rate, within the upper and lower bounds outlined above. The user is responsible for manually changing the terminal's speed setting. (This parameter provides the same function as the SPEED command.)

Operation Notes

The SET command specifies several elements of the command interpreter including the terminal echo and baud rate.

In a job, 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 does not execute if the job fails. $STDLIST then prints, allowing you to study your listing and to 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.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

The following example illustrates using the SET command from within a program:

```
!JOB EXAMPLE, USER.TECHPUB,XGROUP
!CONTINUE
!RUN UPDATE.PUB.SYS;PARM=1;MAXDATA=16000
!IF JCW < FATAL THEN
!SET STDLIST=DELETE
!ENDIF
!EOJ
```

Related Information

Commands	SETMSG, SPEED, ECHO
Manuals	None

SETCATALOG

Catalogs, or enables, the user-defined commands (UDCs) in a specified catalog file at the user, account, or system level. You can also use this command to disable all UDCs on the system. (Native Mode)

WARNING **If you do not specify a *catfilename*, all UDC's are disabled (deleted from the UDC directory) regardless of whether or not the ;DELETE option is used.**

Use only MPE/iX flat files as UDC files. Issuing the SETCATALOG command for any other file type may cause unpredictable results.

Syntax

```
SETCATALOG [ catfilename[ ,catfilename, ...[ ,catfilename]]]
[ ;SHOW] [ ;SYSTEM] [ ;ACCOUNT]
[ ;USER=username[ .acctname]]
[ ;RESET][ ;APPEND][ ;DELETE]
```

Parameters

catfilename The name of a file containing user-defined commands to be cataloged. Commands within the file must be separated from each other by a line whose first character is an asterisk (*).

SHOW Specifies a listing of the user-defined commands as the UDC files are cataloged. Error messages are printed for command lines that contain any errors. This parameter is useful for locating errors in UDC files.

ACCOUNT Specifies cataloging of the file at the account level. Using this parameter requires account manager (AM) capability.

SYSTEM Specifies cataloging of the file at the system level. Using this parameter requires system manager (SM) capability.

USER Allows users who have AM capability to change the UDC catalog set for users in their account. Users having SM capabilities can change the UDC catalog set for any user. USER does not rebuild an executing UDC directory, but becomes effective when the user logs off and then logs on after the command has been invoked.

RESET Causes the file(s) being cataloged to replace all files that are already cataloged. RESET is the default if no option is specified.

APPEND Permits the user to add UDCs to the directory. This option causes the file(s) being cataloged to be appended to the existing catalog. It also finds and makes adjustments for any logon UDCs if appropriate.

DELETE Deletes the file(s) from the existing UDC directory. This permits the user to delete individual files from the catalog directory. The original order of the catalog is maintained. It also finds and makes adjustments for logon UDCs. The ACCOUNT and SYSTEM options allow the user to delete the cataloged file at the account or system levels. The default is user level.

Operation Notes

The SETCATALOG command allows you to catalog user-defined commands.

When you set your own UDCs, the change takes place in your UDC catalog immediately. If you specify the ACCOUNT or SYSTEM parameter, your UDC catalog is changed immediately, but other users in your account or system must log on again in order to have those changes available to them. If you set a UDC and specify another user (USER=), that user must log on again in order to have the changes available.

The ability to delete or append files is particularly useful because, although most UDC files do not change, new UDC commands are frequently added or modified. Using the DELETE or APPEND parameter allows you to make changes without incurring the overhead of recataloging the entire directory for every change. Grouping UDC files into functions further reduces the work involved in modifying UDCs.

The RECURSION option relieves the user of having to define a particular command more than once in a catalog set, and from having to maintain a particular order for commands within a catalog set. Refer to the discussion on options in "User Commands" in *Using the HP 3000 Series 900: Advanced Skills*.

If SETCATALOG is used in a UDC, all valid commands through and including the SETCATALOG command execute. But execution of the UDC terminates after the execution of the SETCATALOG command. Commands that follow do not execute. The SETCATALOG command does not have this effect when executed in a command file.

The SETCATALOG command may be invoked only from the logon command interpreter (user main), where it is passed through the scanner/parser. . It cannot be invoked from any other program (any child process).

Use

This command is available in a session, job, or in BREAK. It is not available from a program. Pressing **Break** has no effect on this command.

Examples

The following command sets the UDC directory for the user JOHN.WORKERS with the commands in the file named UDCA. The USER option cannot be specified with the ACCOUNT or SYSTEM options. Attempting to do so produces an error.

```
SETCATALOG UDCA; USER=JOHN.WORKERS
```

The following two command sequences are equivalent:

```
SETCATALOG UDCA, UDCB

SETCATALOG UDCA
SETCATALOG UDCB ;APPEND
```

In the first example, the command has an implied RESET, and thus overwrites the previous file set in the directory. In the second example, UDCA is entered into the directory, and then UDCB is appended to the directory without affecting UDCA. It also finds new logon commands if appropriate.

The following command deletes UDCA from the directory at the account level, provided it was cataloged at the account level. If other account-level UDCs reside in the directory along with UDCA, they remain undisturbed by this deletion. When appropriate, a new logon UDC is set up.

```
SETCATALOG UDCA ;DELETE ;ACCOUNT
```

It is *not* a good practice to create UDC's which have the same name as other files, especially *command* files or any other files your users may confuse with UDC's.

If you enter a fully qualified file name that has the same name as an existing UDC, the group and account part of the fully qualified name are passed to the UDC as a parameter. For example, if COMM is a UDC, entering COMM.GROUP.ACCT will cause .GROUP.ACCT to be passed to COMM as a parameter even if COMM.GROUP.ACCT is a separate file.

Related Information

Commands	SHOWCATALOG, HELP <udcname>
Manuals	*System Startup, Configuration, and Shutdown Reference Manual*
	Using the HP 3000 Series 900: Advanced Skills

SETCLOCK

Alters the system time or system time zone.

SYNTAX

```
SETCLOCK{DATE=date spec; TIME=time spec [ ;GRADUAL | ;NOW]}
{CORRECTION= correction spec}
{TIMEZONE= time zone spec}
{ ;CANCEL}
```

Parameters

date spec A specification of local date in the form *mm/dd/yy[yy]*. The year may be expressed in two or four digits. If a date is provided, a time must also be provided.

time spec A specification of local time in the form *hh:mm[:ss]* where seconds are optional. This specification uses a 24-hour clock; it is not permissible to specify the time using A.M. or P.M. If a time is provided, a date must also be provided.

The operating system will experience problems if the system date and time are too close to the base time of midnight, January 1, 1970. Therefore, for proper system operation this command requires the date and time to be later than ten minutes past midnight on January 1, 1970.

correction spec An integer specifying the desired change in the system time. The units are seconds. Thus a positive correction will cause the system clock to advance by the specified number of seconds, while a negative correction will cause the system clock to slow by the specified number of seconds.

time zone spec A specification of the time zone in the form *hh:mm*, preceded by a required "W" or "E" to specify the Western or Eastern Hemisphere. Thus a specification of W7:00 represents a seven-hour displacement from Universal Time (GMT) with the time zone being in the Western Hemisphere.

Providing a *time zone spec* is the only way to change the system time and maintain both local and Universal Time (GMT) accurately. See the Operation Notes section for details.

`GRADUAL` This option is meaningful only when the date and time specifications are provided. `GRADUAL` causes the system clock to speed up or slow down until the time change is completed, at which time the system clock will resume its normal pace. `GRADUAL` is the default for the Date-Time form of the command.

`NOW`	This option is meaningful only when the date and time specifications are provided. `NOW` forces the change to be immediate. See the warning in the Operation Notes section about the dangers of changing the system time immediately.
`CANCEL`	Cancels a current time correction. Any correction which has already taken place before the cancellation will remain; this option does not undo a correction which has already been accomplished. See the Operation Notes and Examples sections for details.

Operation Notes

The `SETCLOCK` command is used to change the system time or to change the system's time zone.

Changing the system time or time zone does not affect any interval timers in effect. Thus, a PAUSE for a given time duration will maintain that same duration regardless of how the system time is changed.

Changing the system time or time zone will cause any jobs streamed with a time specification (;AT=, ;DAY=, ;DATE= or ;IN=) to be introduced in accordance with the newly-changed system time. Thus, a job streamed with ;AT=9:00 will be introduced when the changed system time is equal to 9:00.

The user may provide SETCLOCK with a date and time, a time correction, or a time zone. The Date-Time form, the Correction form, and the Time Zone form are mutually exclusive; for instance, the user may not provide specifications for both a time correction and a time zone in a single command.

The Date-Time and Correction forms of the command are intended for slight adjustments of the system time. For example, these forms would be used to move the time forward or backward slightly in order to keep the system time synchronized with an external time source. Both local and Universal (GMT) time are adjusted.

The Time Zone form of the command is intended for the larger time changes required to move the system to a new time zone, such as moving between Standard Time and Daylight Savings Time. This form of the command alters the local time without changing Universal Time.

Date-Time: If the Date-Time form of the command is used, the system time is adjusted to the specified date and time. This adjustment is gradual by default. It may be made immediate if `;NOW` is specified and the user has System Manager (SM) capability.

Correction: If the Correction form of the command is used, the system time is adjusted forward or backward by the amount of the correction. This adjustment is always gradual.

Time Zone: If the Time Zone form is used, local time is adjusted to match that of the specified time zone. In addition, the system time zone offset is changed to reflect the new time zone.

The Use of The Time Zone Offset

On the HP3000 Universal Time (GMT) is calculated by starting with local time and adding or subtracting a time zone offset. When changing time zones (such as moving from Standard to Daylight Savings Time and back) the local time is altered, but this change must not affect Universal Time. To prevent Universal Time from being altered, both the local time and the system time zone offset must be adjusted. **Therefore, using the Time Zone form of this command is the only way to accurately change time zones.**

If the Date-Time or Correction form of the command is used, Universal Time will drift along with local time. Thus, the Date-Time and Correction forms of this command should only be used to adjust the clock for drift, not to change time zones.

Results of the Time Zone Form

- If the change in time zone is to a later time (a change to Daylight Savings Time or an "Eastern" geographic movement), both local time and the time zone offset are changed immediately.

  The effect is that users of local system time will see an immediate jump forward to the new time zone, while users of Universal Time will see no change.

- If the change in time zone is to an earlier time (a change from Daylight Savings to Standard Time or a "Western" geographic movement), the time zone offset is changed immediately. Then the local time slows down until the system time corresponds to the time in the new time zone.

  The effect is that users of local system time will see a gradual slowdown to match the new time zone, while users of Universal Time will see an immediate forward jump, then a slowdown until the system time again matches "real" Universal Time.

This method of changing time zones ensures that no out-of-sequence time stamps will occur either in local time or in Universal Time.

How a Gradual Time Change Works

Whether the Date-Time or a Correction form is used, the default method of changing the time is to gradually speed up or slow down the system clock until the change is achieved. Thus, even when a previous time is requested, the system clock will still move forward, although at a slower pace than real time. This slower pace will continue until the desired time "catches up" with the system clock. Because of the system clock's forward motion, there will never be a case where two consecutive timestamps appear to be out of sequence and where system time appears to run backwards.

This change in clock speed is accomplished by establishing a system time correction which is gradually consumed. During this time the system clock speeds up or slows down as necessary. When the correction reaches zero, the system clock resumes its normal pace. The rate of the correction depends on the load on the system. The correction rate will be slowed down by frequent timestamp requests, file accesses and frequent operating system activity such as context switches. In general, the correction will take no longer than twice the requested time difference. For example, a request to slow down the clock by one hour will take a maximum of two hours to complete.

Results of the ;CANCEL Parameter

Any time during an on-going correction, issuing this command with the ;CANCEL parameter will immediately set the correction to zero and cause the system clock to resume its normal pace. Any previous correction will remain. When this option is used, the system will report the amount of correction which was cancelled.

How a System Time Change Affects Accounting Information

Changing the system time, even gradually, may cause accounting CONNECT-MINUTES to be distorted. Anyone logging on before the change and then logging off after the change is completed will have their accounting CONNECT-MINUTES data distorted; if the time change is forward, CONNECT-MINUTES will be increased by the amount of the time change, and if the time change is backward, CONNECT-MINUTES will be decreased by the amount of the time change.

Dangers in Using the ;NOW Parameter

The ;NOW parameter permits immediate forward or backward time changes. However, several dangerous situations can occur:

- Any applications which rely on the forward progression of time may give inconsistent results if the time is immediately set backwards. Such applications include the processing of timestamped transactions in which the sequence of those transactions is important.
- In order to recover data in case of an unexpected hardware or software failure, some applications require that the system time must never seem to go backwards. For instance, some applications log transactions to a circular file. These transactions are timestamped, and if the transactions must be recovered, the recovery program determines the end of data by looking for timestamps which are out of sequence. If the system time is set backwards immediately, transactions which occur after the time change may not be recovered. Therefore, do not set the time backwards using the ;NOW option if there are applications which log their transactions using timestamps.
- Accounting CPU-SECONDS data may be distorted. The user whose process was active during an immediate forward or backward change might seem to have a CPU-SECONDS time which is an extremely large positive or negative number.
- STORE/RESTORE, TurboSTORE/XL, or any other file archive system based on dates or times may not store or restore the files in the expected manner, since some files may have creation or access times in the future or may even have access times which precede their creation times.
- Some compilation tools like MAKE rely on the relative modification dates of the files in the compilation unit. Setting the system time backward and then modifying the main file in the compilation unit may force an unnecessary full compilation, since the main file may have an earlier modification time than the files it depends on. Setting the system time backward and then changing a file needed by the main file will cause MAKE to think that the changed file's modification time precedes that of the main file. Thus, the changed file will not be included in the recompilation.

This list is only meant to include a few of the dangers associated with an immediate time change; this list does not represent all of the problems likely to be encountered. **Therefore, if the ;NOW option must be used, it should be used only with a full knowledge of its effects on the system's workload.**

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Diagnostician (DI) and either Operator (OP) or System Manager (SM) capabilities are required to issue this command. Additionally, System Manager (SM) capabiltiy is required to use the ;NOW parameter.

Examples of Date-Time and Correction Forms:

The following example illustrates setting the system time by providing a date and time:

```
:SETCLOCK DATE=07/04/1993;TIME=15:00
```

The following example illustrates providing a time correction to advance the system time by one hour.

```
:SETCLOCK CORRECTION= +3600
```

or

```
:SETCLOCK CORRECTION= 3600
```

Both of the above examples cause Universal Time (GMT) to change as well as local time, and therefore while they are useful in correcting the system time for drift (time gain or loss), they are not accurate ways to change time zones.

The following example illustrates setting a time correction, executing a SHOWCLOCK command, cancelling the correction, then again executing a SHOWCLOCK command. Note that by the time of the first SHOWCLOCK the correction has already begun to be consumed.

```
:SETCLOCK CORRECTION= -3600

:SHOWCLOCK

SYSTEM TIME: FRI, JUL 24, 1987, 8:47:35 AM
CURRENT TIME CORRECTION: -3568 SECONDS
TIME ZONE: 7 HOURS 0 MINUTES WESTERN HEMISPHERE

:SETCLOCK; CANCEL

CORRECTION OF -3550 SECONDS HAS BEEN CANCELLED

:SHOWCLOCK

SYSTEM TIME: FRI, JUL 24, 1987, 8:52:53 AM
CURRENT TIME CORRECTION: 0 SECONDS
TIME ZONE: 7 HOURS 0 MINUTES WESTERN HEMISPHERE
```

Note that in the example above the system clock was slower than normal for several minutes. Cancelling the correction did not undo that change; it merely prevented any further time change. Thus after this sequence of commands, the system clock is set to a slightly earlier time than if no SETCLOCK command had been issued.

Examples of the Time Zone Form:

Moving from Standard Time to Daylight Savings Time:

The following example illustrates changing the system time zone offset from 8 hours 00 minutes in the Western Hemisphere (Pacific Standard Time) to 7 hours 00 minutes in the Western Hemisphere (Pacific Daylight Savings Time). This command will cause local time to jump forward immediately one hour. Universal Time will be unchanged.

```
:SETCLOCK TIMEZONE=W7:00

SYSTEM TIME: SUN, APR 4, 1993, 7:12:00 AM
CURRENT TIME CORRECTION: 3600 SECONDS
TIME ZONE: 7 HOURS 0 MINUTES WESTERN HEMISPHERE
```

Moving from Daylight Savings Time to Standard Time:

The following example illustrates changing the system time zone offset from 7 hours 00 minutes in the Western Hemisphere (Pacific Daylight Savings Time) back to 8 hours 00 minutes in the Western Hemisphere (Pacific Standard Time). This command will cause local time to slow down until it loses one hour. Users of Universal Time will see an immediate one-hour jump forward, followed by a slowdown until system Univeral Time again matches real Universal Time.

```
:SETCLOCK TIMEZONE= W8:00

SYSTEM TIME: SUN, OCT 31, 1993, 06:23:14 AM
CURRENT TIME CORRECTION: -3600 SECONDS
TIME ZONE: 8 HOURS 0 MINUTES WESTERN HEMISPHERE
```

Related Information

Commands	SHOWCLOCK, SHOWTIME
Manuals	*Performing System Management Tasks*

SETCOUNTER

Sets the next value of a specified resource counter, and optionally enables automatic rollback when a specified limit is reached. Duplicate values are avoided.

Syntax

```
SETCOUNTER[ COUNTER=] [ INSP | OUTSP | JOBNUM | SESSNUM ]
[ ;BASE = num] [ ;MAX = num ]
[ ;SHOW]
```

Parameters

INSP Specifies the input spoolid counter.

OUTSP Specifies the output spoolid counter.

JOBNUM Specifies the job number counter.

SESSNUM Specifies the session number counter.

The target counter (INSP, OUTSP, etc.) is only optional if the SHOW option is used by itself to display BASE and MAX values for all counters without changing any of them. For any other form of the command, the target counter is a required parameter.

num A positive integer. For MAX, *num* may also equal zero. A non-zero *num* for MAX must be less than or equal to the maximum possible value for that counter. Those values are:

INSP	9999999
OUTSP	9999999
JOBNUM	16383
SESSNUM	16383

For BASE, *num* must be less than MAX, except when MAX is equal to zero.

Operation Notes

The SETCOUNTER command allows you to specify limits other than 1 and the maximum possible value of one of four counters (but within that range) You may set limits for one counter with each use of the command and, therefore, you must invoke the command four times to change the limits of all four coutners.

You may also use SETCOUNTER to display the current values of the counters. Only one invocation of the command is necessary to see all current values.

To set a maximum operating value for the specified counter and enable its operation, enter a positive value for the MAX keyword. Specify MAX=0 to disable the operation, that is, the counter's limit is then its maximum possible value. Omitting MAX leaves its previous value in force. Once MAX is reached, the next value tried is the BASE value. If you specify a non-zero value for MAX, it must be greater than the current BASE for the corresponding counter, but less than the maximum possible value

The BASE keyword causes the specified counter to be immediately yanked to the specified value. If you supply a value, it must be less than the supplied or current value of MAX (other than 0), and in any case, less than the maximum possible value. If you do not specify BASE, it is not changed, nor is current sequencing affected.

For each counter, duplicate values are avoided. For example, if #O10 is in use when due to be assigned as the next output spoolid, it is skipped and #O11 is tried. This process continues until an available value is found.

The defaults, established when the system is booted, are MAX=0 and BASE=1. This is for backward compatibility; if these settings are not changed, the system will operate as it does today. These boot time settings can be modified by including one or more instances of this command in SYSSTART.PUB.SYS.

The SHOW option can be used alone to display the current values of BASE and MAX for a specified counter or for all four counters. If used in addition to either BASE or MAX, the value(s) displayed are the new setting(s).

This command may be issued from a session, job, program, or in Break. Any display specified by the SHOW option is breakable, but command operation is not. Any user may execute this command with only the SHOW option to display current values of BASE, Next, and MAX for the specified counter (or all counters if none is specified). When changing either value, this command may be executed only:

- at a console session,
- by a user with SM capability, -OR-
- by any user who has been allowed the use of the SETCOUNTER command with the ALLOW command.

Examples

To display the current BASE, Next, MAX, and maximum possible values for all four counters, enter:

```
:SETCOUNTER ; SHOW
                                             Absolute
  COUNTER          BASE     Next      MAX    maximum

Input spoolid        1      172     16383    9999999
Output spoolid       1     1872     32767    9999999
Job number           1      172         0     16383
Session number       1     2753         0      16383
```

To limit input spoolids to the same range as their corresponding jobs, enter:

```
:SETCOUNTER INSP; MAX=16383
```

Related Information

Commands None

Manuals None

SETDUMP

Arms the system debug facility for a process abort. (Native Mode)

Syntax

SETDUMP[DB [,ST [,QS]]] [;ASCII] [;DEBUG="*commands*"]

Parameters

DB	This parameter is ignored.
ST	This parameter is ignored.
QS	This parameter is ignored.
ASCII	This parameter is ignored.
"*commands*"	A string of system debug commands surrounded by quotation marks. Refer to the DEBUG command in this chapter.

Operation Notes

This command enables the stackdump facility for any process created later under the current session or job. If the call is armed (enabled), and the process aborts, SETDUMP executes the system debug commands given in the "*commands*" parameter.

If no *commands* are specified, a default command string is used to produce a stacktrace and register dump.

If the process is interactive, it subsequently enters the system debugger to wait for further commands. If it is not interactive, the process simply terminates instead of entering the debugger.

Any combination of the four strings (DB, ST, QS, or ASCII) is parsed without error in MPE/iX, but they have no effect on the functional behavior of the commands. The "*commands*" string, preceded by the DEBUG keyword, is interpreted as a series of system debug commands and is sent to the system debugger that way.

The "*commands*" parameter may contain a maximum of 255 characters.

NOTE The DB, ST, QS, and ASCII parameters are retained for compatibility reasons. These parameters are ignored.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

To arm the stackdump/debug facility, enter:

```
SETDUMP
```

Related Information

Commands	`DEBUG`, `RESETDUMP`
Manuals	*System Debug Reference Manual*

SETJCW

Creates or assigns a value to a job control word (JCW) variable.

Syntax

SETJCW ***jcwname delimiter value***[{ + - } *value*]

Parameters

jcwname The name of a new or existing user-defined 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 delimits the name and value.

value One of the following:

- An octal number between %0 and %177777, inclusive.
- A decimal number between 0 and 65,535, inclusive.
- An MPE/iX-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 for 0 + 3).
- 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.

Operation Notes

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.

Job control words in MPE/iX are classed as system variables of type JCW. You may delete user-created variables. You may modify the two system-defined variables CIERROR and JCW. Refer to appendix A, "Predefined Variables in MPE/iX," for a list of system-defined variables.

The SETVAR command creates and assigns variables too, but variables created or assigned with SETVAR are not of type JCW and cannot function as true job control words.

If you create or assign a value to a variable using the SETJCW command and later reassign its value using the SETVAR command, the reassignment succeeds. If the new value is out of range for a JCW, the variable type is changed to that of an ordinary user-defined variable:

```
 SETJCW PROGCNTR 0
....
 SETVAR PROGCNTR 65536
JCW VARIABLE RECLASSIFIED AS A STANDARD VARIABLE
(CIWARN 8126)
```

PROGCNTR is now a user-defined variable and does not function as a job control word.

JCWs can be tested against specific values. The user can use IF and WHILE conditional statements that act according to the results of these tests. The user-defined JCWs can also be set to user-selected values by a process so that they reflect the completion of steps within that process. System-defined JCWs can be used to determine whether certain events have occurred within MPE/iX.

The values in the system-reserved JCWs can be inspected by the user, but not altered.

To display the contents of a JCW use the SHOWJCW or the SHOWVAR 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/iX, since all 16 bits are used for numeric information, rather than using the most significant bit as a sign bit.

MPE/iX 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 WARN-4 (16,380). A mnemonic may also be added to another mnemonic, as in WARNFATAL.

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 is generated, and the value of the JCW remains 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 OK16385 is 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 the MPE/iX variable 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 (for example, WARN999999, or SYSTEM200000). If the computed value exceeds 65,535, MPE/iX does not recognize the term as a valid mnemonic, and treats 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 results 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 cause the following error message to be displayed:

```
VALUE NOT IN RANGELEGAL RANGE IS 0 TO 65535 (CIERR 1712)
```

System-Defined JCWs

JCW and CIERROR are MPE/iX system-defined JCWs created for each job and session. The JCW named JCW is always initialized to zero at the beginning of the job or session and remains 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:

```
 LISTF
 ^
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, WHILE, SETJCW, SETVAR, or CALC command. The names of the system-reserved JCWs are reserved.

The following describes 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/iX IF/THEN (ELSE/ELSEIF), ENDIF, and WHILE/ENDWHILE 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 CHEKPROG sets the value of 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 use the SHOWVAR command to display the value of any specified variable or any group of variables, including JCW type variables. You can display the contents of a system-defined JCW with the SHOWJCW command only if you specify the *jcwname*.

In the following example the CONTINUE command prevents an abort in case of errors; the RUN CHEKPROG edits, verifies, and counts valid transactions; the IF command specifies that if no fatal errors occur, schedule shipments; the RUN command schedules the shipments; the ELSE command produces the error report and resets the JCW to 0; and the RUN command produces a final report:

```
!SETJCW CHEKPROGSTAT=OK
!CONTINUE
!RUN CHEKPROG
! IF CHEKPROGSTAT<FATAL THEN
!  RUN SHIPPROG
! ELSE
!  SHOWJCW CHEKPROGSTAT
!  RUN ERRORRPT
!ENDIF
!RUN FINALRPT
```

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Examples

To set the job control word CURR1 to 100, and use a comma (,) as the delimiter instead of an =, enter:

```
SETJCW CURR1,100
```

To set CURR1 to the value of the mnemonic WARN, and use a slash (/) as the delimiter instead of an =, enter:

```
SETJCW CURR1/WARN
```

To use an arithmetic operation to set one JCW value relative to another, enter:

```
SETJCW NEWJCW=LASTJCW + 56
```

To schedule a full backup job on Saturdays and a partial backup job on the other days of the week, you could create a user command:

```
SETJCW FRIDAY=6
IF HPDAY = FRIDAY THEN
    SCHEDJOB FULLBKUP;IN=1
ELSE
    SCHEDJOB PARTBKUP;IN=1
ENDIF
```

Related Information

Commands	DELETEVAR, SETVAR, SHOWJCW, SHOWVAR
Manuals	Appendix A, "Predefined Variables in MPE/iX"

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, program, or in BREAK. Pressing **Break** has no effect on this command.

Operation Notes

Allows a 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.

Examples

To block messages, enter:

```
SETMSG OFF
```

To receive messages, enter:

```
SETMSG ON
```

Related Information

Commands	SET, TELL
Manuals	None

SETVAR

Assigns values to MPE/iX variables. (Native Mode)

Syntax

SETVAR ***varname***{ <space> , ; } *expression*

Parameters

varname The variable that is to be set to a value.

expression The expression that is evaluated and assigned to *varname*.

Operation Notes

This command assigns values to MPE/iX variables. Variable names may be any combination of letters and numbers plus the underbar character, up to a total of 255 characters. Variables must start with a letter or the underbar character.

The *expression* parameter may be an MPE/iX expression, a Boolean, integer, or string value, or the name of another variable. If *expression* consists of elements and operators MPE/iX accepts (`'abc' + 'cd'` or `2*5+1`), `SETVAR` will evaluate it. The operators defined in Table 7-1 may be used in *expression*.

Table 7-1 Logical Operators - The SETVAR Command

Logical operators:	AND, OR, XOR, NOT
Boolean functions and values:	BOUND, TRUE, FALSE, ALPHA, ALPHANUM, NUMERIC, ODD
Comparison operators:	=, <>, <, >, <=, >=
Bit manipulation operators:	LSL, LSR, CSR, CSL, BAND, BOR, BXOR, BNOT
Arithmetic operators:	MOD, ABS, * , / , + , -, ^ (exponentiation)
Functions returning strings:	CHR, DWNS, UPS, HEX, OCTAL, INPUT, LFT, RHT, RPT, LTRIM, RTRIM, STR
Functions returning integers:	ABS, LEN, MAX, MIN, ORD, POS, TYPEOF
Other functions:	FINFO, SETVAR

The allowed operands are any variable, integer constant (hexadecimal ($), octal (%), or decimal) quoted string constant, the Boolean constants TRUE and FALSE, or the JCW mnemonics (`SYSTEM`, `FATAL`, for example, as defined in the `SETJCW` command).

Note that all variables are global, so the CI variable name should not be the same as the JCW name that is being used or the operation of the code that uses that JCW will be affected.

Compound logical expressions can be formed using the AND, NOT, XOR, and OR logical operators, and nested within parentheses.

The Boolean value of the keyword TRUE or FALSE is overridden if there is a variable of the same name. For example, to store the string value 'ABC' in X, enter:

```
SETVAR TRUE 'ABC'
SETVAR X TRUE
```

The SETVAR command may be used to set the command interpreter's search path (HPPATH), the command interpreter's prompt (HPPROMPT), and all other variables. You use SHOWVAR to see all the variables that were created by the user. Issuing SHOWVAR @ causes the display of every predefined and user-defined variable.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** terminates an INPUT ( ) function.

Example

To change the command interpreter prompt to your *username.accountname*, enter:

```
SETVAR HPPROMPT "!HPUSER.!HPACCOUNT:" or
```

```
SETVAR HPPROMPT HPUSER+"."+HPACCOUNT+":"
```

The result is the same regardless of which form of the command you use.

Related Information

Commands	DELETEVAR, INPUT, SETJCW, SHOWJCW, SHOWVAR
Manuals	Appendix A, "Predefined Variables in MPE/iX"
	Appendix B, "Expression Evaluator Functions"

SHOWALLOCATE

Displays status information about the ALLOCATE command.

Syntax

SHOWALLOCATE[STATUS[,*listfile*]
ALLOCATE [,[*fileset*] [,*listfile*]]
ALL[,[*fileset*] [,*listfile*]]]

Parameters

STATUS	Request to display a summary of status information includes:
	(1)Number of programs allocated;
	(2)Size and percentage of utilization of the following system tables:
	Code segment table, code segment extension block table, and loader segment table.
ALLOCATE	Request to display programs for ALLOCATE specified by fileset, and the number of users sharing each program.
ALL	Request to display all information provided by parameters: STATUS and ALLOCATE and the default.
fileset	Specifies the set of files to be searched for. Default is @.@.@. This parameter is of the form: **filesdesignator[.groupdesignator[.acctdesignator]]** *fileset* can be entered in any of the following formats and may use wild card characters, in any order, as replacements.
file.group.account	SHOWALLOCATE file named in specified group and account.
file.group	SHOWALLOCATE specified file named in any group and any account.
file	SHOWALLOCATE specified file named in any group and any account.
@.group.account	SHOWALLOCATE all files in specified group and account.
@.@.account	SHOWALLOCATE all files in all groups in specified account.
@.@.@	SHOWALLOCATE all files in system and default.
@	SHOWALLOCATE all files in all groups in all accounts.
@.group	SHOWALLOCATE all files in specified group in any account.
file.@.account	SHOWALLOCATE specified file in any group of specified account.

NOTE The characters @, #, and *?* can be used as wild card characters in the *fileset* parameter. These wild card characters have the following meanings: @ specifies zero or more alphanumeric characters.

specifies one numeric character.

? specifies one alphanumeric character.

The characters can be used as follows:

n@ All files starting with the character *n*.

@n All files ending with the character *n*.

n@x All files starting with the character *n* and ending with the character *x*.

n##..# All files starting with the character *n* followed by up to seven digits (useful for listing all EDIT/3000 temporary files).

?n@ All files whose second character is *n*.

n? All two-character files starting with the character *n*.

?n All two-character files ending with the character *n*.

listfile Name of an output file to which all output is written. When specified, a new ASCII file with variable length records closed in permanent domain, user-supplied carriage control (CCTL), OUT access mode, and EXC (exclusive access) option. This parameter may also be a back-referenced file. Default is `$STDLIST`.

Operation Notes

This command generates the status information of the specified system tables and lists files which are allocated.

Use

This command requires system manager (SM) capability to execute for other groups or accounts.

Examples

To display status information for all allocated files in the system.

```
SHOWALLOCATE ALLOCATE

    ALLOCATED PROGRAMS         SHARE COUNT

     EDITOR.PUB.SYS . . . . . . . .    0
     FCOPY.PUB.SYS  . . . . . . . .    2
     LISTDIR5.PUB.SYS . . . . . . .    1

     NUMBER OF PROGRAMS FOUND = 3
```

To display status information for all allocated files starting with a character "S" in the account named SYS.

```
SHOWALLOCATE ALLOCATE,S@.@.SYS

    ALLOCATED PROGRAMS         SHARE COUNT
```

```
      SPOOK5.PUB.SYS . . . . . . . .    1
      SLPATCH.PUB.SYS . . . . . . . .    0

      NUMBER OF PROGRAMS FOUND = 2
```

To display summary status information regarding allocation.

```
 SHOWALLOCATE STATUS

    ALLOCATION STATUS

      NUMBER OF PROGRAMS ALLOCATED = 3

      ALLOCATION RELATED TABLES      SIZE   %USED

      CODE SEGMENT TABLE              191    52
      CSTX BLOCK TABLE                144    13
      LOADER SEGMENT TABLE           32000    3
```

Related Information

Commands   ALLOCATE

Manuals   *Performing System Operation Tasks*

SHOWALLOW

Displays which operator commands have been allowed.

Syntax

SHOWALLOW[{ *@.@* *user.@* *@.acct* *user.acct* }]

Parameters

@ All users, if used in place of *user*, or all accounts, if substituted for *acct*.

Default is that the commands allowed for the logged-on user are displayed.

user Defines a particular user.

acct Defines a particular account.

user.account Defines a particular user in a particular account.

Operation Notes

This command displays the operator commands that 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.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command. Account manager (AM) or system manager (SM) capability is required to execute this command for other groups or accounts.

Example

To list the operator commands allowed to the user `USER.SYS`, enter:

```
 SHOWALLOW USER.SYS
#S86 USER.SYS
  USER HAS THE FOLLOWING COMMANDS ALLOWED:
ABORTIO     ACCEPT     DOWN     GIVE

THERE ARE NO GLOBAL ALLOWS DEFINED.
```

Related Information

Commands `ALLOW`, `DISALLOW`

Manuals *Performing System Operation Tasks* (32650-90137)

SHOWCATALOG

Displays information about user-defined commands (UDCs). (Native Mode)

Syntax

SHOWCATALOG[*listfile*] [;USER=*username*[*.acctname*]]

Parameters

listfile — An arbitrary file name that identifies the output from SHOWCATALOG that is sent to the line printer. Specifying *listfile* sends the listing to device class LP (line printer). You may use a file equation to direct the listing of the catalog to a disk or tape file. If you omit this parameter, the listing is sent to the $STDLIST device.

USER — Permits the user to list other users' cataloged files. Account manager capability (AM) is required to show cataloged files for users within your logon account. System manager capability (SM) is required to show users' cataloged files in other accounts.

username.
acctname — Specifies the user and/or account name whose file names are to be displayed. The @ wildcard character may be used to specify all the members of a set:

```
USER=username
USER=username.acctname
USER=@.acctname
USER=@.@
```

Operation Notes

This command lists user-defined command files, their commands and the level at which they were cataloged (user, account, or `system). This may not be the executing UDC catalog directory, as with the USER option. The user may specify a *listfile* to send the listing to the line printer. You may use a file equation to direct the listing of the catalog to another disk or tape file. Default is that the listing is sent to the $STDLIST device.

If SETCATALOG is performed with the USER option after the user logs on, the user's executing UDC directory is not affected. Only the UDC catalog set is affected. The next time the user logs on, the UDC directory is built from this set. Thus the SHOWCATALOG command with the USER option shows the UDC catalog set. The SHOWCATALOG command alone shows the currently executing UDC directory commands.

Use

This command is available from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command.

Examples

To display the account-level UDC files of all users in the GRIMSBY account, enter:

```
SHOWCATALOG ;USER=@.GRIMSBY
```

To display the system-level UDC files of all users in all accounts, enter:

```
SHOWCATALOG ;USER=@.@
```

To display all UDC command files for the current user and send the listing to the line printer (LP), enter:

```
SHOWCATALOG MYFILE
```

To display all UDC command files for the current user and send the listing to the disk file called MYFILE, enter:

```
FILE MYFILE;DEV=DISK
SHOWCATALOG *MYFILE
```

To send all system-level UDC files to the line printer under the name LISTALL, enter:

```
SHOWCATALOG LISTALL,@.@
```

To display a list of the cataloged files for the user SCOTT in your account, enter:

```
SHOWCATALOG,SCOTT
```

Related Information

Commands	SETCATALOG, HELP <udcname>
Manuals	*System Startup, Configuration, and Shutdown Reference Manual* (32650-90042)

SHOWCLOCK

Displays information about the system date and time.

SYNTAX

SHOWCLOCK

Parameters

None.

Operation Notes

Prints the current time, date, the time correction in effect, and the time zone. See the command SETCLOCK for information about time correction and time zone.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

```
:<user |SHOWCLOCK|

SYSTEM TIME: FRI, JUL 24, 1987, 8:47:35 AM
CURRENT TIME CORRECTION: -3428 seconds
TIME ZONE: 7 HOURS 0 MINUTES WESTERN HEMISPHERE
```

Related Information

Commands	SHOWTIME, SETCLOCK
Manuals	None

SHOWDEV

Reports the status of input/output devices.

Syntax

SHOWDEV[*ldev* *classname*] [;ACD]

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.

ACD Keyword requesting display of ACD (access control definition) for the device.

Operation Notes

The SHOWDEV command displays the status information for all input and output devices on the system. The display spacing is important and has been changed after the 4.7 release. The display appears in the following format:

```
 SHOWDEV

Total number of blanks between items after release 4.7

   5     9         9       9    3
LDEV   AVAIL     OWNERSHIP     VOLID     DEN  ASSOCIATION
  1   DISC     N/A
  2   DISC     N/A
  3   DISC     N/A
  4   DISC     N/A
  5   AVAIL
  6   SPOOLED    SPOOLER OUT
  7   AVAIL
  8   AVAIL
  9   AVAIL
 10  A AVAIL
 11   DISC     N/A
 12   DISC     N/A
 13   DISC     N/A
 14   DISC     N/A
 15   DISC     N/A
 16   DISC     N/A
 17   AVAIL
 18   AVAIL
 19   SPOOLED
 20  A UNAVAIL    #S914: 8 FILES
 21  A AVAIL
```

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 disks as follows:

- AVAIL The device is available as a real, nonshareable device.
- AVAIL W The device is a tape with write enable on the media.
- SPOOLED The device is available for 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 disk and is always available.
- DISC (RPS) The device is a CS-80 disk 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.
- #J*nnn* Controlled by the indicated job.
- #S*nnn* Controlled by the indicated session.
- *nn* FILES Indicates number of files currently in use on a disk.
- DOWN Device is offline, requested by system operator with the DOWN command.
- DP Device is being taken offline (DOWN command operation pending).

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 tape volume that has a label.
- NOLABEL The named magnetic tape volume that has no label. Default.

DEN Density of the tape, which 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 that have been established by the user with the ASSOCIATE command.

ACD Access Control Definition. May include any of the following information per username.acctname:

R READ access.

W WRITE access.

L LOCK access.

A APPEND access.

X EXECUTE access.

NONE NO access.

RACD Copy or read the ACD.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this 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
```

Related Information

Commands DISCRPS, ABORTIO

Manuals *Performing System Operation Tasks*

SHOWIN

Reports the status of input device files.

Syntax

SHOWIN[#I*nnn* STATUS] [;SP] [;*item*[;*item* [;...]]]

Parameters

#I*nnn* — Identifies the particular input device file for which information is to be displayed. Default is that MPE/iX displays information for all input device files used by the logon job or session.

STATUS — Summarizes the status information for all current input device files. Default is that MPE/iX displays information for all input device files used by the logon job or session. The information appears in following format:

```
8 FILES DISPLAYED
  0 ACTIVE
  0 READY;INCLUDING 0 SPOOFLES, 0 DEFERRED
  8 OPENED; INCLUDING 0 SPOOFLES
  0 SPOOFLES; 0 SECTORS
0 LOCKED; INCLUDING 0 SPOOFLES
```

SP — Displays status information for the currently spooled input device files associated with the logon job or session. Default is a display of status information for all input device files.

item — Displays the status of current input device files as identified. Default is that MPE/iX displays status information for all input device files used by this job.

Syntax for Item

[DEV=*ldev*] [JOB={ @J @S @ [#] J*nnn* [#] S*nnn* }] [{ ACTIVE | OPENED | READY }]

Parameters for Item

ldev — Displays the status of input device files identified by logical device number *ldev*.

JOB= — Displays the status of input device files. JOB= may be one of the following options:

- @J — Displays the status of input device files for all jobs.
- @S — Displays the status of input device files for all sessions.
- @ — Displays the status of device files for all jobs and sessions. (Default.)

[#]J*nnn* Displays the status of all input device files for a specified job.

[#]S*nnn* Displays the status of all input device files for a specified session.

ACTIVE,
OPENED, or
READY Displays the status of all input files in a specified state. ACTIVE displays the status of active device files. OPENED displays the status of opened device files. READY displays the status of ready device files.

Operation Notes

This command displays the status information about one or more currently defined input device files. 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 "Parameters"), the format of the information is as follows:

```
 DEV/CL DFID  JOBNUM FNAME  STATE   FRM SPACE RANK PRI #C
 10     #I10  #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 #I*nnn*. The number displayed in the DFID is identical to the LDEV number.

JOBNUM Job or session number (*jsnum*) of the job or session using the device file, 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 device file.

STATE One of the following:

ACTIVE Input being read from a spooled device to a disk.

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 appears only if a forms alignment message applies to the device file. Does not apply to input files.

SPACE Approximate disk 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.

The letter D following RANK indicates a deferred file for spooled device files 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 device file, requested by the user or adjusted by the system operator. Specified for spooled output device files only.

#C The number of copies needed, specified for spooled output device files only.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command.

Examples

The following is an example of how to determine the status of an individual input device file:

```
  SHOWIN #I80

 DEV/CL DFID JOBNUM FNAME    STATE   FRM SPACE RANK PRI #C
 43     #I43 #S37  $STDIN   OPENED              8
```

If you do not know the device file identification number (DFID) of the device file 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     #I43 #S37  $STDIN  OPENED
```

You may also request displays of device file information using various combinations of qualifications (devices, jobs/sessions, and states). For example, to display information about all OPENED input device files 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      #I7  #S38  MASTER  OPENED
 26     #I26 #S18  $STDIN  OPENED
 32     #I32 #S41  $STDIN  OPENED
 34     #I34 #S26  $STDIN  OPENED
 42     #I42 #S28  $STDIN  OPENED
 43     #I43 #S37  $STDIN  OPENED
 50     #I50 #S40  $STDIN  OPENED
 51     #I51 #S17  $STDIN  OPENED
 8 FILES (DISPLAYED):
   0 SPOOFLES: 0 SECTORS
```

Related Information

Commands SHOWOUT, LISTSPF

Manuals *Performing System Operation Tasks* (32650-90137)

SHOWJCW

Displays the current state of one or more job control word (JCW) variables.

Syntax

SHOWJCW[*jcwname*]

Parameters

jcwname The name of a valid job control word (JCW) variable. Default is that all user-defined and system-defined JCWs are displayed.

Operation Notes

The SHOWJCW command is used to display the current state of one or more job control words (JCWs). Job control words in MPE/iX are classed as variables of type JCW. Specifying a particular JCW (user-defined, system-defined, or system-reserved) displays the value 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*. The SHOWVAR command can be used to show variable values as well.

You may retrieve the value assigned a JCW with the FINDJCW and HPCIGETVAR intrinsics.

You may test the value of a JCW with an IF or WHILE 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 USER.TECHPUBS;COMPILE FAILED
!ELSE
!  TELL USER.TECHPUBS;COMPILE COMPLETED
!ENDIF
```

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command.

Examples

To show the current state of all user-defined 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
```

Related Information

Commands	DELETEVAR, SETJCW, SETVAR, SHOWVAR
Manuals	Appendix A, "Predefined Variables in MPE/iX"

SHOWJOB

Displays status information about jobs/sessions.

Syntax

SHOWJOB[[#] S*nnn* [#] J*nnn* STATUS SCHED *item*[;*item*[;...]]] [;**listfile*]
[;JOBQ]

Parameters

#S*nnn* The session number (assigned by MPE/iX) of the session for which the status information is to be displayed. The information appears in Type I format, described under "Operation Notes." Default is that the status information for all jobs/sessions is displayed.

#J*nnn* The job number (assigned by MPE/iX) of the job for which status information is to be displayed. The information is in Type I format, described under "Operation Notes." 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 jobfence and job/session limits. This information is in Type II format, described under "Operation Notes." 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 Notes."

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.

**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.

JOBQ Which will indicate the queue name to which the job belongs. A new field JOBQ is added into the showjob output format.

Syntax for Item

[JOB={ @J | @S | @ [@,] *username.acctname* [*jsname*,] *username.acctname* }]
[;{INTRO | EXEC | SUSP | WAIT[‘N | ,D]}]

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. Default.

[*jsname,*] *username. acctname* The *jsname* is an optional name given to the session or job by the user. The *username* parameter is the user name established by the account manager. This name may consist of one to eight alphanumeric characters beginning with an alphabetic character. The *acctname* 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 disk. 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 device files.

D Displays the status of deferred READY device files.

If information for only one device file is displayed, output is in Type I format; if information for more than one device file is displayed, output is in Type I followed by Type II format. (Format types are described under "Operation Notes.")

Operation Notes

This command enables you to determine the number of jobs and sessions in each processing state, the current jobfence and job/session limits, and allows you to keep track of individual spooled and streamed jobs that are entered in the system. The command jobq will indicate the queue name to which the job belongs.The output appears in the following 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 is 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,MR.SAM

3 JOBS (DISPLAYED)
JOBFENCE=7; JLIMIT=2; SLIMIT=20
```

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command.

Examples

To determine the number of jobs and sessions in each processing state, the current jobfence 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:
  0 INTRO
  0 WAIT; INCL 0 DEFERRED
  2 EXEC; INCL 2 SESSIONS
  0 SUSP
JOBFENCE= 2; JLIMIT= 1; SLIMIT= 16
```

```
:SHOWJOB;jobq
```

```
JOBNUM  STATE IPRI JLIST  JOBQ      INTRODUCED  JOB NAME

  #J3   EXEC    LP      HPSYSJQ   WED 11:46A FTPMON,FTP.SYS

  #J7   EXEC    LP      SYSMGRQ   WED 5:47P EMG,MGR.SYSMGR

  #S81  EXEC    34                 THU 12:17P  MGR.GOPI
```

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.03.26 FILE COPIER (C) HEWLETT-PACKARD CO. 1984

MON, MAY 7, 1987, 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 USER.PUBS
#S47  EXEC    10S LP   MON 7:26A SUPPORT.DOC
#S48  EXEC    102 102  MON 7:28A USER.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 changes to EXEC. The number shown in the EXEC state is the sum of the jobs and sessions in both EXEC and EXEC*.

Related Information

Commands	ABORTJOB, BREAKJOB
Manuals	*Performing System Operation Tasks*

SHOWLOG

Displays the number of the system's current log file and the percentage of disk space used. (Native Mode)

Syntax

SHOWLOG

Parameters

None.

Operation Notes

The log file number, *xxxx*, and percentage of file space used, *yy*, is displayed in the format:

```
  SYSTEM LOG FILE #xxxx IS yy% FULL
```

If the logging system is disabled, MPE/iX displays the message:

```
  NO LOGGING
```

If logging is enabled but currently suspended due to an error, both messages are displayed.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. System supervisor (OP) capability is required to use this command.

Example

To display the current log file status, enter:

```
  SHOWLOG
 SYSTEM LOG FILE #7 IS 20% FULL
```

Related Information

Commands	ALTLOG, CHANGELOG, GETLOG, LISTLOG, LOG, OPENLOG, RELLOG, RESUMELOG, SHOWLOGSTATUS, SWITCHLOG
Manuals	*Performing System Operation Tasks*
	System Startup, Configuration, and Shutdown Reference Manual

SHOWLOGSTATUS

Displays status information about currently opened user logging files assigned to a logging identifier.

Syntax

SHOWLOGSTATUS[*logid*]

Parameters

logid Displays status of the user logging file associated with the logging identifier, *logid*, created by the GETLOG command. Default is that the status of all logging identifiers is displayed.

Operation Notes

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. By default this command gives the following information about all currently running logging processes. To display the status of the logging identifier LEN, enter:

```
  SHOWLOGSTATUS LEN

  LOGID  CHANGE AUTO USERS STATE  CUR-REC  MAX-REC % USED CUR-F

  LEN    NO      NO   4  INACTIVE  100       1000  10%   1
```

The information provided in this format is defined as follows:

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 through 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 START RECOVERY is issued.
CUR-REC	The number of records in the log file.
MAX-REC	The maximum number of records permitted.
% USED	The percentage of the maximum used.

`CUR-F` The current file number in the set.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

Refer to "Operation Notes."

Related Information

Commands	`ALTLOG`, `CHANGELOG`, `GETLOG`, `LISTLOG`, `LOG`, `OPENLOG`, `RELLOG`, `RESUMELOG`, `SHOWLOG`, `SWITCHLOG`
Manuals	*Performing System Operation Tasks* *System Startup, Configuration, and Shutdown Reference Manual*

SHOWME

Reports the status of a job or session. (Native Mode)

Syntax

SHOWME

Parameters

None.

Operation Notes

To display the status of the current job/session enter:

```
SHOWME
USER: #S485,MGR.DSUSER,PUB   (NOT IN BREAK)
RELEASE: V.UU.FF MPE XL HP31900 A.11.70 USER VERSION: V.UU.FF
CURRENT: MON, MAY 7, 1987, 11:09 AM
LOGON:  MON, MAY 7, 1987, 11:08 AM
CPU SECONDS: 3     CONNECT MINUTES: 1
$STDIN LDEV: 88     $STDLIST LDEV: 88
```

The system welcome message, if one exists, appears immediately following the `SHOWME` display. The information provided in the format above is defined as follows:

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/iX command interpreter.

`RELEASE: V.UU.FF` The `RELEASE: V.UU.FF` number is determined by Hewlett-Packard at build time of the operating system and provides an identity for software releases (also known as the MIT). This number cannot be changed. (Prior to MPE/iX release A.11.70, this was referred to as `BASE`.

`USER VERSION` The `USER VERSION: V.UU.FF` can be given a value during a SYSGEN and allows you to identify any changes to your total software package such as patch level, third party software, or other specifics. Any ASCII character can be used. In prior releases, this number was printed out immediately after the MPE/iX product number HP31900.

`HP31900 A.11.70` The `PRODUCT V.UU.FF` immediately follows the product number HP31900. It is determined by Hewlett-Packard when a new version of the MPE/iX operating system is compiled. This `V.UU.FF` number cannot be changed and is used when entering a service request (SR) against the MPE/iX operating system product for that particular release.

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 the 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.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command.

Example

Refer to "Operation Notes."

Related Information

Commands HELLO, JOB, SHOWJOB

Manuals None

SHOWOUT

Displays the status of output device files.

Syntax

SHOWOUT[{ #O*nnn* STATUS SP *item*[;*item*[;...]] }]

Parameters

#O*nnn* Identifies a particular output device file for which you want information. The information is displayed in Type I format, which is described in the "Operation Notes" section of this command. The default is to display status information for all output device files used by the logon job or session.

The number of the device file identifier is identical to the LDEV number of the device. For example, if the LDEV number is 20, the device file identifier appears in the DFID column as #20.

STATUS Summarizes the status information for all current output device files. The information is displayed in Type II format, described in the "Operation Notes" section. The default is to display status information for all output device files used by the logon job or session.

SP Displays the status information for currently spooled output device files associated with the logon job or session. The information is displayed in a combination of two formats, Type I format followed by Type II format, which is described in the "Operation Notes" section. The default is to display status information for all output device files used by the logon job or session.

item Displays the status of all current output device files as identified. If information for only one device file is displayed, the output appears in Type I format. If information for more than one device file is displayed, the output appears in Type I format followed by Type II format. The syntax for *item* follows:

Syntax for Item

[DEV={ *ldev classname* }]

[JOB={ @J | @S | @ | [@,]*username.acctname* |[*jsname*,] *username.acctname*}

[;[INTRO | EXEC | SUSP | WAIT [,N | ,D]}]

Parameters for Item

ldev or *classname* Displays the status of output device files. 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 device files residing on all devices in a class name.

`JOB=` Displays the status of output device files using one of the following options:

- `@J` Displays the status of output device files for all jobs.
- `@S` Displays the status of output device files for all sessions.
- `@` Displays the output device files for all jobs and sessions.
- `[#]J`*nnn* Displays all output device files for specified job.
- `[#]S`*nnn* Displays the status of all output device files for a specified session.

`ACTIVE`, `OPENED`, `READY`, or `LOCKED`

Displays status of all output files in the specified state. An ACTIVE file is one that is currently being produced on your printer or plotter. 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 disk; during that time, however, the file is not ready to be printed. READY files are completely spooled and ready to be output. A LOCKED file is READY but cannot be accessed until the system relinquishes its exclusive use of the file.

READY files may include one of the following:

- `N` Displays the status of nondeferred READY device files.
- `D` Displays the status of deferred READY device files.

Operation Notes

This command displays the status information for one or more currently defined output device files. 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/iX, 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 by output priority, device class, and rank. If the priorities are equal, the spooler alternates between queues.

Information about all spoolfiles on the system is available only from the console. Information about spoolfiles created in a specific job or session is available during that job or session only.

To list information about an individual output device file, you may specify its device file identifier (`DFID`) in the `SHOWOUT` command:

```
SHOWOUT #O26

DEV/CL DFID  JOBNUM FNAME  STATE FRM SPACE RANK PRI #C
EPOC  #O26  #J242 $STDLIST READY    36  D  1  1

OUTFENCE = 6
```

The information provided in this format is defined as follows:

COLUMN	**MEANING**
`DEV/CL`	Logical device number or device class name of the device.

DFID Device file identification, which begins with the letter O (not zero) followed by a number. The numeric portion of the DFID is identical to the LDEV number of the device.

JOBNUM The job/session number (*jsnum*) of job or session using the device file.

FNAME File name assigned to device file.

STATE The status, indicated by one of the following subparameters:

- ACTIVE The spooled device file on disk is actually being written to a printer or plotter.
- OPENED The device file on disk is being accessed by a program. If the device file is spooled, a program is currently writing to the disk.
- READY The spooled device file on disk is ready for output.
- LOCKED READY, but the system has exclusive access to the file.

FRM The forms message indicator (the letter F) appears only if a forms alignment message applies to this device file.

SPACE The approximate disk space currently being used, expressed in sectors. This applies only to spooled output device files.

RANK The ranking of the file and 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 intrinsic determines the file's rank.

The letter D following RANK indicates a deferred file. This applies only to spooled device files. 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 a user or as adjusted by the system operator for spooled device files only. A priority of 1 is lowest, and 13 is highest.

#C Number of copies needed, for spooled device files 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
32   #O32 #S16  $STDLIST OPENED
OUTFENCE=6
```

Type II:

```
19 FILES
O 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
```

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command.

Examples

To display the total number of output device files currently existing, the number of those that are spooled, and their current status, enter:

```
SHOWOUT STATUS
11 FILES:
  1 ACTIVE
  1 READY; INCLUDING 1 SPOOFLES, 0 DEFERRED
  9 OPENED; INCLUDING 1 SPOOFLES
  0 LOCKED; INCLUDING 0 SPOOFLES
  3 SPOOFLES: 7212 SECTORS
OUTFENCE= 2
:
```

You can also request information about a specific output device file, 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   #O43 #S37  $STDLIST OPENED

OUTFENCE= 2
:
```

Related Information

Commands `SHOWIN`, `LISTSPF`

Manuals *Performing System Operation Tasks*

SHOWPROC

Displays information about the specified process(es). (Native Mode)

Syntax

```
SHOWPROC[ [ PIN=]{pinspec | (pinspec [ ,pinspec ] ...)}]
[[;JOB=]{jobspec | (jobspec [ ,jobspec] ...)}]
[[;FORMAT=]{SUMMARY | DETAIL}]
[{;TREE | ;NOTREEE}]
[{;USER | ANYUSER}]
[{;SYSTEM}]
[{;TRUNC | ;NOTRUNC}]
```

Parameters

pinspec The process that you want to see.

The *pinspec*, expressed [#p]pin, is a Process Identification Number (PIN). Specifying *pinspec* is optional and has no default; see *jobspec*.

An ordinary user may show processes matching their own user and account names (those which "belong to" the user) by specifying 0 as the *pinspec*. A user with SM or OP capabilities may show any process on the system. A user with SM capability (the system manager) may see system processes by specifying the `SYSTEM` option.

`NOTREE` is the default for all *pinspec* target processes, and can be overridden with the `TREE` option.

The `USER` and `ANYUSER` options do not apply to *pinspec*.

jobspec The name of the job or session whose processes you want to display. A *jobspec* can be any of the following: jobnumber, username, @S, @J, or @. A *jobspec* is optional and defaults to the user's current job ID, for example, `#!HPJOBTYPE!HPJOBNUM`.

The jobnumber must be in the form *#Jnnn* or *#Snnn*. SM or OP capability is required to specify another user's job or session number. The username must be in the form *user*[.*account*]. SM or OP capability is required to specify another user's username. If there is more than one job or session under the same username, all are displayed.

You can use wildcards; they have the following meanings:

@S - all sessions

@J - all jobs

@ - all sessions and jobs

An ordinary user can only see their own processes, even when *jobspec* is wildcarded. For example, if the user name is JEFF.MFG and you enter the command as shown below, then only processes for jobs logged on as JEFF.MFG are displayed.

```
:SHOWPROC job=@J
```

On the other hand, if the user STEVE.UI (who has OP or SM capability) enters the command shown below, then all processes for all jobs on the system are displayed.

```
:SHOWPROC job=@J
```

If the user STEVE.UI only wants to see his own job processes, he must enter:

```
:SHOWPROC job=@J; user
```

The USER option, and its counterpart option, ANYUSER, are described below.

The SYSTEM option is ignored for all *jobspec* target processes.

TREE is the default for all *jobspec* target processes, and can be overridden with the NOTREE option.

SUMMARY This format displays a subset of a process' attributes. These include the subqueue name, process priority, CPU time, execution state, associated JOB or SESSION number, PIN (indented to show tree structure), program name, and INFO=string, if any (or command step if the process is CI.PUB.SYS). The INFO=string and command step information is only visible to the system manager and to processes that belong to the user. SUMMARY is the default format.

DETAIL This format displays a more comprehensive set of the attributes associated with a process.

TREE This option displays each process specified, as well as all of its descendents. TREE is the default for all *jobspec* target processes.

NOTREE This option displays only the process specified. No information appears for the process's descendants. NOTREE is the default for all *pinspec* target processes.

SYSTEM The SYSTEM option is required if the target process from *pinspec* is a system process. It displays system processes as well as descendant user processes. SM capability is required. SYSTEM is ignored for all *jobspec* processes.

USER The USER option filters output when *jobspec* is wildcarded by displaying only processes matching the user's name. USER is the default for users without OP and SM capability.

ANYUSER — This option defeats the filtering of the wildcarded *jobspec* and displays all matching processes. SM or OP capability is necessary to specify ANYUSER, and users with these capabilities get ANYUSER by default. OP or SM users may reduce the SHOWPROC output to just their own processes by using the USER option.

TRUNC — The TRUNC option truncates output records that would exceed the record width of $STDLIST for the user. A $ replaces the last character of the line to signify truncation. TRUNC is the default option.

NOTRUNC — This option displays output records in their full form. As a result, output from the command may wrap around the display.

Operation Notes

The SHOWPROC command displays information about processes except lockwords, which are never displayed. By default, the processes shown are the root CI and its descendents (TREE option). Any user may issue this command. Users with OP or SM capability may see information for processes belonging to other users. SM users may also see system processes via the SYSTEM option.

Any user may issue the SHOWPROC command and see information about all processes that belong to them. A process "belongs" to a user if one or more of the following conditions exists:

1. the process is within the user's logon job/session
2. the process' user and account names match the user's user and account names *and* the system's JOBSECURITY is set to LOW
3. the user has OP or SM capability.

If rule 1 or 2 applies or the user has SM capability then all information (except lockwords) is visible. Otherwise, only the Command Interperter (CI) command and/or program names are shown. That is, the parameters of a CI command and the INFO= string passed to a program are not visible.

When SHOWPROC is executed in a job, regardless of capabilities and process ownership, only the CI command name and program are displayed.

If you specify both the ;PIN= and ;JOB= parameters, information for the list of pins will precede the information for the list of jobs. Duplicate specifications are not detected.

SHOWPROC may be issued from a Session, Job, Program, or in BREAK. Pressing **Break** aborts the execution of this command.

The fields displayed are described below. The field's width, in characters, is shown within parentheses. A "v" indicates that the field has a variable size width.

CPUTIME (8): — CPUTIME is consumed in hh:mm:ss or m:ss.mls. A pair of asterisks (**) appears in the hours field when hours overflows. The three-character "mls" sub-field holds milliseconds.

JOBNUM (6): — The job or session number for the process.

LOGON (v): The job/session, user, and account name associated with this process.

PARENT (5): Process Identification Number for the process' parent (decimal). This field is unique to the DETAIL format. The DETAIL format displays PARENT so that process relationships can be determined. A zero indicates that the process does not have a parent (for example, PROGEN).

PIN (5): Process Identification Number for the process (decimal). The SUMMARY format indents the PIN column by two spaces for each child process so that you can clearly see a process' descendants. The DETAIL format precedes the pin with a percent sign (%) to indicate that the process is an artificial member of its workgroup, and does not indent the display.

PRI (5) The priority at which the process is currently executing. A lower numeric value indicates a higher priority. It also indicates whether the process is linear, runs with fixed priority (L), or is decayable (D). This field is unique to the DETAIL format.

PROGRAM (v): The file name of the program the process is executing.

QUEUE (v): The scheduling queue attribute associated with this process. The QUEUE field is unique to the DETAIL format.

QPRI (5): A combination of SUBQUEUE and PRIORITY which appears as Qnnn[*]. Q is a single character abbreviation of the process' scheduling queue attribute. The nnn is the process' priority, and * indicates that this process is a system process. The QPRI field is unique to the SUMMARY format.

STATE (5): The execution state of the process, which can be one of the following:

- BLKIO blocked for terminal write or control.
- WAIT generic process block, usually waiting for a message.
- BLKCB blocked for control block.
- BLKMM blocked for memory manager.
- READY ready to execute (or executing).

STEP (v): The command that the displayed CI process is currently executing. This field is not shown for non-CI processes.

WORKGROUP (v): The workgroup of which the process is a member. WORKGROUP appears as [%]name, where % indicates that the process is an artificial member of the workgroup, and name is the workgroup name. A process becomes an artificial member when it is explicitly placed into the workgroup via ALTPROC or AIFPROCPUT instead of naturally meeting the membership criteria of the workgroup.

On the next page is a sample output of the DETAIL format. In this example, pin 2 is a system mode process, running linearly at priority 142. Pin 99 is a user mode process running linearly at priority 160. Pin 121 is a user mode process that is an artificial member of the "Payroll_Online" workgroup

```
:SHOWPROC pin=(2,99,121,188);format=detail;system

 PIN PARENT  PRI  CPUTIME  STATE  JOBNUM (PROGRAM) STEP
-  -   -
 2 1     142 L  7:23.687 WAIT      (LOAD.PUB.SYS)

LOGON     :
PROGRAM    : LOAD.PUB.SYS
QUEUE     : BS
WORKGROUP   : BS_Default

***********************


 PIN PARENT  PRI  CPUTIME  STATE  JOBNUM  (PROGRAM) STEP
-  -   -
 99 68    160 L  0:05.020 BLKIO  S45  (QEDIT.PUB.SYS)

LOGON     : NMTEST,SLC.MYTEST
PROGRAM    : QEDIT.PUB.SYS
QUEUE     : BS
WORKGROUP   : Program_Development

***********************


 PIN PARENT  PRI  CPUTIME  STATE  JOBNUM (PROGRAM) STEP
-  -   -
 121 97    158 D  0:12.045 READY  J51   :tdp "text report"

LOGON     : JREPORT,GREG.MYTEST
PROGRAM    : TDP.PUB.SYS
QUEUE     : DS
WORKGROUP   : %Payroll_Online


************************

 PIN PARENT  PRI  CPUTIME  STATE  JOBNUM (PROGRAM) STEP
-  -   -
 188 101   100 D  0:04.200 WAIT   S56   (TDP.PUB.SYS) text test1

LOGON     : CMTEST,DOUG.MYTEST
PROGRAM    : TDP.PUB.SYS
QUEUE     : BS
WORKGROUP   : BS_Default
```

Below is a sample output of the default SUMMARY format. The information in the (PROGRAM) STEP column is visible only when the user issuing the command has SM capability, or when the process specified on the command line (in this case, #P54) belongs to the user.

```
:SHOWPROC #P54; tree; trunc

QPRI CPU    STATE JOBNUM PIN  (PROGRAM) STEP

C152 0:12.999 WAIT S12   54  :tdp "text myfile"
C152 0:02.000 WAIT S12    38  (TDP.PUB.SYS) text myfile
C152 0:01.030 READY S12     67  (FCOPY.PUB.SYS)from=foo.pub.sys;to=b$


:SHOWPROC #P54; tree; notrunc

QPRI CPU    STATE JOBNUM PIN  (PROGRAM) STEP

C152 0:12.999 WAIT S12   54  :tdp "text myfile"
```

```
C152 0:02.000 WAIT S12    38  (TDP.PUB.SYS) text myfile
C152 0:01.030 READY S12     67  (FCOPY.PUB.SYS)from=foo.pub.sys;to=ba
r;new
```

Example

To display a summary of information for all non-system processes in the current job/session, enter:

```
:SHOWPROC
```

To display a summary of information for PIN 42, enter:

```
:SHOWPROC #p42
```

To display a summary of information for PIN 42 and all of its descendants, enter:

```
:SHOWPROC #p42; tree
```

To display the detail information for PIN 42, enter:

```
:SHOWPROC #p42; format= detail
```

To display a summary of information for all processes (requires SM capability), enter:

```
:SHOWPROC 1 ;system ;tree
```

To display a summary of information for all non-system processes that are jobs (requires SM or OP capability), enter:

```
:SHOWPROC job=@j; anyuser
```

To display a summary of information for PINs 150, 247, and 211, enter:

```
:SHOWPROC (150,#p247,211)
```

To display a summary of information for all non-system processes logged on as MGR.PAYROLL (requires SM or OP capability), enter:

```
:SHOWPROC job=mgr.payroll
```

To display a summary of information for all non-system processes belonging to Job 2 or logged on as ME.AP (requires SM or OP capability), enter:

```
:SHOWPROC job=(#j2,me.ap)
```

To display the detail information for all non-system processes in the current job/session, enter:

```
:SHOWPROC detail
```

To display the detail information for all non-system processes on the system (requires SM or OP capability), enter:

```
:SHOWPROC job=@; format= detail
```

Related Information

Commands	TUNE, ALTPROC, SHOWQ, NEWWG, ALTWG, PURGEWG, SHOWWG
Manuals	*MPE/iX Intrinsics Reference Manual*

SHOWQ

Displays scheduling data for all processes and the scheduling characteristics of the CS, DS and ES scheduling subqueue(s). (Native Mode)

SYNTAX

SHOWQ[;ACTIVE] [;STATUS]

Parameters

ACTIVE — Displays only the processes currently running or those about to run. This is the right-hand portion of the display. The STATUS lines are printed last.

STATUS — Reduces the output from SHOWQ to the final status lines of display (base and limit priorities, quantum bounds).

Operation Notes

The process scheduling and subqueue information appears in two major columns: DORMANT and RUNNING. RUNNING processes are those 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. DORMANT processes are those waiting on longer-term events.

On occasion, a process appears in more than one column, indicating that it was changing state when you executed SHOWQ.

As the default, SHOWQ lists dormant and running processes and the scheduling characteristics of the CS, DS, and ES subqueues. However, the ACTIVE and STATUS options permit you to filter the SHOWQ output which, on large systems, may display hundreds of live processes.

Use the ACTIVE option to display running processes and the scheduling characteristics of the CS, DS, and ES scheduling subqueues. Use the STATUS option to display just the scheduling characteristics of the CS, DS, and ES subqueues. (Note that the ACTIVE output appears when both options are specified, since status information is a subset of the active information.)

Below is an example of the two-column output produced by the SHOWQ command. The symbols that may appear in such a listing are explained in the remainder of the discussion.

```
DORMANT       RUNNING
Q PIN JOBNUM   Q PIN JOBNUM

A 1        C M163 #S263
B 2        C U215 #S256
B 3
A 4
D U29 #J30
C M37 #S234
C M55 #S248
```

Each entry in the three columns displays the following information for a single process; the meaning is explained below.

{ A B C D E } [M U] pin [#Jnnn #Snnn]

A the queue attribute of the process is AS

B the queue attribute of the process is BS

C the queue attribute of the process is CS

D the queue attribute of the process is DS

E the queue attribute of the process is ES

M this is a job or session main process

U this is a user process

pin process identification number, a decimal

J nnn job number: a process executing in a batch job

S nnn session number: a 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 characteristics currently in effect. In the example below, QUEUE is the scheduling subqueue and BASE, LIMIT, MIN QUANTUM, MAX QUANTUM, BOOST and TIMESLICE are scheduling values set by the TUNE command. MIN and MAX quantums are bounds for the quantums and ACTUAL quantum is the current quantum value.

```
          QUANTUM
QUEUE BASE LIMIT MIN  MAX  ACTUAL BOOST TIMESLICE
-  - -  -   - -
CQ   152  200   1  2000  200   DECAY 200
DQ   202  238  2000 2000  2000  OSC  200
EQ   240  253  2000 2000  2000  DECAY 200
```

You may issue the SHOWQ command from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command. SHOWQ requires System Supervisor (OP) capability.

NOTE The MPE/iX Scheduler now supports the workgroup concept. However, backward compatibility is maintained through five default workgroups created by the system. The scheduling characteristics of the CS_Default, DS_Default, and ES_Default workgroups mimic those of the CS, DS, and ES scheduling subqueues. In fact, the information displayed for the CS, DS, and ES scheduling subqueues is the same information as that for the default workgroups.

Please refer to the NEWWG and SHOWWG commands for more detail.

Since SHOWQ displays limited information regarding workgroup processes, Workload Manager users should use the SHOWWG and SHOWPROC commands rather than SHOWQ. Non-Workload Manager users may choose to use these commands if they prefer the format for viewing the default workgroups.

Example

To display the active processes and the current scheduling subqueue characteristics, enter:

```
:SHOWQ;ACTIVE

DORMANT       RUNNING
Q PIN JOBNUM   Q PIN JOBNUM

         C M163 #S263
         C U215 #S256


          QUANTUM
QUEUE BASE LIMIT MIN  MAX  ACTUAL BOOST TIMESLICE
-  - -  -   - -
CQ  152  200  1   2000  200   DECAY 200
DQ  202  238  2000 2000  2000  OSC  200
EQ  240  253  2000 2000  2000  DECAY 200
```

Related Information

Commands TUNE, ALTPROC, SHOWPROC, NEWWG, ALTWG, PURGEWG, SHOWWG

Manuals *MPE/iX Intrinsics Reference Manual*

Performing System Management Tasks

SHOWTIME

Prints current time and date. (Native Mode)

Syntax

SHOWTIME

Parameters

None.

Operation Notes

Prints current time and date, as indicated by system clock.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

To display the time and date, enter:

```
SHOWTIME
MON, JUL 24, 1987, 8:47 AM
```

Related Information

Commands	SETCLOCK, SHOWCLOCK
Manuals	None

SHOWVAR

Displays specific variable names and their current values. (Native Mode)

Syntax

```
SHOWVAR[ varid] [ ,varid] ... [ ,varid]
       [job= jobID]
       [;USER | HP | ANY]
```

Parameters

varid The name of the variable for which the current value is to be displayed.

jobid The job or session number who’s variables are to be displayed. Example: #J123 or S4321. SM capability is required to see the variables from another job or session. Only user-defined variables are visible when “jobID” is specified. It is recommended to always specify the USER option when using JOB=. This adds clarity to scripts and job streams, and preserves their functionality should JOB= be enhanced to display predefined variables.

USER Selects only the user-defined variables matching each *varid*. USER is the default when *varid* is omitted. It is recommended to use USER in conjunction with JOB=, see the note above.

HP Selects only the predefined HP variables matching each *varid*.

ANY Allows all variables matching *varid* to be seen. ANY is the default when one or more *varid*s are supplied, as long as *jobid* is not specified>

Operation Notes

This command displays to $STDLIST the variables specified and their values. It displays information in the format :

`VARIABLE NAME = value`.

Users with SM capability may display user-defined variables for another job or session by using the JOB= parameter. If *jobid* matches the job ID of the user execuiting the command no restrictions are placed. Plaese specify the USER option in scripts and jobs that use JOB=. This documents the intent, and allows these scripts and jobs to function the same if JOB= is later enhanced to show predefined and use user-defined variables.

Anyone can specify the USER, HP and ANY options. However, an error is reported if HP is used in conjunction with a *jobid*.

Table 7-2 Specified Variable-ID/Result

Variable-ID	Displays
(omitted)	All variables and values that the user has set.
@	All variables.
A,B,C	Values for variables A, B, and C.
B@	All variables whose names begin with B.

You may use the wildcard characters @, #, ?, and [ ] to specify a set or range of variables or file names in many commands.

@	Specifies zero or more alphanumeric characters, or the underbar character (_). Used by itself, it specifies all possible combinations of such characters. Used with other characters it indicates all the possible names that include the specified characters (@ABC@ = all names that include ABC anywhere in the name).
#	Specifies one numeric character. A###@ = all names that begin with A followed by any three digits, followed by any combination of zero to three alphanumeric (or underbar) characters.
?	Specifies one alphanumeric character. A?# = all three-character names that begin with A, followed by an alphanumeric character, followed by a digit.
[ ]	Specifies a set or range of characters. The set may appear anywhere in the name. This range specification is not case sensitive and, therefore, [A-K] is the same as [a-k]. If you specify a null set such as [k-a], then MPE/iX gives you a warning that this specification is invalid.
@[abc]@# =	All names containing a, b, or c and ending in a single digit.
[a-k]@ =	All names that begin with any one of the letters a through k.
[n-a] =	Not valid in variables and would be flagged as an error.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command.

Examples

To display two specific variables, enter:

SHOWVAR ***firstvariable, secondvariable***

To display all variables beginning with a single alphabetic character and ending with the characters axval, enter:

```
SHOWVAR ?axval
```

To display all variables created by the user with the SETVAR, INPUT, or SETJCW command, or with the HPCIPUTVAR, PUTJCW, or SETJCW intrinsics, enter:

```
SHOWVAR
```

To display all variables created currently in the variable table, those created by the user and all predefined variables, enter:

```
SHOWVAR @
```

To display all user-defined variables for session 32. Must have SM capability, enter:

```
SHOWVAR ;job=#s32
```

To display all user-defined variables matching *s@* for job 23. Must have SM capability, enter:

```
SHOWVAR s@ ;job=J23 ;user
```

To display all user-defined variables beginning with the letter “H”. Note: the predefined HP variables, like HPPATH, are not shown, enter:

```
SHOWVAR h@ ;user
```

To display all predefined variables containing “TIME” in their names. User created variables, like MYTIME, would not be seen, enter:

```
SHOWVAR @time@ ;hp
```

Related Information

Commands	DELETEVAR, INPUT, ECHO, SETVAR, SHOWJCW
Manuals	Appendix A, "Predefined Variables in MPE/iX"
	Using the HP 3000 Series 900: Advanced Skills

=SHUTDOWN

Initiates a shutdown of MPE/iX.

Syntax

=SHUTDOWN[*system* *terminal* *dtc* *tape* *disc* *network* *other*]

Parameters

None.

Operation Notes

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 (**CTRL A**) is ineffective.

Device configuration changes that were made after the preceding load (UP, DOWN, ACCEPT, REFUSE, and spooling commands) are not retained. Configuration changes made during a system startup from tape are recorded and retained until the next system startup from tape. Newly assigned or released global resource identification numbers (RINs) are permanently recorded.

All communication lines must be closed before issuing a =SHUTDOWN command or a manual halt of the system may be necessary. Note that data is lost if a transmission is in progress when the halt is performed. If any network service (NS) lines are left open when the =SHUTDOWN command is issued, lines to the remote system remain open and any remote sessions become 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 stop operation immediately upon receiving a =SHUTDOWN command. A START RECOVERY retains spoolfiles which are printed when the system is returned online.

You can use any of the options to indicate the reason that you are shutting down the system. These options were developed to identify any possible type of system hang that might occur. For example, if you shutdown to clear a DTC hang, you can use the =SHUTDOWN *dtc* option.

Use

This command may be issued only at the physical console.

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.

CTRL A
=SHUTDOWN
10:49/#S40/25/LOGOFF
10:49/20/ALL JOBS LOGGED-OFF
```

To shut down the system in order to identify a DTC hang, use the *dtc* option. The console responds by listing shutdown messages similar to these:

```
 CTRL A
=SHUTDOWN dtc

Shutdown of operating system begins. (Shut 1)
Shutdown of user processes begins. (Shut 2)
Shutdown of jobs & sessions begins. (Shut 3)
Spoolers notified of a shutdown. (Shut 16)
Shutdown of system processes begins. (Shut 4)
Shutdown of system managers begins. (Shut 5)
Shutdown of operating system complete. (Shut 6)
```

Related Information

Commands	=LOGOFF
Manuals	*System Startup, Configuration, and Shutdown Reference Manual*

SHUTQ

Closes the spool queue(s) for the specified logical device, device name, or all members of a device class. (Native Mode)

Syntax

SHUTQ{ *ldev*[;SHOW] *devclass*[;SHOW] *devname*[;SHOW] @ }

Parameters

ldev The logical device number of the device.

devclass The device class name of the devices.

devname The device name of the device. Note that it is not possible to have a device class name and a device name that are the same. If you enter an alphanumeric character string, the command searches the device class list first, and then the device name list.

`SHOW` The `SHOW` parameter displays the current queue state (enabled or unenabled) of the devices specified with the `SHUTQ` command.

@ The @ parameter globally disables all currently open spooling queues without closing the spooling queues. Thus when the spooling queues are globally reenabled with the `OPENQ @` command, all spooling queues that were opened before being globally disabled will again be open.

Refer to the *Native Mode Spooler Reference Manual* (32650-90166) for more discussion on enabling and disabling of spooling queues.

Use the @ option without any other parameter. The `SHOW` option entered with the @ option returns an error.

Operation Notes

The `SHUTQ` command closes the spool queue(s) for a logical device or all members of a 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` and `SPOOLER` commands, which are documented in this chapter.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console unless distributed to users with the `ALLOW` or `ASSOCIATE` command.

Examples

To shut the queue for all devices in class LP, enter:

```
SHUTQ LP
```

To shut the spool queue and show the state of the queue and other information about the specified device, enter:

```
SHUTQ 6;SHOW
```

Related Information

Commands	OPENQ, STARTSPOOL, SPOOLER
Manuals	*Native Mode Spooler Reference Manual*
	Performing System Operation Tasks

8 Command Definitions SP-Z

SPEED

Sets the input and output speed for the user's terminal.

Syntax

`SPEED` *newinspeed, newoutspeed*

or

`SET SPEED` = *newspeed*

Parameters

newinspeed The new input speed in characters-per-second (CPS). The input and output speeds must always be equal. Acceptable values for *newinspeed* and *newoutspeed* are 30, 120, 240, 480, 960, and 1920.

newoutspeed The new output speed in characters-per-second (CPS). The input and output speeds must always be equal. Acceptable values for *newinspeed* and *newoutspeed* are 30, 120, 240, 480, 960, and 1920.

newspeed Used with the `SET` command to specify both input and output speeds, which are equal. Refer to the `SET` command.

Operation Notes

MPE/iX automatically senses the input/output speed of a terminal when you log on 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.

Since terminal input and output speeds are the same, it is not necessary to specify them individually.

When the `SPEED` command is entered, MPE/iX 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/iX Intrinsics Reference Manual* (32650-90028).

Use

This command may be issued from a session, program, or in BREAK. This command is not available from a job. Pressing **Break** has no effect on this command.

Examples

To manually change the speed and enter MPE (the { is a random character), enter:

```
CHANGE SPEED AND INPUT "MPE":
{
```

To change the input and output speeds to 240 CPS (2400 baud), enter:

```
SPEED 240,240
```

or

```
SET SPEED=2400
```

Related Information

Commands	SET
Manuals	*MPE/iX Intrinsics Reference Manual*

SPL

Compiles a compatibility mode SPL/V program. SPL/V is not part of the HP 3000 Series 900 Computer System 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/iX `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 Manual* (30000-90011).
- By creating a new USL file with the MPE/iX `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 Notes."

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 appear at the beginning of the source listing. For more information, refer to the *Systems Programming Language Reference Manual* (30000-90024).

Operation Notes

This command compiles an SPL program into a user subprogram library (USL) file on disk. If *textfile* is not specified, MPE/iX expects the source program to be entered from your standard input device. If *listfile* is not specified, the program output is sent to your standard list device.

Use

This command may be issued from a session, job, or program, but not in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

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 disk file SOURCE and stores the object code into the USL file OBJECT. The program listing is sent to the disk file LISTFL:

```
SPL SOURCE,OBJECT,LISTFL

SAVE OBJECT
```

Related Information

Commands	`SPLGO`, `SPLPREP`, `PREP`, `RUN`
Manuals	*Systems Programming Language Reference Manual*

SPLGO

Compiles, prepares, and executes a compatibility mode SPL/V program. SPL/V is not part of the HP 3000 Series 900 Computer System 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` 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 that appear in front of the source listing.

Operation Notes

This command compiles, prepares, and executes an SPL program. If *textfile* is omitted, MPE/iX 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`.

Use

This command may be issued from a session, job, or program but not in BREAK. Pressing **Break** suspends the execution of this command. Entering the `RESUME` command continues the execution.

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 disk file `SOURCE` and send the resulting program listing to the disk file `LISTFL`, enter:

```
SPLGO SOURCE,LISTFL
```

Related Information

Commands	SPL, SPLPREP, PREP, RUN
Manuals	*MPE Segmenter Reference Manual*
	Systems Programming Language Reference Manual

SPLPREP

Compiles and prepares a compatibility mode SPL/V program. SPL/V is not part of the HP 3000 Series 900 Computer System 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 the 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. When you omit *progfile*, the MPE segmenter creates the program file, which then resides 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/iX `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 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 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 `$STDIN` if *textfile* is not specified. If two files being merged have identical line numbers, the lines from *textfile* or from `$STDIN` overwrites 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" 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 Notes."

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 appear at the beginning of the source listing.

Operation Notes

Compiles and prepares an SPL program into a program file on disk. If *textfile* is not specified, MPE/iX expects you to enter your source program from your standard input device. If you do not specify *listfile*, your program output is sent to your standard list device.

The user subprogram library (USL) file created during compilation, $OLDPASS, is a temporary file passed directly to the MPE 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.

Use

This command may be issued from a session, job, or program but not in BREAK. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

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 disk file SFILE into the program file MYPROG. The program listing is sent to your standard list device:

```
SPLPREP SFILE,MYPROG
```

In the next example, the first positional parameter is omitted. This indicates to MPE/iX that you intend to enter the source text from your standard input device. The object code is stored in the default USL file $OLDPASS, and the prepared program segments are stored in FILEZ. $OLDPASS is then saved in the permanent file domain under the new name NUSL.

```
SPLPREP,FILEZ
SAVE $OLDPASS, NUSL
```

Related Information

Commands	`SPL`, `SPLGO`, `PREP`, `RUN`
Manuals	*MPE Segmenter Reference Manual*
	System Programming Language Reference Manual

SPOOLER

Controls spooler processes. (Native Mode)

Syntax

```
SPOOLER[ DEV=] { ldev | devclass | devname}
{ ;SHOW }
{ ;OPENQ [;SHOW]}
{ ;SHUTQ [ ;SHOW]}
{ ;START [ ;OPENQ | ;SHUTQ] [ ;SHOW]}
{ ;STOP [ ;FINISH | ;NOW] [ ;OPENQ | ;SHUTQ] [ ;SHOW]}
{ ;SUSPEND[[ ;FINISH | ;NOW][ ;NOKEEP | ;KEEP] | [ ;OFFSET= [+ | -] page] | [
;OPENQ | ;SHUTQ] [ ;SHOW]]}
{ ;RESUME [ ;OFFSET= + | - ] page ] [ ;OPENQ | ;SHUTQ [ ;SHOW]}
{ ;RELEASE [ ;OFFSET= + | - ] page ] [ ;OPENQ | ;SHUTQ [ ;SHOW]}
```

Parameters

ldev The logical device number of the spooled device.

devclass The device class name of the spooled devices. *devclass* must begin with a letter and consist of eight or fewer alphanumeric characters.

devname The device name of the spooled device. *devname* must begin with a letter and consist of eight or fewer alphanumeric characters. Note that it is not possible to have a device class name and a device name that are the same. If you enter an alphanumeric character string, the command searches the device class list first, and then the device name list.

START **OUTPUT SPOOLERS:**

The START parameter creates and activates a new spooler process to own and manage the device and print spool files destined for it. If a class is specified, then a spooling process is created and activated for each device in the class. If neither the OPENQ nor the SHUTQ option is specified, OPENQ is taken as the default.

INPUT SPOOLERS:

The START parameter creates and activates a new spooler process to own and manage the device, to read data from it, and to create job or data input spool files for later processing by a CI (job) or user process (data). If a class is specified, then a spooling process is created and activated for each device in the class.

STOP **OUTPUT SPOOLERS:**

The STOP parameter terminates the spooling process associated with the specified device. If a class is specified, then spooling processes for all devices in the specified class are terminated. A spooler in the active state first moves to the STOP pending state (shown as *STOP with the SHOW option) while it finishes its work on its current file (including any required trailer). When this is complete, or if the spooler was previously in the idle state, the spooler displays the following on the console (or the $STDLIST of an associated user) and terminates. If neither the OPENQ nor the SHUTQ option is specified, SHUTQ is taken as the default.

```
Output spooler, LDEV #ldev: Stopped.
```

You may determine the spooler state at any time by entering the following:

```
SPOOLER ldev;SHOW
```

or

```
SPOOLER devclass;SHOW
```

or

```
SPOOLER devname;SHOW
```

The STOP option is valid only if a spooler is in the ACTIVE, SUSPEND or IDLE state, or (if accelerating a previous STOP ;FINISH to STOP ;NOW) the STOP pending (*STOP) state. If neither the NOW nor the FINISH option is specified, NOW is taken as the default.

NOTE Because of the large amount of data buffered in the file system and the device, an output device may continue to print, making it appear as if the STOP parameter has not had any effect. In reality, the spooler stops sending data to the device when the command is received but must wait until all buffered data has been printed before stopping. Depending on both the content of the data and the amount of buffering, this may require a significant part of a page or even several pages. The spooler process notifies you via the following message that it has processed the command:

```
IOutput spooler, LDEV ldev:
Received a command while outputting a file
```

If the STOP is received while the spooler is printing a file, the page number of the last complete page that was printed is saved in the spool file's file label extension (FLABX). The next time the file is selected for printing by any spooler, the output resumes at the page following the page number saved in the FLABX.

INPUT SPOOLERS:

The STOP parameter terminates the spooling process associated with the specified device. If a class is specified, then spooling processes for all devices in the specified class are terminated. The spooler first moves to the STOP pending state (shown as *STOP with the SHOW option) while it finishes its work on its current file (closing and deleting it; rewinding the tape and placing it offline). When this is complete, the spooler displays the following message on the console (or the $STDLIST of an associated user) and terminates:

```
Input spooler, LDEV #ldev: Stopped.
```

You may determine the spooler state at any time by entering the following:

```
SPOOLER ldev;SHOW
```

The STOP option is valid only if a spooler is in the IDLE or ACTIVE state. Except for a short period during startup when it is in the START state, an input spooler is always in the IDLE or ACTIVE state.

The NOW, FINISH, OPENQ, and SHUTQ options are not applicable to an input spooler process and result in an error message if any is used.

SUSPEND

The SUSPEND option is valid only for output spooler processes. It suspends output to one or more spooled devices. The associated spooler processes remain alive, but inactive. A spooler in the ACTIVE state first moves to the SUSPEND pending state (shown as *SUSPEND with the SHOW option) while it finishes its work on its current file (including any required trailer). When this is complete, or if the spooler was previously in the IDLE state, the spooler displays the following on the console (or the $STDLIST of an associated user) and enters the SUSPEND state.

```
Output spooler, LDEV #ldev: Suspended.
```

If neither the NOW nor the FINISH option is specified, NOW is taken as the default. If neither the KEEP nor the NOKEEP option is specified, KEEP is taken as the default. If the OFFSET option is not specified, the spooler retains the present location in the output spool file. This is the default.

The combination of the NOW, KEEP, and no OFFSET parameters (all defaults) is a special case. When an active spooler receives this form of the SUSPEND option, it suspends after processing the current record. A subsequent SPOOLER...; RESUME with no OFFSET parameter and without an intervening SPOOLER...;RELEASE causes the spooler to resume at the next record, as if it had never been interrupted.

If a spooler process is suspended in the middle of a spool file *and* the file is not retained by the spooler, a page number is saved in the spool file's file label extension (FLABX). This page number is either the last complete page that was printed (if no OFFSET was specified) or one page prior to that specified by the final OFFSET applied to the file (with a lower limit of 0). The next time the file is selected for printing by any spooler, output resumes at the page following the page saved in the FLABX.

NOTE Because of the large amount of data buffered in the file system and the device, the device may continue to print, making it appear as if the SUSPEND parameter has not had any effect. In reality, the spooler stops sending data to the device when the command is received but must wait until all buffered data has been printed before suspending. Depending on both the content of the data and the amount of buffering, this may require a significant part of a page or even several pages.

The spooler process notifies you via the following message that it has processed the command:

```
IOutput spooler, LDEV ldev:
Received a command while outputting a file
```

If a spooler process is suspended in the middle of a spool file *and* the file is not retained by the spooler, a page number is saved in the spool file's file label extension (FLABX). This page number is either the last complete page that was printed (if no OFFSET was specified) or one page prior to that specified by the final OFFSET applied to the file (with a lower limit of 0). The next time the file is selected for printing by any spooler, output resumes at the page following the page saved in the FLABX.

RESUME — The RESUME option resumes a suspended spooler process and is therefore valid only for output spoolers. The spooler must be in the SUSPEND state. If the spooler retains a spool file when it is suspended (meaning the KEEP option was specified or taken by default), and the spool file is not subsequently released, the OFFSET option is valid. If no offset is specified with either the earlier SUSPEND or the present RESUME, then output resumes where it left off. If an OFFSET is specified at either time (or both), the spooler resumes at the final location indicated by the offsets. If OFFSET is specified and the spooler does not have a retained file, a warning is generated and the spooler prints the next available spool file from the beginning.

RELEASE — The RELEASE parameter directs a suspended output spooler to close (release) a spool file that it is currently retaining due to an earlier SUSPEND ;KEEP option. It is invalid and generates a warning if used in any other context. The OFFSET option may be used to change the resumption point of the file the next time it is selected for printing.

When the file is released by the spooler, a page number is saved in the spool file's file label extension (FLABX). This page number is either the last complete page that was printed (if no OFFSET was specified) or one page prior to that specified by the final OFFSET applied to the file (with a lower limit of 0). The next time the file is selected for printing by any spooler, output resumes at the page following the page saved in the FLABX.

FINISH — Directs the spooler to complete the currently active spool file and then suspend or stop. This option may be used only in conjunction with the SUSPEND or STOP options. If it is used in any other context, a warning is issued and the FINISH option is ignored. The FINISH parameter may not be used with either the KEEP/NOKEEP or OFFSET parameters.

The FINISH option is not valid for spooled input devices.

Either a STOP or SUSPEND that includes the FINISH option may be accelerated to a higher-priority command without waiting for the present spool file to finish printing. For example, SPOOLER...; SUSPEND; FINISH may be followed by:

```
SPOOLER...;SUSPEND;NOW
```

or

```
SPOOLER...;STOP;FINISH
```

or

```
SPOOLER...;STOP;NOW
```

Similarly, a SPOOLER...;STOP;FINISH may be accelerated to SPOOLER...;STOP;NOW. To go in the opposite direction is an error.

NOW

Directs the spooler to immediately stop the current output. This option may be used only in conjunction with the SUSPEND or STOP options. If it is used in any other context, a warning is issued. This is the default.

If NOW is used on the SUSPEND option with either the NOKEEP or OFFSET parameters, the spooler prints a trailer if required; otherwise output pauses and may be resumed later at the point of suspension.

The NOW option is not valid for spooled input devices.

KEEP

Directs the device to retain ownership of the spool file that it is currently processing. This is the default. KEEP is valid only if all three of the following conditions are satisfied:

- KEEP is used as a parameter to the SUSPEND option or, it is taken as the default.
- The spooler is actively processing a file or is suspending.
- The NOW parameter is also specified or taken by default.

If the OFFSET parameter is not specified (or this condition is taken by default), the spooler suspends after processing the current record.

NOKEEP

Directs the spooler to close the spool file that it is currently processing. NOKEEP is valid only if all three of the following conditions are satisfied:

- NOKEEP is used as a parameter to the SUSPEND option.
- The spooler is actively processing a file or is suspending.
- The NOW parameter is also specified or taken by default.

The spooler stops sending data after the current record, ejects a page, processes any specified OFFSET, saves the result of that processing (or the last completely printed page if no OFFSET was specified) in the FLABX (file label extension), prints a trailer with (INCOMPLETE) on it if trailers are enabled, and returns the file to the READY state. The next spooler that prints the file starts the first copy with the page following the page number saved in the FLABX and the file's header and trailer (if any) include (RESUMED) if printing starts anywhere but at the first page.

[+/-]*page* The *page* parameter may be used only in conjunction with the `SUSPEND`, `RESUME`, or `RELEASE` option. The *page* parameter must be an integer representing a physical page offset, either absolute or relative, within the file. Offsets are applied in the order they are entered, whether absolute or relative. If + is specified, the offset is adjusted forward relative to the current location by the number of pages specified. If – is specified, the adjustment is backward. If *page* is specified without + or -, then printing resumes at that page, absolute from the beginning of the file. No matter which combination of offsets are specified, the final location is limited by the bounds of the file.

A page is defined as follows:

- For CIPER protocol devices: a physical sheet.
- For the HP2680 or HP2688: a physical sheet (which may contain one or more logical pages).
- For serial printers: the `OFFSET` option (except for `OFFSET=1` or `OFFSET=0`, the beginning of the file) is not reliable. No error or warning message is generated if it is used on such devices; however, results are unpredictable.

  This is because page numbers are accurate only for CIPER protocol devices and HP2680 and HP2688 page printers.

The spooler's serial printer storage manager makes an approximate guess as to the correct page. However, it is only a guess based on an extremely limited interpretation of the spool file by the storage manager, because a serial printer does not return page data to its storage manager.

The storage manager does not attempt to interpret the spool file data, looking for escape sequences that may advance paper, eject a page, or change the page length or line density. This would degrade performance to an unacceptable level. Instead, it checks the carriage control character supplied as part of the user's `FWRITE` intrinisc call.

If that character is an ASCII "1" or an octal 300 (indicating skip to VFC channel 1, which by industry standard, is a page eject), it notes that this type of page control is in use and assembles its own checkpoint based on the location of this record in the spool file. If a `RESUME` with `OFFSET` is later required, it counts these checkpoints to try to find the proper restarting point. The storage manager ignores any other carriage-control character.

The page eject carriage control is not required in user data, and many applications do not use it. In this case, the storage manager is forced to assume a static number of records (60) per page. Historically, this is the number of lines that fit on a standard 11-inch page at 6 lines per inch, allowing three lines of margin at the top and the bottom of the page. This is often a flawed assumption, as the following examples show:

- For many applications (for example, A4 paper, 8 lines per inch, and so on) 60 lines per page is the wrong value.
- Other applications are designed for specific forms and manage their own paper advancement. These applications may attach a carriage-control value to a record which causes paper to advance (say) five lines after printing a line of data. The storage manager counts this as one record.
- Control records (those that affect some aspect of printer operation but do not print anything) are included in the 60 record count.

The last two examples come about because the storage manager does not interpret the data in the spool file, as mentioned earlier, and so cannot detect these situations.

In summary, if the storage manager cannot interact with the device to determine page boundaries, it uses a carriage control "1" or %300, or 60 records per page to simulate checkpoints for `SPOOLER` *ldev*`;RESUME.` Therefore, for the most consistent results with serial printers you should always include an `OFFSET=1` parameter, with the `SUSPEND` option. You can also include the parameter with a subsequent `RESUME` option, but this does not prevent another spooler process from printing the file from the "wrong" place in the meantime.

`SHOW` The `SHOW` parameter displays the status of the spooling process(es) associated with the device(s) specified. All other parameters on this command are processed first, so the `SHOW` option reflects the updated state of the process(es) at the completion of the command executor. Please refer to the note following the example below.

`OPENQ` The `OPENQ` option or parameter enables spooling for a specified logical device, device name, or all devices of a device class. This allows users to generate spool files on that device(s). See the `OPENQ` command for more information.

`OPENQ` is the default value for the `START` option.

`SHUTQ` The `SHUTQ` option or parameter disables spooling for a specified logical device, device name, or all devices of a device class. This prevents users from generating spool files on that device(s). See the `SHUTQ` command for more information.

`SHUTQ` is the default value for the `STOP` option.

Operation Notes

This command allows the user to start, stop, suspend, and resume spooler processes, and release files from the spooler process(es). At least one of the options must be specified for the `SPOOLER` command.

Spooler processes come in two varieties: input spoolers and output spoolers.

- An input spooler reads data from its device and uses that to create an input spool file. The data may consist of one or more batch jobs, data files, or any combination of the two. Input spool files are private files, meaning they are only accessible to a user running in privileged mode. They are not printed, but are used strictly as input for other processes.
- An output spooler processes output spool files files that were created by a user accessing a spooled output device such as a printer or plotter. A spooled output device processes spool files first in order of priority and then the time the spool file entered the `READY` state. Only files that have an output priority greater than the outfence are considered for output.

Because this command may affect more than one process (if applied to all devices in a class), it is possible to get errors for some of those devices and not for others. For example, if class `LP` consists of LDEVs 6, 11, and 19, and LDEV 11 is already owned by a spooler process, the command `SPOOLER LP;START` creates and activates spooler processes for LDEVs 6 and 19, but also generates the message `DEVICE 11 IS ALREADY SPOOLED`.

NOTE `SPOOLER DEV=PP` is *not* a valid command; but `SPOOLER DEV=PP;SHOW` or `SPOOLER DEV=PP; OPENQ; SHOW` are valid commands.

Use

This command may be issued from a session, job, in BREAK, or from a program. It is not breakable. It may be executed from the console or by a user to which the command has been allowed or associated.

Example

Here are some examples of the use of the `OFFSET` option:

1. A spooler is printing physical page 30 of its output, and the following sequence is entered:

   ```
   SPOOLER dev;SUSPEND;KEEP;OFFSET=-3
   SPOOLER dev;RESUME;OFFSET=-6
   ```

   Output resumes at page 21 (30-3-6=21).

2. A spooler is again on page 30 when the following sequence is entered:

   ```
   SPOOLER dev;SUSPEND;KEEP;OFFSET=-15
   SPOOLER dev;RESUME;OFFSET=20
   ```

   Output resumes at (absolute) page 20.

3. Under the same original conditions as the previous two examples:

   ```
   SPOOLER dev;SUSPEND;KEEP;OFFSET=20
   SPOOLER dev;RELEASE;OFFSET=-5
   ```

   The next time this copy is selected by a spooler, its output will start at page 15 (absolute page 20-5).

4. To ensure that a file resumes at the beginning, enter:

   ```
   SPOOLER dev;SUSPEND;NOKEEP;OFFSET=1
   ```

When you use the SHOW option, the display shows the current state of the selected spooler(s) *at the time the command executor has completed processing the command*. This means that the selected spooler(s) may not actually be in the requested state, but in a pending state on the way to achieving the requested state. This is because it has not finished acting on the command and updating the process state before the SHOW option is performed. If this is so, an asterisk (*) precedes the process state on the SHOW display to denote that the state is pending. Please refer to LDEV 14 in the example display of the SHOW option above.

An example of output using the SHOW option might be:

```
SPOOLER LP;SHOW

LDEV DEV  SPSTATE QSTATE OWNERSHIP  SPOOLID

 6 LDEV6 IDLE  OPENED OUT SPOOLER
14 LDEV14 *SUSPEND OPENED OUT SPOOLER #O237
15 LDEV15 ACTIVE OPENED OUT SPOOLER #O264
19 LDEV19    OPENED NO SPOOLER
```

Related Information

Commands	SPOOLF, LISTSPF, OPENQ, SHUTQ
Manuals	*Native Mode Spooler Reference Manual*

SPOOLF

Allows a qualified user to alter, print, or delete output spool file(s). (Native Mode)

Syntax

SPOOLF{ [[IDNAME=] { *spoolid* (*spoolid* [,*spoolid*] . . .) } [;DEV=
{ *ldev* *devclass* *devname* }] [;PRI=*outpri*] [;COPIES= *numcopies*] [;SELEQ=
{ [*select-eq*] ^*indirect_file* }] [;ALTER] [;SPSAVE] [;DEFER ;UNDEFER]
[;SHOW]]] [[IDNAME=] { *fileset* (*fileset* [,*fileset*] . . .) } [;PRINT] [;DEV=
{ *ldev* *devclass* *devname* }] [;PRI=*outpri*] [;COPIES= *numcopies*] [;SPSAVE]
[;DEFER ;UNDEFER] [;SHOW]] [[IDNAME=] { *spoolid* (*spoolid*
[,*spoolid*] . . .) } [;DELETE] [;SELEQ= { *select-eq* ^*indirect_file* }] [;SHOW]] }

Parameters

spoolid

One or more spool file IDs: #I*nnn* for input spool files or #O*nnn* for output spool files. These IDs are assigned by the spooling subsystem at spool file creation time. The # is optional. So is the O if you are displaying output spool files; that is, if you specify neither [#]O nor [#]I, [#]O*nnn* is assumed. Do not attempt to specify a qualified file name. You must enter *spoolid* or *fileset*.

There is no default.

The symbol @ may be used to specify all spool files.

The symbol O@ may be used to specify all output spool files.

The symbol I@ may be used to specify all input spool files.

If @, O@, or I@ is specified, it must be the only value supplied. @, O@, and I@ are mutually exclusive.

If you specify duplicate *spoolids*, a warning message is displayed.

If you specify multiple spool files, you must separate them by commas and enclose the set in parentheses.

A console user or a user with SM or OP capability who specifies O@ acts on all output spool files on the system. A user with AM capability who specifies O@ acts on all output spool files created by users in the same account. All other users are limited to files they have created.

fileset

Specifies the set of files to be printed. You must enter either *fileset* or *spoolid*. There is no default.

This positional parameter has this form:

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

You may use wildcards. Files that are not of the type SPOOL are ignored. An error is returned for each input spool file in the fileset.

If the file name or set is not fully qualified, the default is the user's current logon group and account. In batch mode, if any file in the set has a lockword, it must be supplied with the command. Therefore, the file cannot be part of a set that contains wildcards. This restriction does not apply in interactive mode because the system prompts the user for each required lockword. In any case, if the lockword is not correctly provided, the print option on that file fails with a warning message, and the command continues on the rest of the files, if any.

select-eq The selection equation is used as a filter on the set of spool files selected. Only spool files whose attributes satisfy all filter requirements are listed.

For example, you use the following command to delete all the output spool files to which you have access and that have less than 100 pages from `user.acct`:

```
SPOOLF O@;DELETE;SELEQ=[(OWNER=user.acct)AND(PAGES<100)]
```

Begin and end selection equations with square brackets, as shown in the preceding example.

The following command prints the output spool files to which you have access with a priority greater than 2 and that were created before September 30, 1994.

```
SPOOLF O@;PRINT;SELEQ=[(PRI>2)AND(DATE<09/30/89)]
```

Selection equations have the following format. In this display, when the expression is expanded, interpret the symbol ::= as "can be replaced by."

select-eq `::=`[*equation*]

equation **`::=`**{ *parm* { > >= < <= <> = } *value* (*equation*) NOT
equation *equation* { AND OR } *equation* }

In a selection equation, the logical operator `AND` takes precedence over the logical operator `OR`. For example, suppose you enter this command:

```
SPOOLF O@;PRINT;SELEQ=[FILEDES=REPT &
OR OWNER=BOB.ACCTG AND PRI>8]
```

In this example, `[FILEDES=REPT OR OWNER=BOB.ACCTG AND PRI>8]` is the same as `[FILEDES=REPT OR (OWNER=BOB.ACCTG AND PRI>8)]`.

value ::= Appropriate values per data type. For example, `STATE=READY` or `PRI>6`.

parm ::= The parameter (*parm*) may be one of several attributes of the spool file to be used as filters. The *parm* choices are described below.

- *parm* ::= `DEV`: `LDEV` number, device name, or device class name. You may use wildcards for device name and device class name.
- *parm* ::= `FILEDES`: Formal or actual file designator for the spool file. You may use wildcards.

  For example, if you enter the file equation below and print to it, `EPOCLONG` will be the spool file's `FILEDES`.

```
FILE EPOCLONG;DEV=EPOC;ENV=LPLONG.ENV.SYS
PRINT MYFILE,*EPOCLONG
```

You may also select files based on a null string by entering `FILEDES= ""` or `FILEDES=` ''. You must include such a construct if you specifically want to select on such an attribute. Note that `""` is not the same as `"    "`. The blank is significant.

- *parm ::=* `SPOOLID`: Spoolfile identifier number in the format #O*nnn* or #I*nnn*.

  The # is optional; but if it is used, an O or I must also be used. If it is not used, the O is also optional for output spool files; that is 123 is the same as #O123. The valid range of `SPOOLID`s is 1 ≤ *nnn* ≤ 9,999,999. (The commas are for clarity; do not enter any commas in the actual equation.)

- *parm ::=* `PAGES`: Number of pages in spool file (if known). Use a positive integer.

  The `PAGES` attribute does not apply to input spool files; therefore, any logical *condition* involving the attribute always returns FALSE when tested against an input spool file.

- *parm ::=* `FORMID`: Form name. You may use wildcards. (The *formid* is an ASCII string up to 8 characters, the first of which must be a letter.)

  You may also select files based on a null string by entering `FILEDES= ""` or `FILEDES=` ''. You must include such a construct if you specifically want to select on such an attribute. Note that `""` is not the same as `"    "`. The blank is significant. Also, this attribute does not apply to input spool files; therefore, any logical *condition* involving the attribute always returns FALSE when tested against an input spool file.

- *parm ::=* `STATE`: `READY, ACTIVE, OPEN, CREATE, PRINT, PROBLM, DELPND, SPSAVE, DEFER, XFER`.

- *parm ::=* `JOBNAME`: Job or session name under which the spool file was created. The job name can consist of up to 8 alphanumeric characters, the first of which must be a letter.

  For a job input spool file, the `JOBNAME` shown is allocated to that job, *not* the job or session that streamed it.

  You may use wildcards.

- *parm ::=* `DISP`: Disposition can be `SPSAVE` or `PURGE`. See the NOTE accompanying the `PAGES` description.

- *parm ::=*`COPIES`: Number of copies. Minimum is 1, maximum is 65,535. (The comma in 65,535 is for clarity; do not enter commas in the actual equation.)

- *parm ::=* `PRI`: Output priority. Minimum is 0, maximum is 14. See the NOTE accompanying the `PAGES` description.

- *parm ::=* `JOBNUM`: Job or session number under which the spool file was created, for example: #S257, #J329, or J*n* (the "#" is optional). 1 ≤ *n* ≤ 16,383. (The comma is for clarity; do not enter any commas in the actual equation.)

  For a job input spool file, the `JOBNUM` shown is allocated to the job, *not* the job or session that streamed it.

You may use some wildcards; J@ accepts all jobs, S@ accepts all sessions. J'@ and S'@ are also allowed, The apostrophe (') indicates an imported spool file or a spool file recovered during `START NORECOVERY`.

- *parm* ::= `RECS`: Number of records in the spool file. A positive integer is expected.
- *parm* ::= `OWNER`: The user under which the spool file was created. The format of the *owner* is *user.account*. If the account is not specified, the user's current account is assumed. You may use wildcards.

  For a job input spool file, the `OWNER` is the user logon for the job, *not* the job or session that streamed it.
- *parm* ::= `JOBABORT`: Select based on whether this is the `$STDLIST` of a job that aborted when an error was encountered when no `CONTINUE` was in effect.

  Valid values are TRUE and FALSE. Only "=" and "<>" are allowed as relational operators. This attribute does not apply to input spool files; therefore, any logical *condition* involving the attribute always returns FALSE when tested against an input spool file.
- *parm* ::= `DATE`: Creation date in the format *mm/dd/yy* or *mm/dd/year*. Note that the year can be in the form of *yy*, as in 10/10/88, or in the form of *year*, as in 10/10/1988; both are legal syntax for the *date* parameter.

indirect_file Specifies the name of a file containing the selection equation. It must be preceded by a caret (^). The selection equation contained in the file may not exceed 509 characters in length, including the brackets in which it must reside. There is no restriction on the indirect file code. If the record size exceeds 509, only 509 characters per record are read and a warning is issued. Backreferencing to a formal file designator is also allowed for an *indirect_file* name; that is, ^**filename* is also allowed. Any file is accepted as an *indirect_file*, unless the file system returns an error from `FOPEN` or `FREAD`.

There is no limit to the number of records in the *indirect_file*, only the total character count.

Records are processed as follows:

- Leading and trailing blanks are stripped.
- If the last non-blank character is an ampersand (&), it is also stripped; otherwise, one blank is added back to the end of the record as a delimiter.
- The character count of the record is added to that of the records processed previously. If the total character count exceeds 509, an error is returned. If the total is less than 509, the current record is appended to previous records.
- This process repeats until either 509 characters have been counted or the end-of-file is detected. Records terminating with or without ampersands may be mixed as desired in the indirect file.

- If the resulting string is ≤509 characters, it is parsed.
- If the parser detects a syntax error, or if any non-blank character follows the closing bracket (]) of the *select-eq*, an error is returned and the *select-eq* is not processed.

ALTER The ALTER option alters the characteristics of specified output spool files. Private output spool files may be altered in a limited fashion; only the keywords PRI, DEFER, and UNDEFER are allowed. A system manager (SM) user may also specify DEV=.

NOTE You cannot alter the attributes of spool files in the SPSAVE state.

If you use the DEFER or DEV keyword on a spool file that is being printed, the spooler process printing the file is interrupted. The spooler process saves the page number of the last complete page that was printed in the spool files's file label extension (FLABX). The next time the file is selected for printing by any spooler, output resumes at the page saved in the FLABX.

Because of the large amount of data buffered in the file system and the device, an output device may continue to print, making it appear as if the DEFER or DEV keyword has not had any effect. In reality, the spooler stops sending data to the device when the command is received but must wait until all buffered data has been printed before releasing the spool file.

Depending on both the content of the data and the amount of buffering, this may require a significant part of a page or even several pages.

PRINT The PRINT option copies the specified filesets to the HPSPOOL account and links the new output spool files into the spool queues for printing. It is especially useful for generating more copies of a spool file in the SPSAVE state.

If the target device or class information exists in the file label extension, that device or class is used as the default. The DEV= option may be used to override this default. If there is no target device in the file label extension or the device specified is not valid, the DEV= parameter must be specified or an error message results. The default values of PRI (8) and COPIES (1) may also be overridden by user-specified parameters.

You must have nonshareable (ND) capability to use the SPOOLF...;PRINT command. Private files cannot be printed using the PRINT option.

DELETE The DELETE option purges all specified private or nonprivate spool files to which the user has access from the system.

If a spool file is not in use (opened by a user, or being printed or stored), it is purged immediately. If it is in use, it is placed in DELPND state. Any spooler process printing it is notified, and printing stops at that point. Each of these files is deleted when its last user closes it, except in the case of STORE, as described below.

NOTE Because of the large amount of data buffered in the file system and the device, an output device may continue to print, making it appear as if the DELETE option has not had any effect. In reality, the spooler stops sending data to the device when the command is received but must wait until all buffered data has been printed before stopping.

Depending on both the content of the data and the amount of buffering, this may require a significant part of a page or even several pages.

ldev Specifies the logical device number of the spool file's new destination device.

If the spool file is in the PRINT state, it is returned to the READY state. It may immediately enter the PRINT state on *ldev* if all requirements are met.

Printing of a spool file is interrupted only if the newly specified target *ldev*, *devclass*, or *devname* is different than the previous target *ldev*, *devclass*, or *devname*.

devclass Specifies the new destination device class name for the spool file. If the spool file is in the PRINT state, it is returned to the READY state. It may immediately enter the PRINT state on a device in *devclass* if all requirements are met.

The *devclass* parameter must begin with a letter and consist of eight or fewer alphanumeric characters. Note that MPE/iX does not allow the same name to be configured as a device class name and a device name. See the NOTE accompanying *ldev*.

devname Specifies the device name of the spool file's new destination device. If the spool file is in the PRINT state, it is returned to the READY state. It may immediately enter the PRINT state on *devname* if all requirements are met. Note that this occurs even if *devname* is the same as the device currently printing the file.

The *devname* parameter must begin with a letter and consist of eight or fewer alphanumeric characters. Note that MPE/iX does not allow the same name to be configured as a device class name and a device name. See the NOTE accompanying *ldev*.

outpri Specifies the output priority of the designated spool files, where 0 is the lowest and 14 is the highest. Only an OP user or the console can specify an *outpri* of 14; other users are limited to 13.

The default is 8 with the PRINT option and no change for the ALTER option.

numcopies Specifies the number of copies of the designated spool files to be printed. The allowable range is 1 through 65,535. (The comma is for clarity; do not enter any commas in the actual command.)

The default is 1 for the PRINT option and no change for the ALTER option.

SPSAVE

The SPSAVE option specifies that the selected spool files are not to be deleted after their last copy has printed. Instead they are retained in the HPSPOOL account in the SPSAVE state until deleted manually. Among other advantages, this option allows documents to be copied to user file space, to be reprinted without being reformatted, and so on.

Private spool files may not be saved.

When a file enters the SPSAVE state, its priority is set to 8 and its number of copies is set to 1. This is so that it will have the proper defaults should it be printed later.

DEFER

The DEFER option changes the spool file's state to DEFER. If it is currently in the PRINT state, its spooler is notified and printing stops at that point. (See the note about buffer retention under the DELETE option.) The spool file's priority remains unchanged. If this option is used with the PRINT option, the spool file is copied to OUT.HPSPOOL and linked to the spooling system, but the state of the spool file is DEFER. The spool file is not printed until a subsequent SPOOLF...;UNDEFER is entered.

NOTE If the DEFER option is used on any file in the CREATE state (opened for original creation), the spool file only enters the DEFER state after it is completed (closed for the last time).

UNDEFER

The UNDEFER option changes a spool file's state from DEFER to READY and causes a spooler to start printing it if the spool file is qualified for an idle printer to print. The spool file's priority remains unchanged.

SHOW

The SHOW option allows a user to display the results of the SPOOLF command. All other parameters are processed before the SHOW. Here is an example:

```
:SPOOLF O@;SELEQ=[DEV=16];ALTER;PRI=8;SHOW

SPOOLID JOBNUM FILEDES PRI COPIES DEV  STATE RSPFN  OWNER
#O414  J5  $STDLIST 8  1 00000016 READY   ALIX.MKT
#O416  J7  HOTSTUFF 8  2 00000016 READY   JACK.SALES
```

Operation Notes

Input spool file attributes cannot be altered, but input spooled DATA files can be deleted. Private spool files may be altered in a limited fashion; only the keywords PRI, DEFER, UNDEFER, and DELETE are allowed. If the user has system manager capability, DEV= is also allowed.

The SPOOLF...;ALTER command can be used on problem state spool files to alter the device attribute so that the spool file becomes ready again. Most of the time, the spool file is in the problem state because the target device of the spool file is invalid.

Use

This command may be issued from a session, job, or program, or in BREAK. `SPOOLF ...;SHOW` is breakable. However, you cannot stop the actions by pressing **BREAK**. The files you can access with the `SPOOLF` command depend on your capabilities.

Example

Related Information

Commands	`SPOOLER`, `LISTSPF`, `LISTFILE`, `ALTSPOOLFILE`, `DELETESPOOLFILE`
Manuals	*Native Mode Spooler Reference Manual*

STARTSESS

Creates a session on the specified device, if the user has programmatic sessions (PS) capability.

Syntax

```
STARTSESS ldev [ sessionname,] user [ /userpass] .acct [/acctpass][ ,group [/grouppass]]
[ ;TERM={termtype}][ [;TIME=cpusecs]
[ ;PRI= {BS | CS | DS | ES}][{ ;INPRI=inputpriority | ;HIPRI}]
[ ;NOWAIT][ ;INFO=ciinfo][ ;PARM=ciparm]
```

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 distributed system (DS) pseudo terminal. The terminal must be configured as type 16 and as subtype 0 or 4.

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 log on to 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. If a password exists, but is *not* supplied in the command syntax, STARTSESS will prompt you for it if:

- STARTSESS is invoked from a session.
- Neither $STDIN nor $STDLIST is redirected.
- STARTSESS is a first level command (it is not nested within a second level STREAM command, or any other second level command such as JOB).

If the password is supplied in the command syntax it must be preceded by a slash (/).

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. If a password exists, but is *not* supplied in the command syntax, `STARTSESS` will prompt you for it if:

- `STARTSESS` is invoked from a session.
- Neither $STDIN nor $STDLIST is redirected.
- `STARTSESS` is a first level command (it is not nested within a second level `STREAM` command, or any other second level command such as `JOB`).

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 exists. If a password exists, but is *not* supplied in the command syntax, `STARTSESS` will prompt you for it if:

- `STARTSESS` is invoked from a session.
- Neither $STDIN nor $STDLIST is redirected.
- `STARTSESS` is a first level command (it is not nested within a second level `STREAM` command, or any other second level command such as `JOB`).

If the password is supplied in the command syntax it must be preceded by a slash (/).

termtype Determines terminal-type characteristics. The value of the *termtype* parameter determines the type of terminal used for input. MPE/iX uses this parameter to determine device-dependent characteristics such as delay factors for carriage returns. The value must be 10, 18, 20, or 21. 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*.

If group and/or account names are omitted, the proposed logon group and/or account name is substituted. Refer to appendix C, "Terminal and Printer Types."

cpusecs Maximum CPU-time that a session may use, entered in seconds. When the limit is reached, the session is aborted. It must be a value from 1 to 32,767, 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 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 user name by the system, MPE/iX assigns the highest priority possible below BS. DS and ES are intended primarily for batch jobs; their use for sessions is generally discouraged.

CAUTION Care should be used in assigning the BS queue, because processes in this priority class lock out other processes. For information on the guidelines for these priority queues, refer to the `TUNE` command in this chapter. 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 jobfence. The *inputpriority* option takes effect at logon time and must be a value from 1 (lowest priority) to 13 (highest priority). If you supply a value less than or equal to the current jobfence set by the system operator, the 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 jobfence, or it can be used to override the session limit:

- When using the HIPRI option to override the jobfence, the system first checks to see if you have system manager (SM) or system supervisor (OP) capability. The user who has either of these capabilities is logged on, and the INPRI defaults to the system jobfence and execution limit. If you do not have either of these capabilities, the system attempts to log you on using INPRI=13 and succeeds if the jobfence is 12 or less, and the session limit is not exceeded.
- In attempting to override the session limit (to log on 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 does not override the session limit automatically.

If the HIPRI option is used without SM or OP capability, the following warning is 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.

ciinfo An INFO string to be passed to the command interpreter. For the MPE/iX CI, it is the first command to be executed by the command interpreter. This parameter replaces the :( ) COMMAND LOGON command and approximates its function. The :( ) COMMAND LOGON command caused the session to terminate after executing the specified command. In contrast, the *ciinfo* parameter does not terminate the session unless *ciparm* is set to 1, 3, or 5.

Running the CI as a child process in this way restricts the flexibility of *ciparm*. More flexibility is available by running the CI as a standalone program.

ciparm The command interpreter parameter number you wish to use. If you are using the MPE/iX command interpreter, the numbers accepted are:

0, 2, 4 Logon UDCs are executed and the CI banner and the WELCOME message are displayed. Default.

1, 3, 5 Same as 0, but the CI terminates after processing the *info*= string.

-1 UDCs are not cataloged. The CI banner and the WELCOME message are not displayed. Invoking this level requires system manager (SM) capability.

-2 Same as -1, but the CI terminates after processing the *info*= command. Invoking this level requires system manager (SM) capability.

Any other value is treated as zero (0). The MPE/iX CI distinguishes between a *ciparm* 1, 3, 5 and 0, 2, 4 when it is run from within the CI, that is, after the session has logged on.

If a user *without* SM capability uses -1 or -2, the system substitutes a parameter value of 0. An error message is *not* produced.

Operation Notes

This command is used to create a session at any terminal on the system. The effect is the same as if a user had logged on at the target terminal.

STARTSESS prompts for any necessary passwords that are not supplied in the command syntax if:

- STARTSESS is invoked from a session.
- Neither $STDIN nor $STDLIST is redirected.
- STARTSESS is a first level command (it is not nested within any second level command, such as JOB).

NOTE The target terminal must be turned on and available, and no other user may be logged on.

No speed sensing is done for the target terminal, so it must be set at the configured baud rate.

When a session is started on the designated terminal, by default it waits for a **Return** before printing to the terminal, unless NOWAIT is specified.

Use

This command is available from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. Programmatic sessions (PS) capability is required to use this command.

Example

To start a session named CH5, with the *username* ERNST, *accountname* UDET, *groupname* JASTA11, and *grouppass* PASS on LDEV 21, enter:

```
STARTSESS 21;CH5,ERNST.UDET,JASTA11/PASS
```

Related Information

Commands	TUNE
Manuals	*Process Management Programmer's Guide*
	Performing System Operation Tasks

STARTSPOOL

Initiates the spooler process for a device.

Syntax

STARTSPOOL[{ *ldev*[;SHUTQ] *devclass* }]

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 that reference the device.

devclass The device class specified in the I/O configuration. Only this device class becomes spooled; it does not affect other device classes or any devices in the class.

`SHUTQ` The spooler prints files waiting in the queue for device *ldev*, but prevents the creation of new spool files. The `SHUTQ` parameter is valid for *ldev* only.

Operation Notes

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 is controlled by the spooler. The logical device itself is not controlled, unless a `STARTSPOOL` has also been issued for the corresponding *ldev*.

If spooling is enabled only on the laser printer's *ldev*, and spooling stops as a result of an I/O error, no new spool files are 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).

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only at the console unless distributed to users with the `ALLOW` or `ASSOCIATE` command.

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 spool files, enter:

```
STARTSPOOL 6;SHUTQ
```

Related Information

Commands	`STOPSPOOL`
Manuals	*Performing System Operation Tasks*

STOPSPOOL

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 nonspooled I/O. When a logical device is assigned to more than one device class, to restart spooling for a specific device class issue an explicit `STARTSPOOL` request for that class.

devclass The device class specified in the system I/O configuration. Subsequent I/O directed to this device class does not take place to/from a spool file. I/O goes directly to/from a logical device if one is available within the device class. If none is available, the program is unable to open the file.

`OPENQ` May be specified with the *ldev* parameter only. The spooler process leaves the queue in an OPEN state, or opens the queue if previously shut. Default is `SHUTQ`.

Operation Notes

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 shuts the queue for that device class. When you specify *ldev*, however, you may shut the spooling queue or leave it open. Default is `SHUTQ`.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console unless distributed to users with the `ALLOW` or `ASSOCIATE` command.

Examples

To terminate spooling to logical device number 6 and cause the spooler process to relinquish control of that device, enter:

```
STOPSPOOL 6
```

Spooling also terminates for any device class that references this device unless `STARTSPOOL` has been issued for a specific device class.

To stop directing output for device class `LP` to a spool file (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
```

Related Information

Commands	STARTSPOOL
Manuals	*STORE and TurboSTORE/iX Manual*
	Volume Management Reference Manual

STORE

Copies disk files onto backup media so that they can be recovered with RESTORE.

Syntax

STORE[[*filesetlist*] [; [*storefile*] [;*option* [;*option*[...]]]]]

where *option* is:

[;SHOW[=*showparmlist*]]

[;ONERR[OR] = { REDO QUIT }]

[{ ;DATE<=*accdate* ;DATE>=*moddate* }]

[;PURGE]

[;PROGRESS [=*minutes*]]

[;DIRECTORY]

[;FILES=*maxfiles*]

[;TRANSPORT[=MPEXL]]

[;COPYACD] [;NOACD]

[;FCRANGE=*filecode/filecode* [,...]]

[;MAXTAPEBUF]

[;NOTIFY]

[;ONVS=*volumesetname*[,*volumesetname* [,...]]]

[;SPLITVS=*split_setname*[,*split_setname*]]

[;RENAME]

[;TREE] [;NOTREE] [;STOREDIR[ECTORY] =*directoryname*]

[;NOSTOREDIR[ECTORY]] [;PART[IAL] DB] [;STATISTICS]

[;INTER] [;STORESET= (*device* [,...])]

[;INTER]

[;STORESET= (*device* [,...]) [, (*device* [,...]) [,...]]]

The following parameters are available with TurboStore/iX and TurboSTORE/iX True-Online Backup products only:

[;COMPRESS[=*compressionparmlist*]]

[;MOSET= (*ldev* [,...]) [, (*ldev* [,...]) [,...]]]

[;NAME=*backupname*]

The following parameters are available with the TurboSTORE/iX 7x24 True-Online Backup product only:

[;ONLINE[={ START} [,*time*] [,ASK]]] { END}

[;LOGVOLSET=*volumesetname*]

Parameters

filesetlist Specifies the set of files to be stored. The default set is @ meaning all files in the current working directory (CWD) regardless of the user's capabilities. If the `DIRECTORY` option is specified, the default file set is empty (no files).

The *filesetlist* parameter has the form shown below:

filesetitem[,*filesetitem*[...

where *filesetitem* can be ^*indirectfile* or *fileset*.

indirectfile A file name that backreferences a disk file. The syntax is

^*indirectfile* or !*indirectfile*

This file may consist of *fileset(s)* and *option(s)*, but only options can appear after the first semicolon (:) on each line. An option specified on one line will operate on all files in the *filesetlist*.

^*indirectfile* is the preferred format. If you use !*indirectfile*, the CI will interpret this as a variable reference, so you will have to specify !!*indirectfile* instead.

fileset Specifies a set of files to be stored and optionally those files to be excluded from the `STORE` operation. The *fileset* parameter has the form:

filestostore[-*filestoexclude*[-*filestoexclude*[-...]

An alternate syntax exists for use with the `RENAME` option:

filestostore[-*filestoexclude*[-...[=*targetname*]

The system stores any file that matches *filestostore* unless the file also matches *filestoexclude*, which specifies files to be excluded from the STORE operation. You may specify an unlimited number of *filestoexclude*.

Since "-" is a valid character for HFS syntax file names, a blank character must separate it from HFS file sets to obtain the special negative file set meaning.

filestostore
filestoexclude Both *filestostore* and *filestoexclude* may be entered in MPE or HFS syntax. Wildcards are permitted for both MPE and HFS syntax.

The MPE syntax is as follows:

filename[.*groupname*[.*accountname*

A lockword may be specified for files to be stored, in the form:

filename/lockword.group.account

The HFS syntax is as follows:

/dir_lev_1/dir_lev_2/.../dir_lev_i/.../filedesig

or

./dir_lev_i/dir_lev_j/.../dir_lev_k/.../filedesig

If the name begins with a dot (.), then it is fully qualified by replacing the dot with the current working directory (CWD).

Each of the components *dir_lev_i* and *filedesig* can have a maximum of 255 characters with the full path name being restricted to 1023 characters. Each of the components *dir_lev_i* and *filedesig* can use the following characters:

Letters a to z

Letters A to Z

Digits 0 to 9

Special characters - _ .

For HFS name syntax, the lowercase letters are treated distinctly from the uppercase letters (no upshifting). Names in MPE syntax are upshifted.

Both MPE and HFS name components can use the characters @, #, and ? as wildcard characters. These wildcard characters have the following meaning:

`@`	specifies zero or more alphanumeric characters.
`#`	specifies one numeric character.
`?`	specifies one alphanumeric character.

These wildcard characters can be used as follows

`n@`	Store all files starting with the character `n`.
`@n`	Store all files ending with the character `n`.
`n##...#`	Store all files starting with character `n` followed by up to seven digits (useful for storing all EDIT/3000 temporary files).
`n@x`	Store all files starting with the character `n` and ending with the character `x`.
`?n@`	Store all files whose second character is `n`.
`n?`	store all two-character files starting with the character `n`.
`?n`	Store all two-character files ending with the character `n`.

Also, character sets may be specified in the following syntax:

`[ct]`	specifies letter `c` or `t`.

`[c-t]` specifies any letter from range `c` to `t`.

`[e-g1]` specifies any letter range `e` to `g` or digit `1`.

Examples of using character sets are:

`[A-C]@` Store all files that begin with the letters `A, B,` or `C`.

`myset[e-g1]` Store all files that begin with the name `myset` and end in `e, f,` or `g`, or `1`.

`myset`
`[d-e1-6]` Store all files that begin with the name `myset` and end in `d` or `e`, or `1, 2, 3, 4, 5,` or `6`.

You may specify up to a maximum of sixteen characters for each character set and you may not nest brackets. You may not use character sets with the `TRANSPORT` option.

A character set specifies a range for only one (1) ASCII character. The range `[a-d]@` gets all files that begin with the letter `a` through the letter `d`. The ranged `[ad-de]` may cause unpredictable results.

Since the hyphen (-) is a valid character for HFS syntax file names, it is allowed inside a character set, immediately following a left bracket ([) or preceding a right bracket (]). When specified between two characters, the hyphen implies a range of characters.

Specifying Database Files

When specifying TurboIMAGE and ALLBASE/SQL databases to be stored, only the root file or DBCon file needs to be specified. `STORE` will determine which other files belong to that database, and will store all of them. If dataset file(s) are specified without specifying a root file, then a warning will be printed for each file, and they will not be stored. Individual database files can be stored without the root file by specifying the `;PARTIALDB` option on the `STORE` command line.

Database corruption may result if not all database files are restored from a backup. Be sure that you only want to restore certain database files before overriding the default behavior with `;PARTIALDB`.

MPE and HFS Naming Equivalences

When an MPE name component is a single @ wildcard, the @ will be "folded" to include all MPE and HFS named files at that level and below. To specifiy only MPE-named files, use ?@ instead.

MPE wildcards are not expanded in filestoexclude. This means that `@.@.@-@.@.@` is NOT an empty fileset. It contains all of the HFS named files on the system.

A fileset may be entered in any of the following formats and may use wildcard characters. Equivalent MPE and HFS formats are grouped together as follows.

file.group.acct/ACCT/
GROUP/FILE' One particular file in one particular group in one particular account.

file.group/LOGON-
ACCT/GROUP/
FILE One particular file in one particular group in the logon account.

file
./FILE One particular file in the logon group and account.

@.group.acct
/ACCT/GROUP/ All files (MPE and HFS) in one particular group in one particular account.

?@.group.acct All MPE name files in one particular group in one particular account.

@.group/LOGON-
ACCT/GROUP/ All the files (MPE and HFS) in one particular group in the logon account.

?@.group All MPE named files in one particular group in the logon account.

@.@.acct
/ACCT/ All the files (MPE and HFS) in all the groups in one particular account, plus all the files and directories under the specified account.

thisisit.@.account Any MPE file named `thisisit` in all groups in one particular account.

?@.@.acct All MPE named files in all the groups in one particular account.

@ All (MPE and HFS) files in the CWD. This is the default for everyone, regardless of permissions.

@.@ All (MPE and HFS) files in the logon account.

@.@.@ All the files and directories (MPE and HFS) on the system.

?@.@.@ All MPE named files on the system.

targetname Specifies the name and creator for the file on the store media. The targetname parameter has the form:

filename[:*creator*[.*creatoraccount*

The filename can be any legal MPE filename or HFS pathname. The *creator* and *creatoraccount* must be legal creator and account names, respectively. The only wildcard character allowed is a single @ for each component of the filename, *creator* or *creatoraccount*. The wildcard character

@ indicates that the source value for that component should be used. An HFS pathname which ends in a / is considered an HFS directory and no wildcard characters are allowed in the filename.

The RENAME option must be specified if the *targetname* is used.

storefile The name of the device to which the stored files are to be written. This may be any magnetic tape or DDS device. This file must be backreferenced, by using an asterisk (*). You must do this by using a File equation before invoking STORE.

A message is displayed on the system console requesting the operator to mount the tape identified by the *storefile* parameter and to allocate the device.

The *storefile* can now reference a remote device. For example, if you issue the following commands, NM Store will store all files to the specified remote device.

```
:FILE REMOTE;DEV=REMSYS#TAPE
:STORE @;*REMOTE;SHOW
```

NM STORE will store all files to the specified remote device. Although the initial tape mount request will appear on the remote console, all of the STORE console messages will be displayed on the local console. Currently, labeled tapes and Magneto-optical devices cannot be used for remote backup.

A message is displayed on the system console requesting the operator to mount the tape identified by the *storefile* parameter and to allocate the device.

If *storefile* is not supplied and the STORESET option is not used, then STORE creates a default *storefile* name. The default file name is the user's logon username. No file equation is used.

Sequential and parallel devices are specified with the STORESET option. Similarly, magneto-optical devices are specified using the MOSET option. *Storefile* should not be specified when using STORESET or MOSET.

If using TurboSTORE/iX 7x24 True-Online Backup, a disk file can also be specified with a file equation for *storefile*. An example of such a file equation would be:

```
:FILE MYDISC=DISCBACK.DAILY.BACKUP;DEV=DISC
```

Note that DEV=DISC must be specified for STORE to recover files from disk backups. All other information in the file equation will be ignored by STORE. STORE creates a binary, fixed record file containing the backup data. This disk file can be restored using the same file equation for RESTORE.

By default, STORE creates the disk file with a 4Gig limit. If the data being stored exceeds this, or an existing file with a smaller limit is specified for the backup, then STORE will create and write to additional disk files. It will append the "reel" number to the disk file name originally specified. For example, if the backup disk file specified was /SYS/BACKUPS/DAILY, and STORE ran out of room, it would create /SYS/BACKUPS/DAILY.2, /SYS/BACKUPS/DAILY.3, and so on. The additional files are HFS-named files.

TurboSTORE/iX 7x24 True-Online Backup must be used to create disk backups.

SHOW Specifies that STORE is to report information for every file that is stored. If you omit the SHOW parameter, then only the names of the files not stored are listed, along with the number of files stored and the number of files not stored. This listing is sent to $STDLIST (formal file designator SYSLIST) unless a FILE command is entered to send the listing to some other device. For instance, if you enter the following file equation before issuing the STORE command, the listing will be sent to a line printer.

```
FILE SYSLIST; DEV=LP
```

showparmlist Tells STORE what information to display for the files that are stored. If you specify ;SHOW and omit *showparmlist*, then the default is SHORT if the recordsize of SYSLIST is less than 132 characters, or LONG if the recordsize is equal to or greater than 132 characters. The format for *showparmlist* is:

showparm [,*showparm*[,*showparm*[,...]

where *showparm* may be one of the options described below.

If an HFS-named file is specified in the *filesetlist*, or the expansion of a wildcard includes an HFS-named file, then an HFS-style output listing will be used. This listing shows the same information as the MPE format, but puts the name of the file at the right end of the listing to allow for longer HFS names. If an HFS name is too long to fit in the record size of the output file, it will be wrapped onto the next line. Wrapping is signified by a "*" as the last character on the line.

showparm

SHORT Overrides a default of LONG and displays file name, group name, account name or the fully qualified path name, volume restrictions, file size (in sectors), file code, and media number.

LONG Overrides a default of SHORT and displays all the information that SHORT does and adds record size, blocking factor, number of extents allowed, allocated, end-of-file, and file starting and ending media number.

NAMESONLY Displays only the filename and the starting and ending media number. NAMESONLY is not allowed with SHORT or LONG.

DATES Displays the creation date, the last date of access, and the last date of modification.

SECURITY For MPE format listings, causes SHOW to display the creator and the file access matrix for all the files which do not have an active ACD. For files with active ACDs only, the phrase *ACD EXISTS* is displayed.

For HFS format listing, the phrase *ACD EXISTS* or *ACD ABSENT* is displayed, depending on whether the file has an ACD.

PATH Forces all file listings to be in HFS format. The full HFS pathname is displayed instead of MPE style names.

OFFLINE Sends an additional copy to the format file designator OFFLINE, which defaults to device LP.

If a 7x24 True-Online backup is performed with the sync point at the end of the backup, additional information will be written to the listing. This information consists of a single character immediately following the volume restrictions. The possible values and meanings of this character are as follows:

^	This file has after image file label data
#	This file has after image file data
+	This file was added to the backup before the 7x24 sync point
-	This file was removed from the backup before the 7x24 sync point

For more information on performing 7x24 True-Online backups, refer the the *Store and TurboSTORE/iX Manual* (30319-90001).

ONERROR Tells STORE what to do if there is a tape write error. If you omit this parameter, then the default option is REDO. ONERR is a synonym for ONERROR.

QUIT Tells STORE to abort after a tape write error.

REDO Tells STORE to perform error recovery on the tape write error. First the tape is rewound, and a bad record is written to the beginning of the tape. The tape is then unloaded, and a new tape is requested. STORE then continues rewriting the files that were on the damaged media.

moddate or accdate Instructs STORE to store only selected files. A *moddate* value (indicated by >=, equal to or greater than) limits the STORE to those files that were modified on or after a particular date.

An *accdate* value (indicated by <=, less than or equal to) limits the STORE to those files that were accessed on or before a particular date.

The date is expressed in the form mm/dd/yy[yy]. The year may be expressed in two or four digits (for example, 87 or 1987).

This option cannot be used for files that are attached to a log set.

PURGE — Instructs STORE to purge all the files that were successfully stored, after the Store operation has ended. In an interactive session, MPE/iX prompts the user to enter any lockwords that have been omitted if the user does not have system manager, system supervisor, or account manager capabilities. In a job, if the user does not have SM, AM, or OP capability, the lockword(s) must be provided.

A file with a negative file code can be purged only by a user who has Privileged Mode (PM) capability.

If a file cannot be purged, a file system error message is sent to the user, stating that the file was not purged.

PROGRESS — Instructs STORE to report its progress at regular intervals by displaying the message `STORE OPERATION IS nnn% COMPLETE`. For interactive users, this message is displayed on $STDLIST. For jobs, this message is sent to the system console.

minutes — A positive number specifying the number of minutes between progress messages. The maximum is 60. The default (and minimum) is 1 minute.

DIRECTORY — Specifies that the file system directory plus all HFS directories are to be stored. This option requires system manager (SM) or system supervisor (OP) capability.

If ONVS or SPLITVS is not specified, the DIRECTORY defaults to storing the system directory. Otherwise, the directories of the specified volume sets are stored. This way, operators and manager can store or copy private volume sets in their entirety.

FILES=*maxfiles* — Maximum number of MPE/iX files that may be stored when using the TRANSPORT option. The default is 4000. If the number of files requested is greater than this number, an error occurs and the store is not performed.

This parameter is ignored when you are storing without the TRANSPORT option. In that case, no limit is imposed.

TRANSPORT — Specifies that an MPE V/E compatible tape is to be written. TRANSPORT invokes the CMSTORE program, which limits the MPE/iX STORE command to the capabilities of the MPE V/E STORE command syntax. Also, you may specify only one file to exclude from the store.

The TRANSPORT option may also be activated by setting the CI variable HPCMSTORE to TRUE.

This option is not available if you have specified DIRECTORY, FCRANGE, SPLITVS, MAXTAPEBUF, STORESET, INTER, COMPRESS, ONLINE, MOSET, NAME, ONVS, TREE, or NOTREE options.

MPEXL — (optional) If MPEXL is specified, then STORE writes out MPE XL compatible media. If the TRANSPORT parameter is used and MPEXL is not specified, then MPE V compatible media is produced. This option is used to facilitate transport of files with a later version attribute to older systems. At present, ACDs are the only attributes that are translated.

COPYACD — Indicates that the access control definition (ACD), if one exists, will be stored with the file. This is the default parameter .

NOACD — Indicates that the access control definition (ACD) should not be stored with the file. If this parameter is not specified, the ACD will be stored.

FCRANGE — The set of file code ranges that are to be stored.

filecode/*filecode* — A file code range. A filecode is an integer between -32768 and 32767. ;FCRANGE=1000/1040 would store only those files having file codes between 1000 and 1040. You may specify a maximum of eight file code ranges.

MAXTAPEBUF — Directs STORE to use the maximum available buffer size during the store operation. Currently, the maximum tape buffer sizes for the following tape drives are (in Kilobytes):

```
7974  16  7978B  32 DDS  32
7976  16  7979  32 MO  32
7978A  16  7980  32 3480  32
```

This option is also available by setting the CI variable HPMAXTAPEBUF to TRUE.

NOTIFY — Notifies the user when the files being stored are available to be accessed. If an ONLINE store is being done, this notification is done at the end of the attach period, when the FILES ARE NOW
FREE message is sent to the console. For a non-ONLINE store, the notification is done at the successful end of the entire store. Notification is done by streaming a job specified by the formal file designator NOTIFY. This file equation should be set up before the store command is run:

```
:FILE NOTIFY=MYJOB.PUB.SYS
```

STORE will attempt to issue a STREAM *NOTIFY at the appropriate time. If STORE is being run from a session, and the job requires passwords, the user will be prompted to enter them. If STORE is being run in a job and passwords are required, the job will fail to stream. The output from streaming the job is sent to $STDLIST. If the job fails to stream for any reason, STORE will print the error, but will not abort.

ONVS — ON Volume Set. Specifies that only files in the *filesetlist* that reside on the volume specified are to be stored.

The example below stores the files on VOLUME_SET_A.

```
:STORE @.@.@;*TAPE; ONVS=VOLUME_SET_A
```

A set name included for the SPLITVS option can not be specified for the ONVS option. However, ONVS and SPLITVS can be both used in the same STORE command with different volume set names. The ONVS option also provides the ability to restrict, or enhance the creation of directory information on the store tape. If the DIRECTORY option is specified in conjunction with the ONVS option, only those accounting structures on the specified volume sets are stored.

Up to twenty volume sets may be specified.

volumesetname A volume set name specified for the ONVS option. This volume set may be a split volume set. However, the files will be stored from the user volumes, not the backup volumes. If the files are in use for writing, they will not be stored.

SPLITVS "Split volume set." Specifies that only files in the *filesetlist* that reside on the backup volumes belonging to the specified split volume set are to be stored. The files may be concurrently in use while they are being stored, since users can only access files on the user volumes.

The following example stores the files on a split volume set called, SPLIT_SET_A:

```
:STORE @.@.@; *TAPE; SPLITVS=SPLIT_SET_A
```

A set name included for the ONVS.. option cannot be specified for the ``SPLITVS option. However, SPLITVS and ONVS can be both used in the same STORE command with different volume set names. The SPLITVS option also provides the ability to restrict, or enhance the creation of directory information on the store tape. If the DIRECTORY option is specified in conjunction with the SPLITVS option, only the accounting structures on the specified split volume set are stored.

Up to twenty volume sets may be specified.

split_setname A split volume set name specified for the SPLITVS option. This volume set must be a mirrored volume set which was split through VSCLOSE; SPLIT.

RENAME Renames the file, group, account, and optionally, specifies a new creator for each entry in a fileset. STORE will rename the files while creating the "file candidate list", which is a list of files created by examination of the fileset parameter of the STORE command.

The targetname syntax is used to specify the new target name for the fileset. For more details on the use of RENAME, refer to the *Store and Turbostore Manual*.

TREE Forces each fileset to be scanned recursively. This is equivalent to using the trailing slash (/) in an HFS name. The TREE option yields a recursive scan in the hierarchical directory. This option is mutually exclusive with the NOTREE option.

NOTREE Forces each HFS syntax fileset to not be scanned recursively. The NOTREE option yields a horizontal cut in the hierarchical directory. The NOTREE option is mutually exclusive with TREE.

STOREDIRECTORY

Specifies that STORE should create a disc file that contains the backup media label and directory information. This file will be placed in the store_dirs directory of the HPSTORE.SYS group (/SYS/HPSTORE/store_dirs/). If this path does not exist, the directory file will not be created. The disc directory file can help to speed up the recovery process, particularly if ONLINE=END was used to create the backup. Because of this, this option is automatically enabled if ONLINE=END is specified.

All disc directory files are created with a file name that uniquely identifies the backup. The format is:

:**/SYS/HPSTORE/store_dirs/store_yyyymmdd_hhmmsstt_pin##_day**

where *yyyymmdd* represents the day the backup was started, *hhmmssstt* represents the time the backup was started, *pin##* is the pin number of the process that created the backup, and day is a three letter abbreviation of the day of the week the backup was started.

storedirname

If specified, a symbolic link will be created with the filename specified. This link will point to the disc directory file created in /SYS/HPSTORE/store_dirs. This allows the user to associate a more meaningful name to the disc directory file. The name can be specified in either MPE or HFS format. If it is not fully qualified, it will be fully qualified using the CWD. If the disc directory file could not be created, then then symbolic link will also not be created.

NOSTOREDIRECTORY

Specifies that STORE should not create a disc file containing the backup directory. This is the default unless ONLINE=END is specified. Use this option with ONLINE=END to prevent STORE from creating the disc directory file.

PART[IAL]DB

Allows RESTORE to restore individual database dataset files without specifying the database's root or DBCon file.

Database corruption may result if not all database files are restored from a backup. Be sure that you only want to restore certain database files before overriding the default behavior with ;PARTIALDB.

STATISTICS

Displays extra statistics about the backup. These include: amount of data written to each piece of media in each parallel set, amount of time required for each piece of media, throughput for each piece of media, and retries for each piece of media. If software compression is used, then the amount of compressed data and the compression ratio for each media is displayed. If an online backup is performed, the amount of log data written is displayed.

INTER

Specifies that file interleaving is to be used, which provides a higher disk data rate. Interleaving is accomplished by reading from several disk drives (files) simultaneously. The file data is blocked together and then stored to the specified device(s). The effect is to accelerate the store process.

INTER cannot be used with the TRANSPORT option.

STORESET Specifies parallel and sequential backup devices. This option cannot be used if the *storefile* parameter is specified, and it cannot be used in conjunction with the TRANSPORT option.

Sequential tapes are specified in this way

```
;STORESET = (*tape1,*tape2,*tape3,...)
```

This instructs STORE to use only one drive at a time from the specified serial pool for the store operation. It will select **tape1* first. When the first reel of tape is exhausted, STORE will shift to the next drive specified (**tape2*), leaving the first free for rewinding and changing reels. Thus, at any given time, only one drive is occupied with the store process. The effect is to accelerate the process by eliminating the wait for a rewind and reel switch to occur. When STORE has written to the last device specified, it will wrap around to the first device.

To specify parallel devices, enter:

```
;STORESET=(*tape1),(*tape2),(*tape3) . . .
```

In this example, all three tapes will be used in parallel during the Store.

You can also specify that a set of tapes be stored in parallel. In the following example, two tapes would be storing at any particular moment, while the other two are rewinding, which permits the operator to switch reels.

```
;STORESET=(*tape1,*tape2),(*tape3,*tape4)
```

device Specifies the device on which the file is to be stored. It must be magnetic tape or DDS. This device should be specified in a file equation before you invoke the STORE command, for example:

```
FILE DEVICE;DEV=TAPE
```

This file equation can also specify a remote device. If you are using the TurboSTORE/iX 7X24 True-Online Backup product, then a disk file can also be specified here. However, disk files can only be used with parallel STORE sets, not serial STORE sets.

STORESET cannot be used in conjunction with TRANSPORT.

THE FOLLOWING OPTIONS ARE AVAILABLE ONLY IF TURBOSTORE XL OR TURBOSTORE XL II IS INSTALLED ON YOUR SYSTEM. TURBOSTORE IS NOT PART OF THE FUNDAMENTAL OPERATING SYSTEM, BUT MAY BE PURCHASED SEPARATELY.

For additional information on TURBOSTORE XL, refer to the *STORE and TurboSTORE/iX Manual* (30319-90001).

COMPRESS Specifies that host data compression is to be used during the store operation. Currently, two levels of data compression are supported in backup. If you do not specify a level, the default is HIGH.

compressionparm list
Informs STORE what type of compression is to be done. HIGH and LOW are the only valid parameters. HIGH and LOW cannot be used together.

HIGH
Specifies that the higher of the two available data compression algorithms is to be used. Although the data will be compressed more, STORE will use more CPU resources.

LOW
Specifies that the lower of the two available data compression algorithms is to be used. Although the files will not compress as well as with HIGH, STORE will use less CPU resources.

MOSET
Specifies parallel Magneto Optical (MO) backup devices. This option is not available if the *storefile* or TRANSPORT options are specified.

Parallel devices are specified by:

```
;MOSET = (12),(13),(15)
```

or

```
;MOSET = (MO),(MO),(MO)
```

All MO devices would be used in parallel during the store process.

NAME
If this parameter is present then the specified name and ensuing options are applied to the backup media. The NAME parameter is only valid for MO backup devices. It specifies the logical name for the backup. For example:

```
:STORE @.@.@;;MOSET=(12);NAME=BK1200PM.D23OCT90.BOZO
```

This name could indicate that a backup was created on 23 Oct 1990 at 12:00 PM on the system called BOZO. If the name parameter is not specified, a similar default name will be generated by STORE based on the other backup options. In either case the backup name is displayed on the SYSLIST/OFFLINE listing as:

```
THE BACKUP TO DASS NAME IS backupname
```

It is recommended that users provide CI variables and scripts to generate their own unique NAMEs for system backups.

backupname
A three field name of a total maximum length of 26 characters. The format is *fname.gname.aname*. The name represents the "handle" to this particular backup and can be used on a subsequent restore to retrieve files from this backup. The fname, gname and aname can be up to 8 alphanumeric characters. For example:

```
DAILY.D24OCT90.SYSTEM
```

THE FOLLOWING OPTIONS ARE AVAILABLE ONLY IF TURBOSTORE/iX 7x24 TRUE-ONLINE BACKUP IS INSTALLED ON YOUR SYSTEM. TURBOSTORE/iX 7x24 IS NOT PART OF THE FUNDAMENTAL OPERATING SYSTEM, BUT MAY BE PURCHASED SEPARATELY.

`ONLINE` Online backup. The store fileset is attached to a log handler and the users can concurrently read, write or purge files in the fileset after the files are attached to the log environment. The files must not be open for write before STORE is invoked, but write access is allowed as soon as the tape mount request appears on the console. The following message indicating completion of the attach phase is also sent to the system console:

```
FILES LOCKED BY ONLINE STORE ARE NOW FREE FOR
READ/WRITE/PURGE
```

See the `NOTIFY` option for an additional way to notify users that the attach phase has completed.

`START` Specifies that a 7x24 true-online sync point should occur at the beginning of the backup, before any files are stored. All files being stored do NOT have to be closed for write access when the backup starts.

`END` Specifies that a 7x24 true-online sync point should occur at the end of the backup, after all files are stored. All files being stored do NOT have to be closed for write access at any time during the backup.

Specifying the option causes file log data to be written at the end of the backup. This media format is NOT backwards compatible, and media created with `ONLINE=END` CANNOT be verfied or restored on a pre-5.5 system.

time Specifies when the true-online sync point should occur, in in 24-hour format, as `HH:MM:SS`.

The *time* must be specified with either `START` or `END`. If specified with `START`, the sync point will occur at the time specified, or after all of the files being stored are attached to shadow log files, whichever happens last. If specified with `END`, the sync point will occur at the time specified, or once all files have been stored, whichever happens last.

If the time specified is before the time the backup is started, then `STORE` will wait until that time the following day. This is helpful if you start the backup at 11:00 PM and want the sync point to occur at 2:00 AM the next morning.

`ASK` When specified, will cause TurboSTORE to pause with an operator request before the true-online sync point. If you reply "N" to this request, you will be given the option of aborting the backup or continuing to wait.

After you reply to the console request, the sync point will occur.

This option can be specified with *time*, and must be specified with either `START` or `END`.

volumesetname The name of the volume set where the shadow log files should reside, which must be a valid, currently mounted volume set.

Operation Notes

- **Usage**

  The STORE command stores one or more disk files onto magnetic tape DDS or MO disc. It will store only those files whose home volume set(s) is (are) mounted.

- **Required capabilities for storing files**

  If you have system manager (SM) or system supervisor (OP) capability, you can store any file in the system. If you have account manager (AM) capability, you can store any file in your account, but you cannot store files having negative file codes unless you have Privileged Mode (PM) capability.

  Before entering a STORE command, you must identify storefile as a magnetic tape or DDS device by using the FILE command (creating a file equation).

- **Invoking the STORE functionality**

  You may invoke the STORE functionality with the RUN command (for example, RUN STORE.PUB.SYS). 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 STORE: prompt will appear. Acceptable responses are a complete STORE command, a complete RESTORE command, or a complete VSTORE command.

  If you have purchased a Turbostore product, it will be installed as TSTORE.PUB.SYS. As long as a non-zero length TSTORE program exists in PUB.SYS, typing any CI STORE, RESTORE, or VSTORE command will invoke Turbostore instead.

- **Performing 7X24 True-Online Backups**

  All databases being stored will be quiesced at the sync point. This means that all current transactions will be allowed to complete, and no new transactions can begin. Once STORE has captured a logically consistent copy of the database(s) begin stored, all databases will be unquiesced. The amount of time between quiesce and unquiesce depends on how may databases are being stored. It will generally be very short (less than a minute). Currently only TurboIMAGE and ALLBASE/SQL databases are quiesced.

  Just before the sync point starts, the following message will be sent to the console:

  ```
  ONLINE BACKUP SYNC POINT STARTING
  ```

  After this message is displayed, all TurboIMAGE and ALLBASE/SQL databases being stored will be quiesced and then unquiesced. Once the sync point has completed, the following message will be sent to the console:

  ```
  ONLINE BACKUP SYNC POINT FINISHED
  ```

  For more information on scheduling, managing, and performing 7x24 True-Online backups, consult the *Store and TurboSTORE/iX Manual* (30319-90001).

Use

If you press [Break] during a STORE operation, the operation continues while you interact with the Command Interpreter. Both ABORT and RESUME can be used within BREAK.

This command may be issued from session, job, or program, but not in BREAK. The user must have Privileged Mode (PM) capability to execute this command for privileged files.

Examples

To store all files on the system (including HFS files), enter

```
:STORE /
```

or

```
:STORE @.@.@
```

To store all MPE named files (and exclude HFS files and directories), enter

```
:STORE ?@.@.@
```

To store all (MPE and HFS) files in the group GP4X in your logon account to a tape file named BACKUP, enter

```
:FILE BACKUP;DEV=TAPE
:STORE @.GP4X;*BACKUP;SHOW
```

The console operator receives a request to mount the tape identified as BACKUP. A listing of the files stored appears on your standard list device.

To store all files on the system except the MPE files in the SYS account, enter

```
:FILE TAP;DEV=TAPE
:STORE @.@.@-@.@.SYS;*TAP;SHOW=SECURITY,DATES,LONG,OFFLINE
```

The console operator receives a request to mount the tape identified as TAP. A listing of the files stored appears on both standard list and at the system line printer. The listing will include all information available from STORE.

To store from indirect file INDFILE which contains

```
FILE1,FILE2;SHOW
FILE3,@.PUB.SYS;DATE>=6/1/87
```

enter:

```
:FILE T;DEV=TAPE
:STORE ^INDFILE;*T
```

The console operator receives a request to mount the tape identified as T. Files FILE1, FILE2, FILE3, and all files in PUB.SYS will be stored if they have been modified since June 1, 1987. A listing of the files stored appears on your standard list device.

To store files from a group and account with a default *storefile*, enter

```
:STORE @.GROUP.ACCOUNT
```

or

```
:STORE
```

Note that the console operator receives a request to mount the tape identified as the user's user name.

To store files from a group and account and to purge them after the STORE, enter

```
:FILE T;DEV=TAPE
:STORE @.GROUP.ACCOUNT;*T;PURGE
```

Related Information

Commands RESTORE, VSTORE, REPLY, RECALL

Manuals | *STORE and TurboSTORE/iX Manual*

Magneto-Optical Media Manager User's Guide

Volume Management Reference Manual

Mirrored Disk/iX User's Guide

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.

The time-related parameters are ignored when the STREAM command is applied to the DATA command, however.

Syntax

```
STREAM[ filename][ ,char]
[ ;AT=timespec] [ ;DAY={ day-of-week day-of-month days-until-month}]
[ ;DATE=datespec] [ ;IN=[ days[ ,[ hours] [ ,minutes] ] ] ]
[JOBQ=queuename]
```

Parameters

filename The Editor (ASCII) 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/iX commands. The user must have READ and LOCK access or EXECUTE access.

queuename The name of the queue into which the job must logon. If no queuename is specified the default system job queue will be used. If queuename is specified it takes precedence over a job queuename in the JOB statement of the file being streamed.

char Character used in place of colon (:) to identify MPE/iX commands within the input file. When the input file is entered on a device configured to accept jobs or sessions, this character can be any ASCII special (nonalphanumeric) 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<=24) and *MM* is the minutes of the hour (0<=MM<=60).

If DAY and DATE are not specified, then:

```
timespec < NOW-> JOB LOGON TOMORROW
timespec > NOW-> JOB LOGON TODAY
timespec = NOW-> JOB LOGON IMMEDIATELY
     WITH 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 *day-of-month* is greater than or equal to the current *day-of-month*, the current month is indicated. If *day-of-month* 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 (for example, if 31 is entered for February). If *day-of-month* is omitted, the current date is used.

days-until- month Days until the end of the 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 is scheduled for the eighth day from the end of the next month.

`DATE` Absolute date specification.

datespec 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 number of days from the current date.

hours Hours. A positive integer (`0<`*hours*`<=23`) indicating the number of hours from the current time. If omitted, zero is used.

minutes Minutes. A positive integer (`0<=`*minutes*`<=59`) indicating the number of minutes from the current time. If omitted, zero is used.

Operation Notes

The `STREAM` command allows you to initiate jobs while in an interactive session by constructing your job from your terminal or by reading records from a disk or tape file. When the job is read, MPE/iX spools it onto a disk file, assigns it a job number, and processes it independently as an entity completely separate from your session. In the meantime, MPE/iX allows you to continue with your session. You can specify the queue name into which a particular job should go. The name specified overrides the queue name specified in the JOB command.

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/iX STREAM facility by entering the `STREAMS` console command. The `STREAMS` console command also specifies a streaming device, which to MPE/iX 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 (not necessarily your terminal) displays the job number assigned by MPE/iX and the listing generated by the job.

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/iX prompts you for input by displaying a greater than (>) character. When you enter STREAM for a device other than your terminal, MPE/iX 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 all commands with an appropriate substitute (other than colon) character, as in !JOB. When the input file is spooled to a disk, MPE/iX 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/iX assigns each job a unique job number (JobID). MPE/iX also assigns each job a preset priority, unless you specify otherwise in the JOB command, and processes the job independently of the initiating job or session. 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 logs on immediately. If the time specified is earlier than the current time, and DAY and DATE are not specified, a warning message is generated, and the job is scheduled for the specified time tomorrow. 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 are scheduled at the same time.

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 is executed when system variables allow.

Terminating Streamed Jobs

To terminate interactive job input, enter a colon (:). In response, MPE XL ceases prompting for batch job input and instead prompts you for another MPE/iX command:

```
>:  ** Denotes end of batch job input **

:  ** MPE XL prompts for next command **
```

Pressing **Break** aborts the execution of this command and any job currently being entered through the command. Incompletely spooled disk space is returned to the system.

If you make an error while entering the MPE/iX JOB command, you 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 *Introduction to MPE XL for MPE V System Administrators* (30367-90003) for information on using the ABORTJOB command to terminate time-scheduled jobs.

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.

NOTE Scheduled jobs survive a START RECOVERY. Any other type of system startup causes scheduled jobs to be deleted. If a job is scheduled for introduction earlier than the system startup, the job enters the WAIT state and executes when the system parameters allow it to execute.

If the system is brought down for any reason, first execute a SHOWJOB command to show the scheduled jobs. Then reschedule the jobs when the system is brought back up on anything other than a START RECOVERY.

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 does not enter the WAIT state at midnight on the specified day. Instead, it uses the time that the STREAM is executed, and enters the WAIT state at that time on the specified day.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command and any partially streamed job.

Examples

To stream a job from a disk 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 file name (the first parameter). In the following example, an asterisk (*) is used as a substitute command identifier:

```
 STREAM ,*
>*JOB USER.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
!JOB WXYZ,WRITER.TEC
!EDITOR
TEXT ABC
n
EXIT
!EOJ
#J87

:
```

In the preceding example, the job input file is `EXAMPLE` which initiates the job `WXYZ`. `WXYZ` invokes the editor subsystem where the file `ABC` is referenced. The `EOJ` command terminates the job and `#J87` is the job number assigned by MPE/iX.

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/iX `JOB` command. Enter:

```
 STREAM
>!JOB USER.TECHPUBS;OUTCLASS=12
```

The following section contains additional examples of using the `STREAM` command. For these examples, assume that the current date and time are Monday, June 8, 1987, 12:00 p.m. Also assume the job file 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, June 9.

`STREAM JOBFILE; AT=20:00` `JOBFILE` will be introduced at 8:00 p.m., Monday, June 8.

`STREAM JOBFILE; IN=,8` `JOBFILE` will be introduced in eight hours, at 8:00 p.m., Monday, June 8.

`STREAM JOBFILE; IN=1,8` `JOBFILE` will be introduced in one day plus eight hours, at 8:00 p.m., Tuesday, June 9.

`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, June 15.

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, June 8.

STREAM JOBFILE; DAY=9; AT=20:00
Since the day of the month (9) is later than the current day of the month (8), the current month is assumed. JOBFILE will be introduced on Tuesday, June 9, at 8:00 p.m.

STREAM JOBFILE; DAY=5 Since the day of the month (5) is earlier than the current day (8), the next month is assumed. Since no time was specified, JOBFILE will be introduced on Saturday, July 5, at 12:00 p.m.

STREAM JOBFILE;
DAY=31 Since there is no June 31, the next month is assumed. Since there is a July 31, this is a legal command. JOBFILE will be introduced on Friday, July 31, at 12:00 p.m. If there were no July 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, June 29, at 12:00 p.m.

STREAM JOBFILE; DAY=-25 The -25 means the twenty-fifth day from the end of the month. If one assumes the current month, that implies June 6, but June 6 is earlier than the current day; therefore, the next month is assumed. JOBFILE will be introduced on Sunday, July 7, at 12:00 p.m.

STREAM JOBFILE; DATE=6/8/87;
AT=8:00 Since the specified time is earlier than the current time, this command is not legal and results in an error.

STREAM JOBFILE; DATE=6/8/87;
AT=20:00 The specified time is later than the current time, so this command is legal. JOBFILE will be introduced on Monday, June 8, at 8:00 p.m.

Related Information

Commands JOB, STREAMS, SHOWJOB, LISTJOBQ

Manuals *Performing System Operation Tasks*

STREAMS

Enables or disables the STREAMS device. Allows or disallows users to submit job/data streams.

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 SYSGEN dialog.

`OFF` Disables the STREAMS facility.

Operation Notes

The operator executes this command after a startup to enable the STREAM facility. The STREAMS device must be enabled each time the system is brought back online in order to allow users to stream jobs. (Streamed jobs are processed separately by MPE/iX, 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/iX processes 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, physically connected to the computer. If this is the case, then the STREAMS device is considered a "pseudo-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`.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console unless distributed to users with the `ALLOW` command.

Examples

To enable jobs and data streams on logical device number 10, enter:

```
STREAMS 10
```

To disable data streams, enter:

```
STREAMS OFF
```

Related Information

Commands	`STREAM`, `SHOWDEV`
Manuals	*Performing System Operation Tasks*

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 spool file and then stop.

Operation Notes

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 spool files listed will be READY for printing; none are ACTIVE, and a SHOWDEV of the spooled device indicates that the device is still spooled. Refer to the SHOWOUT command in this manual.

When suspending an ACTIVE spool file, 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/iX returns a colon prompt (:). The command is not executed, however, until the output device is returned online. Only then do you receive the SPOOLER SUSPENDED message.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console unless distributed to users with the ALLOW or ASSOCIATE command.

Examples

To suspend printing on logical device 6, enter:

```
SUSPENDSPOOL 6
```

To suspend printing on logical device 6 once the currently active spool file is completely printed, enter:

```
SUSPENDSPOOL 6;FINISH
```

Related Information

Commands	RESUMESPOOL, SHOWOUT, SHOWDEV

Manuals	*Performing System Operation Tasks*

SWITCHLOG

Closes the current system log file, then creates and opens a new one. (Native Mode)

Syntax

SWITCHLOG

Parameters

None.

Operation Notes

When the SWITCHLOG command is executed, MPE/iX displays the previous system log file number (*xxx*), the percentage of file space used (*yy*), and the current open log file (*zzz*), as shown in the following example:

```
SYSTEM LOG FILE #xxx IS yy% FULL
SYSTEM LOG FILE #zzz IS ON
```

If this command is issued and logging is not active the following message is displayed:

```
NO LOGGING
LOG FILE xxx IS yy% FULL
```

NOTE Do not create new log files with the BUILD command since MPE/iX creates them automatically. If you use the BUILD command to create a new log file and then attempt to switch the current log file to the file you created, user logging suspends in an error state and the following message is displayed:

```
SYSTEM LOG FILE #xxx ENCOUNTERED ERROR #nnn
LOGGING SUSPENDED.
```

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. System supervisor (OP) capability is required to use this command.

Example

To switch logging to a new log file, enter:

```
SWITCHLOG
```

Related Information

Commands	CHANGELOG, RESUMELOG
Manuals	*SPU Switchover/XL User's Guide*

SYSGEN

Starts configuration dialog and/or installation tape creation. The equivalent compatibility mode command is SYSDUMP. (Native Mode)

Syntax

SYSGEN[*basegroup*] [*,newgroup*] [*,inputfile*] [*,outputfile*]

Parameters

basegroup The name of a base configuration group in the SYS account which contains configuration data to be used as a basis for any changes made during the SYSGEN session and/or to be used for creation of the installation tape. If the name of a base group is not specified in the SYSGEN command, it defaults to the group used to bring up the system (normally CONFIG). The base configuration group given or defaulted on the SYSGEN command can be changed with the SYSGEN BASEGROUP command.

newgroup The name of a group in the SYS account which is used as the default for keeping a new set of configuration data or a copy of the configuration data in the base configuration group. If the name of a new group is not specified on the SYSGEN command, it defaults to *basegroup*. The new configuration group given or defaulted on the SYSGEN command can be overridden by specifying a group name with the SYSGEN KEEP command.

inputfile Actual file designator of the file to be used for command input during the execution of SYSGEN. The formal file designator used by the SYSGEN program for this file is SYSGIN. The default is $STDIN.

outputfile Actual file designator of the file to be used for any output requested during the configurator/user dialog. The formal file designator used by the SYSGEN program for this file is SYSGOUT. The default is $STDLIST.

Operation Notes

The SYSGEN command initiates the configurator/user interface. Once executing, SYSGEN can be used to create new system configurations, to modify existing ones, and to create installation tapes for any MPE/iX system.

System supervisor capability (OP) is required to view configuration data. System manager (SM) capability is required to make configuration changes and keep them or to create an installation tape.

To begin interaction with the MPE/iX configurator, the SYSGEN command is entered. During the interaction, system configurations can be created, modified, or used to create installation tapes.

The base for configuration changes or tape creation can be specified on the SYSGEN command with the base group. The group name to which the configuration is to be kept with a SYSGEN KEEP command can be specified on the SYSGEN command line with the *newgroup* parameter.

Input for the configurator interaction can be redirected from a file with the SYSGEN command *inputfile* parameter. Any output during the interaction can be redirected to a file with the SYSGEN command *outputfile* parameter. In addition, input and output can be redirected with file equations using the formal designators SYSGIN and SYSGOUT, respectively, prior to entering the SYSGEN command.

Use

This command is available in a session and programmatically. It is not available from a job. Pressing **Break** suspends the execution of this command. Entering the RESUME command continues the execution.

Examples

The following four examples perform the same action. Each causes the group CONFIG.SYS to be used as the basis for configuration data, the group NEWCONF.SYS to be used for any KEEP command without a group specification, the file $STDIN to be used for input and the file $STDLIST to be used for output.

```
SYSGEN CONFIG,NEWCONF,$STDIN,$STDLIST

SYSGEN CONFIG,NEWCONF

SYSGEN ,NEWCONF

FILE SYSGIN=$STDIN

FILE SYSGOUT=$STDLIST

SYSGEN ,NEWCONF
```

Related Information

Commands	NMMGR, VOLUTIL
Manuals	*System Startup, Configuration, and Shutdown Reference Manual*
	Performing System Management Tasks

TELL

Sends a message to another session.

Syntax

TELL{ [#] S*nnn* [*sessionname*,] *username.acctname* @ @.*acctname* @S } [[;] *text*]

Parameters

[#]S*nnn* The session number as assigned by MPE/iX. This session number receives the TELL message.

[*sessionname*] *username. acctname* 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 command lists all the *username.acctnames* to which you may direct a TELL message. Sessions with an active SETMSG OFF command are listed as being in QUIET mode and do not receive your TELL message. This is also true for a session on the system console. If several users are running under the same session identity, MPE/iX sends 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 semicolon (;) and consisting of any string of ASCII characters. The default is that no text is printed; however, MPE/iX still prints the FROM message as follows:

FROM/*sessionid*

Operation Notes

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/iX queues the message as high as possible among the current input/output requests but does not interrupt any read or write 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/iX blocks the TELL command if the receiving device is operating in the QUIET mode (refer to the SETMSG command) and informs the sender with:

S*nnn* *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:

```
TARGET MUST BE INTERACTIVE, NO MESSAGE SENT.
(CIWARN 1627).
```

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

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
```

Related Information

Commands	TELLOP, WARN
Manuals	*Performing System Operation Tasks*

TELLOP

Sends a message to the system console. (Native Mode)

Syntax

`TELLOP`[*text*]

Parameters

text Message text, preceded by a space and consisting of any string of ASCII characters. Default is that no text is printed; however, MPE/iX still prints the `FROM` as follows:

`FROM/`*sessionid*

Operation Notes

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 or if an active session is running in QUIET mode.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command.

Example

To ask the system operator to mount a tape, enter:

```
TELLOP PLS MOUNT MYTAPE,VERSION 1
```

Related Information

Commands	`TELL`, `WARN`
Manuals	None

TUNE

Changes scheduling characteristics of the scheduling subqueues. These characteristics include base and limit priorities, quantum bounds (min and max), boost property and timeslice. (Native Mode)

Syntax

```
TUNE[ minclockcycle] {   ;CQ=qinfo   ;DQ=qinfo   ;EQ=qinfo }   [ ... ]
```

Where *qinfo* is written in the following form:

```
[  base  [   ,  [ limit]  [   ,  [ min]  [   ,  [ max]  [   ,DECAY   ,OSCILLATE ]    ,
[ tslice] ] ] ] ]
```

NOTE Misuse of this command can significantly degrade system operating efficiency.

PARAMETERS

minclockcycle This parameter is ignored. It appears here for MPE V/E compatibility only.

base An integer from 150 to 255 specifying the priority at which user processes executing in the CS, DS, and ES scheduling subqueues begin their Dispatcher transactions. Priority is inversely related to the integer: a higher-priority process has a lower number. While the full range is provided for compatibility, avoid setting the base priority between 150 and 152, since user processes running at priorities greater than 152 can adversely affect system performance.

limit An integer specifying the lowest priority at which a process in the CS, DS, or ES scheduling subqueues can execute. Priority is inversely related to the integer: a higher-priority process has a lower number. The *limit*, which can range from 150 to 255, must be greater than or equal to the *base*.

min The minimum quantum is a lower bound for the dynamically calculated quantum (average transaction time) value. The quantum value determines the rate of priority decay for processes within the scheduling subqueue. Values range between 1 and 32767 milliseconds.

max The maximum quantum is an upper bound for the dynamically calculated quantum (average transaction time) value. The quantum value determines the rate of priority decay for processes within the scheduling subqueue. Values range between 1 and 32767 milliseconds. The value of *max* must be greater than or equal to the value of *min*.

DECAY Sets the subqueue to the default decay behavior associated with circular scheduling subqueues. If set, a process decays normally to the *limit* priority and returns to the *base* priority when the Dispatcher transaction is complete. DECAY is the default boost property.

`OSCILLATE` Sets the subqueue to oscillate behavior. If set, a process returns to the *base* priority once its priority has decayed to the *limit* of the subqueue, even if it has not completed a Dispatcher transaction.

tslice The number of milliseconds a process in a given subqueue can hold the CPU. A process that has held the CPU continuously for this number of milliseconds is interrupted. This value must be set to a multiple of 100 milliseconds and has a minimum value of 100 milliseconds.

OPERATION

The system manager uses the `TUNE` command to change the characteristics of the circular scheduling subqueues to more efficiently manage the current processing load.

A process in the CS, DS, or ES scheduling subqueues typically begin execution at the *base* priority. When the process stops (for disk I/O, terminal I/O, preemption, etc.), the amount of CPU it has consumed is used to determine its new priority. If the process has completed a Dispatcher transaction, typically by issuing a terminal read, its priority is reset to the *base*, and the quantum value for that workgroup is recalculated. If the process has exceeded the quantum (filter) value since its priority was last reduced, the priority is decreased without exceeding the *limit* priority. If the boost property for the workgroup is oscillate, process priorities are reset to the *base* value once they decay to the *limit*.

The parameters *min* and *max* refer to the absolute bounds of the quantum, or a filter representing the average transaction time of processes in that subqueue. The quantum is recomputed after every user Dispatcher transaction is complete, and then compared against the CPU time of a process to determine whether the priority of the process should be decreased.

NOTE With Release 5.0 of MPE/iX, all three circular scheduling subqueues, CS, DS, and ES, have dynamically calculated quantums. By default, the DS and ES subqueues have their bounds set to the same value.

If the values specified for *max* are too large, system response may become erratic. If they are too small, excessive memory management may occur due to frequent process swapping. Either case degrades system performance. The values for *min* and *max* may range from 1 to 32,767. The recommended settings are listed in the table below.

The timeslice value determines how long a process in a given scheduling subqueue will be allowed to hold the CPU. This value is different than the quantum, which determines how rapidly process priorities decay. The timeslice does interrupt the process if the process is interruptable. The timeslice is a multiple of 100 milliseconds and has a minimum value of 100 milliseconds.

The following default settings are established when the system is booted from the system disk (a `START RECOVERY` or `START NORECOVERY`), unless the user has customized a `TUNE` configuration.

```
START RECOVERY or START NORECOVERY

CQ base: 152  DQ base: 202  EQ base: 240
  limit: 200    limit: 238    limit: 253
     min:  1     min: 2000     min: 2000
     max: 2000   max: 2000      max: 2000
  boost: DECAY  boost: DECAY  boost: DECAY
    tslice: 200  tslice: 200     tslice: 200
```

NOTE The MPE/iX Scheduler now supports the workgroup concept. However, backward compatibility is maintained through five default workgroups created by the system. The scheduling characteristics of the CS_Default, DS_Default, and ES_Default workgroups mimic those of the CS, DS, and ES scheduling subqueues. In fact, changing the scheduling characteristics of the CS, DS, and ES scheduling subqueues, via the TUNE command, is equivalent to changing the characteristics of the corresponding default workgroup through ALTWG. Please refer to the NEWWG and ALTWG commands for more detail.

Workload Manager users should use ALTWG rather than TUNE since TUNE does not modify user-defined workgroups. If you aren't using Workload Manager, and you want to change one of the system-defined workgroups, you may wish to use ALTWG because it only examines member processes of a specific workgroup and not all processes on the system.

The TUNE command may be issued from a session, job, program or in BREAK. Pressing **Break** has no effect on this command. TUNE requires System Supervisor (OP) or System Manager (SM) capability.

EXAMPLE

To set the CS subqueue's *base* to 152, *limit* to 200, and *max* quantum (filter) to 300; and the DS subqueue's *base* to 202, *limit* to 238, *min* and *max* quantum (filter) to 1000, and cause oscillation boosting, enter:

```
TUNE CQ=152,200,300,300;DQ=202,238,1000,1000,OSCILLATE
```

To set the CS subqueue to oscillation with a 300 millisecond timeslice and the DS subqueue's *base* to 180, *limit* to 238, boost property to decay, and timeslice to 1500, enter:

```
TUNE CQ=,,,,OSCILLATE,300;DQ=180,238,,,DECAY,1500
```

Related Information

Commands	SHOWQ, ALTPROC, SHOWPROC, NEWWG, ALTWG, PURGEWG, SHOWWG
Manuals	*MPE/iX Intrinsics Reference Manual*

UP

Returns a particular device to its normal function on the system; cancels any DOWN command issued for the device. This command does not apply to disk drives.

Syntax

UP *ldev*

Parameters

ldev The logical device number of the device being returned to service online.

Operation Notes

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 retains ownership even after the UP command is executed.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console unless distributed to users with the ALLOW or ASSOCIATE command.

Example

To allow logical device number 10 to function again, enter:

```
UP 10
SHOWDEV 10
LDEV AVAIL OWNERSHIP VOLID ASSOCIATION

10 A AVAIL
```

Related Information

Commands DOWN, SHOWDEV

Manuals *Performing System Operation Tasks*

VMOUNT

Enables or disables the MPE/iX movable volume facility. (Native Mode)

Syntax

```
VMOUNT{  ON [ ,AUTO]  OFF } [ ;ALL,]
```

Parameters

ON or ON,AUTO Enables the movable volume facility so that all valid user MOUNT/VSRESERVE and operator LMOUNT/VSRESERVESYS requests are allowed. When ON is used without AUTO, the operator must reply to all MOUNT/VSRESERVE requests.

When ON,AUTO is used, MPE/iX attempts to satisfy user MOUNT/VSRESERVE and operator LMOUNT/VSRESERVESYS requests without operator intervention.

OFF Requests to use the movable volume facility are rejected.

ALL Prints all volume set mount-related console messages, including those not requiring operator intervention, on the console.

Operation Notes

If the movable volume facility is enabled when you issue a VMOUNT OFF command, users having reserved volume sets are unaffected; the command is satisfied when the last access is complete.

The MPE/iX naming convention for volume sets differs from that of MPE V/E for private volumes. Refer to the MOUNT, DISMOUNT, VSRESERVE, and VSRELEASE commands in this chapter.

Once the movable volume facility has been enabled, use the VSUSER command to determine which users have which volume sets reserved. Refer to the VSUSER command in this chapter.

The movable volume facility is enabled immediately following a system startup. (The setting is equivalent to VMOUNT ON,AUTO.) However, you still receive console messages concerning volume set requests.

The operator has the greatest interactive control over the use of volume sets by using VMOUNT ON;ALL. The command that least interrupts the operator when users are accessing volume sets is VMOUNT ON,AUTO.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console unless distributed to users with the ALLOW command.

Examples

To disable the movable volume facility so that no messages are sent to the console when users attempt to reserve volume sets (the default condition) enter:

```
VMOUNT OFF
```

To disable the movable volume facility and still receive messages on the console when users attempt to reserve volume sets, enter:

```
VMOUNT OFF;ALL
```

Related Information

Commands	`VSUSER`, `DISMOUNT`
Manuals	*Volume Management Reference Manual*

VSCLOSE

Informs the system to close the specified volume set and take it offline. (Native Mode)

Syntax

```
VSCLOSE volumesetname[ [ ;PARTVS=] {   USER   BACKUP } ] [   ;NOW   ;SPLIT ]
```

Parameters

volume- setname The volume set that is to be closed. Any user who is accessing a file at the time this command is issued is allowed to finish accessing the file. However, users who are not accessing files are unable to open files on the volume set, and VSRESERVE and MOUNT requests are denied. Refer to "Operation Notes," below.

PARTVS This option is available only with the Mirrored Disk/XL, a separately purchased product. For information, refer to !Mirrored Disk User's Guide> *Mirrored Disk/iX User's Guide* (30349-90003). This parameter only applies to a previously split volume set. Specify it when you want only half of split volume set to be closed.

USER	Close only the user volumes.
BACKUP	Close only the backup volumes.

If PARTVS is not specified, both volume set halves are closed. If PARTVS is specified for a nonsplit volume set, an error is returned and the volume set is not closed.

NOW Instructs the system to abort any job or session that is using any file that resides in the specified volume set. However, if a VSRESERVESYS or an LMOUNT command has already been issued for the specified volume set, then the operator should execute a VSRELEASESYS command, followed by a VSCLOSE ;NOW command, in order to take the volume set offline.

The NOW parameter permits the operator to remove a volume set without having to use VSUSER and then perform an ABORTJOB on the users of the volume set. This command may be issued only from the system console.

SPLIT This option is available only with the Mirrored Disk/iX, a separately purchased product. For information, refer to *Mirrored Disk/iX User's Guide* (30349-90003). It splits the volume set into user volumes and backup volumes if it is a mirrored volume set and if it is in the proper state.

The SPLIT option can*not* be used with the NOW option. All members of the volume set and both members of each pair must be present. There can be no repair taking place. Both members of each volume pair must be identical at the time of the split. There can be no users logged onto the volume set when the split is processed.

For each mirrored pair, the system assigns a backup volume and user volume. An attempt is made to place the backup volumes and user volumes on separate hardware channels. The volume with the greatest path number is selected as the backup volume.

If SPLIT is specified for a nonmirrored volume set, an error is returned and the volume set is not closed.

Operation Notes

This command notifies the system to close the volume set and take it offline. This is done when all users have ceased using files on the volume set, and when any program file that has been allocated on the volume set has been deallocated (via the DEALLOCATE command). Once the VSCLOSE command is issued for a volume set, individual users can no longer issue VSRESERVE or MOUNT commands for the volume set.

Specifying the NOW parameter permits the operator to take the volume set offline immediately, unless a VSRESERVESYS or an LMOUNT command has been issued, or unless a program file has been allocated on the volume set.

This command restricts access to the volume set. Jobs or sessions are granted access to the volume set only if they have at least one open file on the volume set or if they have already issued an explicit VSRESERVE or a MOUNT command for the volume set.

The MPE/iX naming convention for volume sets differs from that of MPE V/E for private volumes.

In MPE V/E, the name A.B.C indicates that B is the name of a group and that C is the name of an account. MPE/iX accepts that name, but no interpretation is made as to the referencing of B and C. Instead, MPE/iX treats A.B.C as a single, long string name. It is the flexibility of the MPE/iX naming convention that makes it possible for MPE/iX to work with a volume set designated A.B.C.

MPE/iX volume set names may consist of any combination of alphanumeric characters, including the underbar (_) and the period (.). The name must begin with an alphabetic character and must consist of no more than 32 characters.

A volume set called MY_OWN_PERSONAL_VOLUME_SET is acceptable in MPE/iX, and so is MY.OWN.PERSONAL.VOLUME.SET; similarly, A.B.C is acceptable. If a volume set is named according to the MPE V/E naming convention (A.B.C), you must use an unambiguous reference when using the MPE/iX volume set commands, such as:

```
 Vcommand A.B.C
```

Entering Vcommand A fails to access the volume set. You cannot specify the first part of the volume set name alone and expect the group and account to default.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. This command may be issued only from the system console unless distributed to other users with the ALLOW command.

Examples

To close the volume set ACCOUNTING_PAYROLL, enter:

```
VSCLOSE ACCOUNTING_PAYROLL
```

However, if a VSRESERVESYS command has been issued for ACCOUNTING_PAYROLL, then a message is displayed on the console. In order to close this volume set and take it offline, the operator has to issue these commands:

```
VSRELEASESYS ACCOUNTING_PAYROLL
VSCLOSE ACCOUNTING_PAYROLL
```

Related Information

Commands	The VSxxxxxx commands in this chapter, DISMOUNT
Manuals	*Volume Management Reference Manual*

VSOPEN

Reopens a volume set that has been closed with VSCLOSE. The volume set becomes available for use again. (Native Mode)

Syntax

>VSOPEN ***volumesetname***[[;PARTVS=] { USER BACKUP }]

Parameters

volume- setname The volume set to be opened. You must specify an unambiguous name. MPE/iX does not accept part of a *volumesetname* and defaults the remainder of the name. Refer to "Operation Notes."

PARTVS This option is available only with the Mirrored Disk/iX, a separately purchased product. For information, refer to *Mirrored Disk/iX User's Guide* (30349-90003). This parameter only applies to a previously split volume set. It notifies the system which split volume set half is to be opened.

USER	Open only the user volumes.
BACKUP	Open only the backup volumes.

If PARTVS is not specified, both volume set halves are opened. If PARTVS is specified for a non split volume set, an error is returned and the volume set is not opened.

Operation Notes

This command notifies the system to open the specified volume set. Because bringing a volume set online opens the set by default, this command is needed only for a volume set for which a VSCLOSE command has been issued.

The MPE/iX naming convention for volume sets differs from that of MPE V/E for private volumes. In MPE V/E, the name A.B.C indicates that B is the name of a group and that C is the name of an account. MPE/iX accepts that name, but no interpretation is made as to the referencing of B and C. Instead, MPE/iX treats A.B.C as a single, long string name. It is the flexibility of the MPE/iX naming convention that makes it possible for MPE/iX to work with a volume set designated A.B.C.

MPE/iX volume set names may consist of any combination of alphanumeric characters, including the underbar (_) and the period (.). The name must begin with an alphabetic character and consist of no more than 32 characters.

A volume set called MY_OWN_PERSONAL_VOLUME_SET is acceptable in MPE/iX, and so is MY.OWN.PERSONAL.VOLUME.SET; similarly, A.B.C is acceptable.

If a volume set is named according to the MPE V/E naming convention (A.B.C), you must use an unambiguous reference when using the MPE/iX volume set commands, such as:

```
Vcommand A.B.C
```

Entering `Vcommand A` fails to access the volume set. You cannot specify the first part of the volume set name alone and expect the group and account to default.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. This command may be issued only from the system console unless distributed to other users with the `ALLOW` command.

Examples

To open the volume set `ACCOUNTING_PAYROLL`, enter:

```
VSOPEN ACCOUNTING_PAYROLL
```

Related Information

Commands	The `VSxxxxxx` commands in this chapter, `DISMOUNT`
Manuals	*Volume Management Reference Manual*

VSRELEASE

Releases a volume set that was explicitly reserved by the user with VSRESERVE. The equivalent compatibility mode command is DISMOUNT. (Native Mode)

Syntax

VSRELEASE[*volumesetname*]

Parameters

volume- setname The volume set to be released. If you omit the parameter, the request is issued for the home volume set of the user's logon group and account. Refer to "Operation Notes."

Operation Notes

This command releases a volume set when it is no longer in use and negates a previous reservation of a volume set.

The MPE/iX naming convention for volume sets differs from that of MPE V/E for private volumes.

In MPE V/E, the name A.B.C indicates that B is the name of a group and that C is the name of an account. MPE/iX accepts that name, but no interpretation is made as to the referencing of B and C. Instead, MPE/iX treats A.B.C as a single, long string name. It is the flexibility of the MPE/iX naming convention that makes it possible for MPE/iX to work with a volume set designated A.B.C.

MPE/iX volume set names may consist of any combination of alphanumeric characters, including the underbar (_) and the period (.). The name must begin with an alphabetic character and consist of no more than 32 characters.

A volume set called MY_OWN_PERSONAL_VOLUME_SET is acceptable in MPE/iX, and so is MY.OWN.PERSONAL.VOLUME.SET; similarly, A.B.C is acceptable.

If a volume set is named according to the MPE V/E naming convention (A.B.C), you must use an unambiguous reference when using the MPE/iX volume set commands, such as:

```
 Vcommand A.B.C
```

Entering: Vcommand A fails to access the volume set. You cannot specify the first part of the volume set name alone and expect the group and account to default.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. Use volumes (UV) or create volumes (CV) capability is required to use this command.

Example

To request that volume set ACCOUNTING_PAYROLL be released, enter:

```
 VSRELEASE ACCOUNTING_PAYROLL
```

Related Information

Commands	The VSxxxxxx commands in this chapter, DISMOUNT
Manuals	*Volume Management Reference Manual*

VSRELEASESYS

Releases a specified volume set previously reserved with the VSRESERVESYS command. The equivalent compatibility mode command is LDISMOUNT. (Native Mode)

Syntax

VSRELEASESYS *volumesetname*

Parameters

volume- setname The name of the MPE/iX volume set for which a previously issued VSRESERVESYS command has been issued. Refer to "Operation Notes."

Operation Notes

This command is used to negate a previously issued VSRESERVESYS command for the specified volume set. It informs the system that the volume set is no longer reserved system-wide.

This command does not prohibit individual VSRESERVE (MOUNT) or VSRELEASE (DISMOUNT) commands issued for the specific volume set by individual users.

The MPE/iX naming convention for volume sets differs from that of MPE V/E for private volumes. Refer to the MOUNT, DISMOUNT, VSRESERVE, and VSRELEASE commands in this chapter.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. This command may be issued only from the system console unless distributed to other users with the ALLOW command.

Example

To request that volume set ACCOUNTING_PAYROLL be released for all users on the system, enter:

```
 VSRELEASESYS ACCOUNTING_PAYROLL
```

Related Information

Commands The VSxxxxxx commands in this chapter, DISMOUNT

Manuals *Volume Management Reference Manual*

VSRESERVE

Notifies the system to keep a particular volume set online. The equivalent compatibility mode command is `MOUNT`. (Native Mode)

Syntax

>VSRESERVE [*volumesetname*] [;GEN=*genindex*]

Parameters

volume- setname The name of the MPE/iX volume set to be kept online. If you omit the parameter, the request is issued for the home volume set of the user's logon group and account. Refer to "Operation Notes."

genindex A value from -1 to 32,767 specifying which generation of the volume set is to be kept online. If you omit the parameter, the system does not check the generation version of the specified volume set.

Operation Notes

This command calls for the specified volume set to be kept online, and prevents the console operator from taking a particular volume set offline. Once this is done, the volume set is designated as being in use by the user. It remains in this reserved state until the user issues a `VSRELEASE` command or the operator issues a `VSCLOSE ;NOW` command; or until the user logs off.

The MPE/iX naming convention for volume sets differs from that of MPE V/E for private volumes.

In MPE V/E, the name `A.B.C` indicates that `B` is the name of a group and that `C` is the name of an account. MPE/iX accepts that name, but no interpretation is made as to the referencing of `B` and `C`. Instead, MPE/iX treats `A.B.C` as a single, long string name. It is the flexibility of the MPE/iX naming convention that makes it possible for MPE/iX to work with a volume set designated `A.B.C`.

MPE/iX volume set names may consist of any combination of alphanumeric characters, including the underbar (_) and the period (`.`). The name must begin with an alphabetic character and consist of no more than 32 characters.

A volume set called `MY_OWN_PERSONAL_VOLUME_SET` is acceptable in MPE/iX, and so is `MY.OWN.PERSONAL.VOLUME.SET`; similarly, `A.B.C` is acceptable.

If a volume set is named according to the MPE V/E naming convention (`A.B.C`), you must use an unambiguous reference when using the MPE/iX volume set commands:

```
 Vcommand A.B.C
```

Entering `Vcommand A` fails to access the volume set. You cannot specify the first part of the volume set name alone and expect the group and account to default.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. Use volumes (UV) or create volumes (CV) capability is required to use this command.

Example

To request that volume set `ACCOUNTING_PAYROLL` be kept online, enter:

```
VSRESERVE ACCOUNTING_PAYROLL
```

Related Information

Commands	The `VSxxxxxx` commands in this chapter, `DISMOUNT`
Manuals	*Volume Management Reference Manual*

VSRESERVESYS

Instructs the system to reserve a volume set online system-wide. The equivalent compatibility mode command is LMOUNT. (Native Mode)

Syntax

VSRESERVESYS ***volumesetname***

Parameters

volume- setname The name of the MPE/iX volume set to be kept online.

Operation Notes

This command calls for the specified volume set to be kept online and reserved system-wide and specifies that the volume set be kept online until a VSRELEASESYS command is issued. This command does not prohibit individual VSRESERVE (MOUNT) or VSRELEASE (DISMOUNT) commands issued for the specified volume set by individual users.

The MPE/iX naming convention for volume sets differs from that of MPE V/E for private volumes. Refer to the DISMOUNT, MOUNT, VSRELEASE, and VSRESERVE commands in this chapter.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the system console unless distributed to other users with the ALLOW command.

Examples

To request that the volume set ACCOUNTING_PAYROLL be put online and reserved for all users on the system, enter:

```
 VSRESERVESYS ACCOUNTING_PAYROLL
```

Related Information

Commands	The VSxxxxxx commands in this chapter, DISMOUNT
Manuals	*Volume Management Reference Manual*

VSTORE

Verifies that the data on a backup media are valid (for example, there are no media errors), and reports any errors incurred by STORE when creating the backup.

Syntax

VSTORE[*vstorefile*] [;*filesetlist*] [;*option* [;...]]

where *option* is:

[;SHOW [=*showparmlist*]]
[;ONERROR= { QUIT SKIP }]
[;DIRECTORY]
[;PROGRESS [=*minutes*]]
[;COPYACD [;NOACD]]
[;TREE [;NOTREE]]
[;NODECOMPRESS]
[;STOREDIR] [ECTORY] =*directoryname*] [;PART [IAL] DB]
[;RESTORESET=(*device* [,...])]

The following parameters are available with TurboStore/iX II and TurboSTORE/iX True-Online Backup products only:

[;RESTORESET=(*device*[,...]) [,(*device*[,...])[,...]]]
[;MOSET=(ldev[,...])[,(ldev[,...]) [,...]]
[;NAME=*backupname*]

Parameters

vstorefile The name of the device that contains the files you want verified on the system. This file must be backreferenced, using an asterisk (*). A File equation for *vstorefile* should be set up before invoking VSTORE. If you want to verify files from a file called SOURCE enter this file equation before running VSTORE:

```
:< user FILE SOURCE;DEV=TAPE
```

The *vstorefile* can now reference a remote device. For example,

```
:< user |FILE REMOTE;DEV=REMSYS#TAPE|
:< user |VSTORE *REMOTE;@;SHOW|
```

NM Vstore will verify all files from the specified remote device. Although the initial tape mount request will appear on the remote console, all of Vstore's console messages will be displayed on the local console. Currently, labeled tapes and Magneto-optical devices cannot be used for remote verification.

A message is displayed on the system console requesting the operator to mount the tape identified by the *vstorefile* parameter and to allocate the device.

If *vstorefile* is not supplied and the RESTORESET option is not used, then VSTORE creates a default file name. The default file name is the user's logon username. No file equation is used.

Sequential and parallel devices are specified with the RESTORESET option. Similarly, magneto-optical devices are specified using the MOSET option. You should not specify *vstorefile* when using RESTORESET or MOSET.

A disk file can also be specified with a file equation for *vstorefile*. An example of such a file equation would be:

```
 :< user |FILE MYDISC=DISCBACK.DAILY.BACKUP;DEV=DISC|
```

Note that DEV=DISC must be specified for VSTORE to verify files from disk backups. All other information in the file equation will be ignored by VSTORE.

NOTE TurboSTORE/iX 7x24 True-Online Backup must be used to create disk backups.

filesetlist Specifies the set of files to be verified. The default depends on the user's capability, as shown below:

Default	Capability
@	None
@.@	Account manager (AM)
@.@.@	System Manager and/or System Supervisor (OP)

The is parameter has the form shown below:

filesetitem[,*filesetitem*[...]]

where *filesetitem* can be ^*indirectfile* or *fileset*.

indirectfile A file name that backreferences a disk file. The syntax is ^*indirectfile* or !*indirectfile*. This file may consist of *fileset(s)* and *option(s)*, but only options can appear after the first semicolon (:) on each line. An option specified on one line will operate on all files in the *filesetlist*. ^*indirectfile* is the preferred format. If you use !*indirectfile*, the CI will interpret this as a variable reference, so you will have to specify !!*indirectfile* instead.

fileset Specifies a set of files to be verified, and optionally those files to be excluded from the VSTORE operation. The *fileset* parameter has the form:

filestovstore[-*filestoexclude*[..]

Any file that matches *filestovstore* will be verified unless the file also matches a *filestoexclude*, which specifies files that are to be excluded from the VSTORE operation. You may specify an unlimited number of *filestoexclude*.

Since "-" is a valid character for HFS syntax file names, a blank character must separate it from HFS file sets to obtain the special negative file set meaning.

filestovstore *filestoexclude*	Both *filestovstore* and *filestoexclude* may be entered in MPE or HFS syntax. Wildcards are permitted for both MPE and HFS syntax.

The MPE syntax is as follows:

filename[.*groupname*[.*accountname*]

A lockword may be specified for files to be verified, in the form:

filename/lockword.group.account

The HFS syntax is as follows:

/dir_lev_1/dir_lev_2/.../dir_lev_i/.../filedesig

or

./dir_lev_i/dir_lev_j/.../dir_lev_k/.../filedesig

If the name begins with a dot (.), then it is fully qualified by replacing the dot with the current working directory (CWD).

Each of the components *dir_lev_i* and *filedesig* can have a maximum of 255 characters with the full path name being restricted to 1023 characters. Each of the components *dir_lev_i* and *filedesig* can use the following characters:

Letters a to z

Letters A to Z

Digits 0 to 9

Special characters - _ .

For HFS name syntax, the lowercase letters are treated distinctly from the uppercase letters (no upshifting). Names in MPE syntax are upshifted.

Both MPE and HFS name components can use the characters @, #, and ? as wildcard characters. These wildcard characters have the following meaning:

@	specifies zero or more alphanumeric characters.
#	specifies one numeric character.
?	specifies one alphanumeric character.

These wildcard characters can be used as follows

n@	Verify all files starting with the character n.
@n	Verify all files ending with the character n.
n##...#	Verify all files starting with character n followed by up to seven digits (useful for storing all EDIT/3000 temporary files).
n@x	Verify all files starting with the character n and ending with the character x.

`?n@` Verify all files whose second character is `n`.

`n?` store all two-character files starting with the character `n`.

`?n` Verify all two-character files ending with the character `n`.

Also, character sets may be specified in the following syntax:

`[ct]` specifies letter `c` or `t`.

`[c-t]` specifies any letter from range `c` to `t`.

`[e-g1]` specifies any letter range `e` to `g` or digit `1`.

Examples of using character sets are:

`[A-C]@` Verify all files that begin with the letters `A, B,` or `C`.

`myset[e-g1]` Verify all files that begin with the name `myset` and end in `e, f,` or `g`, or `1`.

`myset`
`[d-e1-6]` Verify all files that begin with the name `myset` and end in `d` or `e`, or `1, 2, 3, 4, 5,` or `6`.

You may specify up to a maximum of sixteen characters for each character set and you may not nest brackets.

A character set specifies a range for only one (1) ASCII character. The range `[a-d]@` gets all files that begin with the letter `a` through the letter `d`. The ranged `[ad-de]` may cause unpredictable results.

Since the hyphen (-) is a valid character for HFS syntax file names, it is allowed inside a character set, immediately following a left bracket ([) or preceding a right bracket (]). When specified between two characters, the hyphen implies a range of characters.

Specifying Database Files

When specifying TurboIMAGE and ALLBASE/SQL databases to be verified, only the root file or DBCon file needs to be specified. `VSTORE` will determine which other files belong to that database, and will verify all of them. If dataset file(s) are specified without specifying a root file, then a warning will be printed for each file, and they will not be verified. Individual database files can be verified without the root file by specifying the `;PARTIALDB` option on the `VSTORE` command line.

MPE and HFS Naming Equivalences

When an MPE name component is a single `@` wildcard, the `@` will be "folded" to include all MPE and HFS named files at that level and below. To specifiy only MPE-named files, use `?@` instead.

MPE wildcards are not expanded in *filestoexclude*. This means that `@.@.@-@.@.@` is NOT an empty fileset. It contains all of the HFS named files on the system.

A fileset may be entered in any of the following formats and may use wildcard characters. Equivalent MPE and HFS formats are grouped together as follows.

file.group.acct/ACCT
/GROUP/FILE' One particular file in one particular group in one particular account.

file.group/LOGON-
ACCT/GROUP/
FILE One particular file in one particular group in the logon account.

file
./FILE One particular file in the logon group and account.

@.group.acct
/ACCT/GROUP/ All files (MPE and HFS) in one particular group in one particular account.

?@.group.acct All MPE name files in one particular group in one particular account.

@.group/LOGON-
ACCT/GROUP/ All the files (MPE and HFS) in one particular group in the logon account.

?@.group All MPE named files in one particular group in the logon account.

@.@.acct
/ACCT/ All the files (MPE and HFS) in all the groups in one particular account, plus all the files and directories under the specified account.

thisisit.@.account Any MPE file named `thisisit` in all groups in one particular account.

?@.@.acct All MPE named files in all the groups in one particular account.

@ All (MPE and HFS) files in the CWD. This is the default for everyone, regardless of permissions.

@.@ All (MPE and HFS) files in the logon account.

@.@.@ All the files and directories (MPE and HFS) on the system.

?@.@.@ All MPE named files on the system.

`SHOW` Request to list names of verified files. The default is a listing of only the total number of files verified, list of files not verified (including the reason each was not verified), and the count of files not requested to be verified. The listing is sent to $STDLIST (formal designator SYSLIST) unless you enter a `FILE` command to send the listing to some other device. For example, you would enter the following file equation before the `VSTORE` command to send the listing to a line printer.

```
FILE SYSLIST; DEV=LP
```

showparmlist Tells `VSTORE` what information to display for the files that are verified. If you specify `;SHOW` and you omit *showparmlist*, then the default is `SHORT` if the recordsize of SYSLIST is less than 132 characters, or `LONG` if the recordsize is equal to or greater than 132. The format for *showparmlist* is:

showparm `[,`*showparm*`[,`*showparm*`[,...]`

where *showparm* may be one of the options described below. If you do not specify `SHORT` or `LONG`, then the base information is `SHORT` if SYSLIST is less than 132 characters, or `LONG` if SYSLIST is 132 characters or more.

NOTE If an HFS-named file is specified in the *filesetlist*, or the expansion of a wildcard includes an HFS-named file, then an HFS-style output listing will be used. This listing shows the same information as the MPE format, but puts the name of the file at the right end of the listing, to allow for longer HFS names. If a HFS name is too long to fit in the record size of the output file, it will be wrapped onto the next line. Wrapping is signified by a "*" as the last character on the line.

showparm

SHORT — Overrides a default of LONG and displays file, group, and account name or the fully qualified path name, volume restrictions, file size (in sectors), file code, and media number.

LONG — Overrides a default of SHORT and displays all the information that SHORT does and adds the ending reel number, record size, blocking factor, number of extents, EOF, and file starting and ending media number. For spoolfiles, the old spoolfile name is also displayed.

NAMESONLY — Displays only the filename and the starting and ending media number. NAMESONLY is not allowed with SHORT or LONG.

DATES — Displays the creation date, the last date of access, and the last date of modification.

SECURITY — For MPE format listing, causes SHOW to display the creator and the file access matrix for all the files which do not have an active ACD. For files with active ACDs only, the phrase *ACD EXISTS* is displayed.

For HFS format listing, the phrase *ACD EXISTS* or *ACD ABSENT* is displayed, depending on whether the file has an ACD.

PATH — Forces all file listings to be in HFS format. Full HFS pathnames are displayed instead of MPE style names.

OFFLINE — Sends another copy of the SHOW output to the formal file designator OFFLINE, which defaults to device LP.

ONERROR

Tells VSTORE what to do if there is a tape read error. If you omit this parameter, then the default option is QUIT for labeled and unlabeled tapes. ONERR is a synonym for ONERROR.

QUIT — Tells VSTORE to abort after a tape read error.

SKIP — Tells VSTORE to perform a file-skip-forward past a tape error, resynchronize, and resume reading from the tape.

DIRECTORY

Specifies that the file system directory is to be verified. Requires OP or SM capability. HFS directories on the media are always verified.

`PROGRESS` Instructs `VSTORE` to report its progress at regular intervals by displaying the message `VSTORE OPERATION IS nnn% COMPLETE`. For interactive users, this message is displayed on $STDLIST. For jobs, this message is sent to the system console.

minutes A positive number specifying the number of minutes between progress messages. The maximum is 60. The default is 1 (one) minute.

`COPYACD` Directs `VSTORE` to copy the ACD associated with the files or directories on the media. This option is on by default.

`NOACD` Directs `VSTORE` to not copy the ACD associated with the files or directories on the media. This option overrides the default `COPYACD` option.

`TREE` Forces each fileset to be scanned recursively. This is equivalent to using the trailing slash (/) in an HFS name. The `TREE` option yields a recursive scan in the hierarchical directory. This option is mutually exclusive with `NOTREE`.

`NOTREE` Forces each HFS syntax fileset to not be scanned recursively. The `NOTREE` option yields a horizontal cut in the hierarchical directory. The `NOTREE` option is mutually exclusive with `TREE`.

`NODECOMPRESS` Normally, `VSTORE` will decompress the data on a Store-compressed media when verifying the files. However, when `NODECOMPRESS` is specified, the files will not be decompressed. Instead, just the integrity of the raw data read from the media will be checked. This results in a faster `VSTORE` of the media, which just verifies physical consistency.

`STOREDIREC`
`TORY` Specifies that `VSTORE` should use the supplied *directoryname* when looking for the disk store directory file. This option should be specified if the disk directory file for this backup resides in a directory other than the default path of `/SYS/HPSTORE/store_dirs/`. This file should be either a directory file created by `STORE`, or a symbolic link pointing to one.

directoryname The name of the disk directory file to be used by `VSTORE`. It can be in either MPE or HFS format. If it is not a fully qualified filename, it will be qualified by the CWD. This file should either be a disk directory file created by `STORE` or a symbolic link pointing to one.

`PART[IAL]DB` Allows `VSTORE` to verify individual database dataset files without specifying the database's root or DBCon file.

THE FOLLOWING OPTIONS ARE AVAILABLE ONLY IF TURBOSTORE XL OR TURBOSTORE XL II IS INSTALLED ON YOUR SYSTEM. TURBOSTORE IS NOT PART OF THE FUNDAMENTAL OPERATING SYSTEM, BUT MAY BE PURCHASED SEPARATELY.

For additional information on TURBOSTORE XL, refer to the *Store and Turbostore/iX Manual* (30319-90001).

`RESTORESET` Specifies parallel and sequential backup devices. This option cannot be use if the *vstorefile* parameter is specified.

Consecutive tapes are specified in the following way:

```
< user ;RESTORESET = (*tape1,*tape2,*ta pe3,...)
```

This instructs MPE/iX to use only one drive at a time for the vstore operation. When the first reel of tape is exhausted, VSTORE will shift to the next available drive, leaving the first free for rewinding and changing reels. Thus, at any given time, only one drive is occupied with the VSTORE operation.

Parallel devices are specified by

```
|;RESTORESET=(*tape1),(*tape2),(*tape3)...|
```

In this example, all three tapes will be used in parallel during the VSTORE operation.

A set of sequential tapes to be verified in parallel would be specified by

```
|;RESTORESET=(*tape1,*tape2),(*tape3,*tape4)|
```

In this example, two tapes would be verifying at any particular moment, while the other two are rewinding, permitting the operator to switch reels.

This option cannot be used if the *vstorefile* parameter is specified.

device Specifies the device from which the files are to be verified. It must be a magnetic tape or DDS. This device should be specified in a file equation before you invoke the VSTORE command, ie:

```
:< user FILE DEVICE;DEV=TAPE
```

This file equation can also specify a remote device or a disk file.

MOSET Specifies parallel Magneto Optical (MO) backup devices. This option is not available if the *storefile* option is specified.

Parallel devices are specified by either of the two following commands:

```
< user ;MOSET = (12),(13),(15)
```

```
< user ;MOSET = (MO),(MO),(MO)
```

All MO devices are used in parallel during the vstore process. The preferred format is specifying just "MO", since VSTORE will use the the NAME parameter to locate the correct media.

This option is not available if the *vstorefile* parameter is specified.

NAME This parameter must be specified with the MOSET option, and cannot be specified without it. It specifies the logical name to be used for the backup. For example:

```
< user VSTORE @.@.@;;MOSET=(12);NAME=DAILY.D23OCT90.BOZO
```

This name could indicate that the VSTORE process should be taken from the daily backup done on 23 Oct 1990 on the system called BOZO.

backupname A three field name of a total maximum length of 26 characters. The format is *fname.gname.aname*. The name represents the "handle" to this particular backup and can is used to retrieve files from this backup. The *fname*, *gname* and *aname* can be up to 8 alphanumeric characters. For example DAILY.D24OCT90.SYSTEM.

Operation Notes

This command verifies that there are no media errors on backup media. It reports any errors that may have occurred during the STORE procedure.

Your capabilities determine which files you may verify. If you have system manager or system supervisor capability, you can verify any file from a STORE backup. If you have account manager capability, you can verify any file in your account. To verify files with negative file codes, you need Privileged Mode (PM), system supervisor (OP), or system Manager (SM) capability. If you have standard user capability, you can verify only those files in your logon account.

This command applies only to NMSTORE tapes created in Native Mode. It does not work on tapes created by Compatibility Mode STORE.

The LOCAL option is no longer needed to verify files. All files will be verified and displayed with their real filenames, even if parts of a file's accounting structure do not exist on the system.

Use

This command may be issued from session, job, or program, but not in BREAK. If you press [Break] during a Vstore, the operation continues while you interact with the CI.

EXAMPLE

To verify all files in a system:

1. Write a file equation to set up a device file.

```
:FILE T;DEV=TAPE
```

2. Use the VSTORE command and backreference the device file.

```
:VSTORE *T;@.@.@;KEEP;SHOW
```

Related Information

Commands	STORE, RESTORE
Manuals	*STORE and TurboSTORE/iX Manual*
	Magneto-Optical Media Management User's Guide

VSUSER

Displays all users of a currently reserved, mountable volume set. (Native Mode)

Syntax

VSUSER[*volumesetname*]

Parameter

volume- setname A fully qualified volume set name. Default is that information for all currently reserved volume sets is displayed.

Operation Notes

The VSUSER command lists all users who have explicitly or implicitly reserved a mountable volume set. It also displays the volume set name, job number, and the job names of all users currently performing a reserve function. The VSRESERVE/VSRELEASE commands enable users to perform explicit reserving and releasing. The FOPEN/FCLOSE intrinsics enable them to perform implicit reserving and releasing.

The MPE/iX naming convention for volume sets differs from that of MPE V/E for private volumes. Refer to the MOUNT, DISMOUNT, VSRESERVE, and VSRELEASE commands in this chapter.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. Use volumes (UV) or create volumes (CV) capability is required to use this command.

Example

To display all of the currently reserved volume sets, enter:

```
VSUSER
VOLUME SET NAME   JOBNUM    JOBNAME
-
USER_MANAGER    #S260    NORMA.MPEM
```

Related Information

Commands	The VSxxxxxx commands in this chapter
Manuals	*MPE/iX Intrinsics Reference Manual*
	Volume Management Reference Manual

WARN

Sends an urgent message to jobs/sessions.

Syntax

WARN{ @ [#] J*nnn* [#] S*nnn* [*jsname*,] *user.acct* } [;*message*]

Parameters

@ All users receive the message (including those running in QUIET mode).

#J*nnn* A job number (assigned by MPE/iX) for the job that is to receive the message.

#S*nnn* A session number (assigned by MPE/iX) 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 running. (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/iX sends the message to all of them.

message The message text, consisting of any string of ASCII characters containing no more than 67 characters. The message is terminated by **Return**. Default is that no message is printed.

Operation Notes

Sends an urgent message, interrupting any current pending read or write in progress. The message appears on the list devices of all sessions (even those that are QUIET) as:

```
OPERATOR WARNING:  text
```

Messages sent with the `WARN` command are received by a job only if the job was submitted on an interactive device.

The user has the option of running a session in QUIET mode. In that case, `TELL` messages from other users are 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.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console unless distributed to users with the `ALLOW` command.

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.
```

Related Information

Commands	TELL, TELLOP
Manuals	*Performing System Operation Tasks*

WELCOME

Defines the welcome message.

Syntax

WELCOME[*welcfile*]

Parameters

welcfile	An ASCII file containing the welcome message.

Operation Notes

The operator uses the `WELCOME` command to compose the message that is transmitted to users when they initiate jobs and sessions. The message is retained when you issue a `START RECOVERY`, `START NORECOVERY`, or `UPDATE/UPDATE NOCONFIG` restart option.

To define the welcome message, enter `WELCOME` and press **Return**. When the # prompt is displayed, begin entering the text of the message. The length of any line cannot exceed 72 characters and the total number of lines may not exceed 26. To terminate the message and complete the command, enter **Return** at the # prompt.

To define the welcome message from an editor file, specify the *welcfile* parameter. Subsequent changes to the editor file do not affect the welcome message until another `WELCOME` command is issued with the file name specified.

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 press **Return** at the # prompt.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** has no effect on this command. It may be issued only from the console unless distributed to users with the `ALLOW` command.

Example

To create a multiline welcome message, enter:

```
WELCOME

#WELCOME TO THE HP3000 COMPUTER SYSTEM.

#FILES WILL BE STORED EACH DAY BETWEEN 6AM AND 7AM.

#Return
```

Related Information

Commands	`HELLO`, `SHOWME`
Manuals	*Performing System Operation Tasks*

WHILE

Used to control the execution sequence of a job, session, UDC, or command file. (Native Mode)

Syntax

`WHILE` *expression* [DO]

Parameters

expression Logical expression, consisting of operands and relational operators. Table 8-1 lists the operators that may be incorporated in *expression*.

Table 8-1 Logical Operators - The WHILE Command

Logical operators:	AND, OR, XOR, NOT
Boolean functions and values:	BOUND, TRUE, FALSE, ALPHA, ALPHANUM, NUMERIC, ODD
Comparison operators:	=, <>, <, >, <=, >=
Bit manipulation operators:	LSL, LSR, CSR, CSL, BAND, BOR, BXOR, BNOT
Arithmetic operators:	MOD, ABS, * , / , + , -, ^ (exponentiation)
Functions returning strings:	CHR, DWNS, UPS, HEX, OCTAL, INPUT, LFT, RHT, RPT, LTRIM, RTRIM, STR
Functions returning integers:	ABS, LEN, MAX, MIN, ORD, POS, TYPEOF
Other functions:	FINFO, SETVAR

Use HELP FUNCTIONS |OPERATORS |EXPRESSIONS for more info

The `WHILE` command evaluates *expression* and displays the result (TRUE or FALSE) to `$STDLIST`. If *expression* does not resolve to a Boolean result, an error is reported.

The `DO` keyword is optional. It may be used or omitted and has no affect on the results.

Operation Notes

This command begins a WHILE block, which consists of all the commands lying between WHILE and the next ENDWHILE statement. The ENDWHILE must have the same nesting level as the WHILE statement. The ENDWHILE statement ends the WHILE block.

The logical expression is evaluated and, as long as *expression* evaluates to TRUE, the WHILE block is executed.

Nesting of IF and WHILE blocks is limited to a *combined* total of 30 levels. Each IF or WHILE block read by the Command Interpreter increments the nesting count by 1.

NOTE You may not write a WHILE construct in such a way that it physically crosses from one user command (UDCs or command files) to another.

Use

This command may be issued from a session, job, program, or in BREAK. Pressing **Break** aborts the execution of this command.

Example

The following is an example of the `WHILE` command:

```
WHILE SETVAR (FILENAME, &
   INPUT ("PLEASE ENTER THE NEXT FILENAME TO PURGE:") &
   )<> "" DO
 PLEASE ENTER THE NEXTFILENAME TO PURGE: OLDFILE1
*** EXPRESSION TRUE
CONTINUE
PURGE !FILENAME
ENDWHILE
```

Related Information

Commands	`ELSE`, `ELSEIF`, `ENDWHILE`, `ESCAPE`, `IF`, `RETURN`, `SETVAR`, `SHOWVAR`
Manuals	Appendix B, "Expression Evaluator Functions"

XEQ

Executes any program or command file. Its value lies in preventing any possible confusion when the name of the program or command file you want to execute is identical to the name of a built-in MPE/iX command or UDC command name. (Native Mode)

Syntax

XEQ ***filename***[*parameterlist*] *

or

XEQ ***filename***[;INFO=*quotedstring*] [;PARM=*parmvalue*] **

```
 * for command files
 ** for program files
```

Parameters

filename The actual file name of the command file or program file to be executed. The search path (HPPATH) is used if *filename* is not qualified.

parameterlist The list of parameters passed to *filename* when executing a command file. This list corresponds to the PARM line(s) of the command file you intend to execute.

quotedstring A parameter string for those program files that accept a parameter string.

parmvalue A parameter for a program file to be executed.

Operation Notes

This command executes *filename*, which may be a command file or a program file. XEQ uses the search path. XEQ is needed only when *filename* references an existing, built-in MPE/iX command or a UDC command, but it may be used for any executable file.

Use

This command may be issued from a session, job, program, or in BREAK. Whether or not the command is breakable depends upon what is being executed at the time you press **Break**. Command files may terminate or suspend execution, unless they specify OPTION NOBREAK.

Example

To execute a command file named FCOPY.PUB.MYACCT, enter:

```
XEQ FCOPY
```

Because FCOPY references an existing, built-in MPE/iX command, failing to use XEQ results in running FCOPY. That happens because FCOPY is found in the command directory and is executed, and the command search terminates. XEQ follows the same searching logic used for command files and implied RUN.

Related Information

Commands	RUN
Manuals	None

A Predefined Variables in MPE/iX

Many variables have been predefined for use by the command interpreter. They may be used anywhere you would use your own variables. Table A-1 lists all valid predefined variables.

Table A-1 Predefined Variables

Variable	Type[a]	Definition	Initial Value
CIERROR	W JCW	last CI error number	zero
HPACCOUNT	R S	user's account name	logon account
HPACCTCAP	R I	current account capability mask	logon account caps
HPACCTCAPF	R S	current account formatted capability mask, for example, "AM, AL, GL, ND, SF, BA, IA"	logon account caps
HPAUTOCONT	W B PL	enables (TRUE); disables (FALSE) the automatic CONTINUE feature	FALSE
HPCIDEPTH	R I PL	number of nested CIs	1(=Root CI)
HPCIERR	W I	last CI error/warning in current session	zero
HPCIERRCOL	W I	error column number for last CI error/warning	zero
HPCIERRMSG	R S	textual message for the most recent CIERROR (length of message is 0 for nonexistent CIERROR values)	(null)
HPCMDNUM	R I PL	current command sequence number	1
HPCMDTRACE	W B PL	enables (TRUE); disables (FALSE) the User Command Tracing facility	FALSE
HPCMEVENTLOG	W I	when set to *n*, $STDLIST displays the following *n* occurrences of tos/reg trap	zero
HPCONNMINS	R I	current session connect-time in minutes	zero
HPCONNSECS	R I	current session connect-time in seconds	zero
HPCONSOLE	R I	LDEV of the console	console LDEV at logon
HPCONTINUE	R B PL	CI's continue state: FALSE=inactive, TRUE=active	FALSE
HPCPUNAME	R S	name of computer model, for example, "SERIES 930"	name of your logon computer model

Variable	Type[a]	Definition	Initial Value
HPCPUMSECS	R I	from root CI = current session CPU-time in milliseconds; from other CI or process = current process CPU-time in milliseconds	zero
HPCPUSECS	R I	from root CI = current session CPU-time in seconds; from other CI or process = current process CPU-time in seconds	zero
HPCWD	R S	current working directory	logon group and account
HPDATE	R I	current day of month	logon day of the month
HPDATEF	R S	current formatted date	logon date
HPDAY	R I	current day of the week (1=SUNDAY)	logon day of the week
HPDTCPORTID	R S	port ID of data terminal	null string
HPDUPLICATIVE	R B PL	indicates whether or not input operations are echoed to a correspondind device; TRUE (duplicative)= echoing occurs; FALSE (nonduplicative)= no echoing occurs	as appropriate
HPERRDUMP	W I PL	number of errors to be dumped from process error stack	zero
HPERRTOSLIST	W B PL	controls destination of CI error messages: TRUE=errors written to $STDLIST, FALSE=errors written to $STDERR	TRUE
HPEXECJOBS	R I	number of jobs and sessions currently in EXEC (executing) state	number of jobs and sessions in EXEC state
HPFILE	R S	currently executing UDC or command file	null string
HPGROUP	R S	current group name	logon group name
HPGROUPCAP	R I	current group capability mask	logon group caps
HPGROUPCAPF	R S	current group formatted capability mask, for example, "IA,BA,PH"	logon group caps
HPHGROUP	R S	home group name	home group
HPHOUR	R I	current hour number (24-hour clock)	logon hour
HPINBREAK	R B PL	FALSE=not in BREAK; TRUE=in BREAK mode (includes process BREAK and rit BREAK)	FALSE
HPINPRI	R I	input priority	logon input priority

Variable	Type[a]	Definition	Initial Value
HPINTERACTIVE	R B PL	interactive (TRUE); noninteractive (FALSE)	as appropriate
HPINTRODATE	R S	formatted job/session logon date	date of logon
HPINTROTIME	R S	formatted job/session logon time	time of logon
HPJOBCOUNT	R I	number of jobs executing	logon number of executing jobs
HPJOBFENCE	R I	fence value for waiting jobs	logon jobfence
HPJOBLIMIT	R I	current job limit	job limit at logon
HPJOBNAME	R S	name of current job/session	logon job name
HPJOBNUM	R I	job/session number, for example, 12	your job/session number
HPJOBTYPE	R S	"S"=session, "J"=job	your job type
HPLASTJOB	W S	job ID of the job you most recently streamed in the form #Jnnnn	null string
HPLASTSPID	R S	spoolfile ID for the job identified in the HPLASTJOB variable	spoolfile ID of last job
HPLDEVIN	R I	LDEV number for $STDIN	logon input LDEV
HPLDEVLIST	R I	LDEV number for $STDLIST	logon output LDEV
HPLOCIPADDR	R S	IP address of a remote client	null string
HPLOCPORT	R I	TCP port number for network service provided to the client	0 if local client; otherwise standard port used by service
HPMINUTE	R I	current minute number	logon minute
HPMONTH	R I	current month number	logon month
HPMSGFENCE	W I PL	fence for the level of error messages printed by the CI: See HELP HPMSGFENCE for values and expression evaluation diagnostics	0
HPNCOPIES	R I	number of $STDLIST copies for jobs	copies subparm of the outclass= parm of the JOB command
HPOSVERSION	R S	operating system version ID (identical to the middle version string in the SHOWME banner	current full version ID of the operating system
HPOUTCLASS	R S	output device class	logon output device class

Variable	Type[a]	Definition	Initial Value
HPOUTFENCE	R I	output fence value	logon output fence value
HPPATH	W S	search path for command files and implied RUN	"!HPGROUP,PUB,PUB.SYS, ARPA.SYS"
HPPIN	R I	Process Identification Number for the executing process	PIN for the root CI
HPPROMPT	W S	CI's prompt string	":" (colon)
HPQUIET	R B	indicates if session is accepting messages: FALSE= accepting messages; TRUE= not accepting messages ("quiet")	FALSE
HPREDOSIZE	W I PL	number of entries in the CI's redo stack	20
HPRELVERSION	R S	operating system release version ID (identical to the left version string in the SHOWME banner	current full version ID of the operating system
HPREMIPADDR	R S	IP address of the remote user	null string
HPREMPORT	R I	TCP port number, assigned by the client, used on an incoming connection	0 if local client; otherwise the assigned TCP port number
HPRESULT	W S I or B	value of the most recent CALC command evaluated (for example, "abc", 12, TRUE)	zero
HPSCHEDJOBS	R I	number of jobs currently in SCHED state (scheduled state)	number of jobs in SCHED state
HPSESCOUNT	R I	number of sessions executing	logon number of sessions executing
HPSESLIMIT	R I	current session limit	session limit at logon
HPSPOOLID	R S	spoolfile ID of the current job	job spoolfile ID
HPSTDIN	R S	file name for job or session input	$STDIN
HPSTDLIST	R S	file name for job or session output listing	$STDLIST
HPSTREAMEDBY	R S	user and account name of the person who streamed a job or invoked STARTSESS; if "the person" is the initial OPERATOR.SYS logon or a job streamed from the SYSSTART.PUB.SYS file, the job or session ID is replaced by the string SYSTEM PROCESS	logon ID of the person who streamed the job

Variable	Type[a]	Definition	Initial Value
HPSUSAN	R I	unique serial number assigned at the factory to each system for use by software	unique serial number assigned to your system at manufacture
HPSUSPJOBS	R I	current number of jobs in SUSP state (suspended)	numbers of jobs in SUSP state at logon
HPSYSNAME	W S	name of computer system (user-definable)	null string (" ")
HPTIMEF	R S	current formatted time	logon time
HPTIMEOUT	W I PL	number of minutes for CI reads (<=0 means no timeout). When this expires on a CI read, session is logged off.	zero
HPTYPEAHEAD	W B	indicates whether or not typeahead is turned on; the BYE or SETVAR commands reset this variable to FALSE.	FALSE.
HPUSER	R S	current user name	logon user
HPUSERCAP	R I	current user's capability mask	logon user caps
HPUSERCAPF	R S	current user's formatted capability mask, for example, "IA,BA,PH"	logon user caps
HPUSERCMDEPTH	R I PL	number of nested UDCs and/or command files	zero
HPUSERCOUNT	R I	number of current online users	0 if user-based pricing is *not* installed; otherwise the number of current users
HPUSERLIMIT	R I	limit of number of online users	-1 if user-based pricing is *not* installed; otherwise the user limit number
HPVERSION	R S	MPE/iX version id (*v.uu.ff*)	current MPE/iX version
HPWAITJOBS	R I	current number of jobs waiting	number of jobs waiting at logon time
HPYEAR	R I	last two digits of the current year	logon year number
JCW	W JCW	job control word (variable)	zero

a. R	READ ONLY variable (cannot be modified).
W	READ/WRITE variable (can be modified).
JCW	A standard MPE/iX JCW.
I	Integer format.
B	Boolean format (TRUE/FALSE).
S	String (ASCII) format.
PL	Process Local. Modifications exist only for the locality of the process.

If a PL variable is changed by a process it returns to its original value when the process terminates. For example, if you logon and set `HPREDOSIZE` (the number of entries in the CI's redo stack) to 25 and then run a program which sets it to 30 (using the `COMMAND` or `HPCICOMMAND` intrinsics) the variable will have the value 30 for that process *only*. When the process terminates the value of this variable for your session remains at the value it was *before* the program was run (in this case 25).

PL (process local) variables are *not* programmatically accessible with the `HPCIGETVAR`, `HPCIPUTVAR`, and `HPCIDELETEVAR` intrinsics. They may be programmatically accessed only with the `COMMAND` or `HPCICOMMAND` intrinsics.

Note that `HPTYPEAHEAD` cannot be set inside a job. All user-created variables may be modified and deleted. However, Hewlett-Packard predefined variables may not be deleted.

JCWs may be considered integer variables with legal values ranging from 0 to 65,535 and with bits 16 and 17 (bit 0 being the leftmost bit of 32 bits) having special interpretations (for example, if bit 16 is set, the JCW setting is `FATAL`) and with bits 0 through 15 reserved.

B Expression Evaluator Functions

The expression evaluator is a system procedure used by the user interface to accept a string, number, or Boolean expression, evaluate it, and return the result. This procedure is used by the CALC, SETVAR, IF, ELSEIF, and WHILE commands and within a ![ ]..

The expression evaluator provides the following:

- consistent evaluation of expressions
- compatibility with MPE V/E job control word evaluation
- user flexibility

The expression evaluator uses algebraic notation and supports the functions defined in Table B-1. The references that appear in this table in parentheses, for example (8), are defined following the table.

See HELP for description of new functions and examples.

E.g. HELP FSYNTAX

Table B-1 Expression Evaluator Functions

Symbol	Function	Example	Result
+(numeric)	addition	4 + 5	9
+(string)	concatenate	“abc‘ + “de‘	abcde
-(numeric)	subtraction	12 - 6	6
-(string)	deletion of first occurrence	“abc‘ - “b‘	ac
*	multiplication	4 * 5	20
/	integer division	79/ 10	7
ˆ	exponentiation (9)	2ˆ3	8
either “ or ‘	string identifier	either “abc‘ or “abc‘	abc
()	parentheses	(3 + 4) * 2	14
<	less than (1)	5 < 6	TRUE
<=	less than or equal (1)	“abc‘ <= “abc‘	TRUE
>	greater than (1)	“xyz‘ > “abc‘	TRUE
>=	greater than or equal (1)	“abc‘>= “abc‘	TRUE
<>	not equal	5 <> 6	TRUE
=	equal	"xyz"= "xyz"	TRUE
ABS(*integer*)	absolute value	abs(-4)	4

ALPHA(*string*)	check if a string is alphabetic	alpha('abcd')
		alpha('ab3d ef')
			TRUE
			FALSE

ALPHANUM(*string*)	check if a string is only alphabetics and digits	alphanum('abCd')
		alphanum('45abd')
		alphanum('3d ef')
			TRUE
			TRUE
			FALSE

AND	logical and	7=7 and 5=5	TRUE
ANYPARM			
BAND	bitwise and	7 band 13	5

Symbol	Function	Example	Result
BASENAME (*string*)	returns the filename component	CALC basename (‘a.b.c’) CALC basename (‘/a/b/c’) CALC basename (’./a/b’) CALC basename (“./a.sl”,”.sl”) CALC basename (‘/’) CALC basename (“*feq”) CALC basename (‘$null’) CALC basename (‘abc.g’,’c’) CALC basename (/usr/lib/liby.a’,’.a’) CALC basename (‘/usr/lib/liby.a’,’liby.a’	A c b a / *FEQ $NULL AB liby liby.a
BNOT	bitwise not	bnot 5	-6
BOR	bitwise or	5 bor 2	7
BOUND(*varname*)	variable definition test (2)	bound(HPPATH)	TRUE
BXOR	bitwise exclusive or	7 bxor 5	2
CHR(*integer*)	ASCII value (integer) ===> character	chr(65)	A
CSL	circular shift left (3)	-2 csl 2	-5
CSR	circular shift right (3)	-7 csr 1	-4
DECIMAL(*string*)	returns a string value of an integer	CALC decimal (255) CALC len(decimal($ff)) setvar i 0 while setvar(i,i+1) < 10 and finfo(“FILE”+DECIMAL(I), ‘exists’) do ...	255 3, $3, %3

Symbol	Function	Example	Result
DELIMPOS (*str*,[,*delims*] [,*nth*][,*start*])	returns index in *str* of the *nth* delimiter beginning at *start*; default *delims* are a space, a comma, a semicolon, an equals sign, left and right parentheses, left and right brackets, single quote, double quote, and Tab; default *nth* is 1; default *start* is 1	DELIMPOS('file a=bb, old;rec=40,,f,ascii')	5
DIRNAME()(*string*)	returns dirctory components of a filename		
DWNS(*string*)	shift string to lowercase (7)	dwns('aBC&#dE')	abc&#de

EDIT(*string,editstr* [,*start*])	performs full REDO-like editing of a string	EDIT(`abcdefg','>dd')
		EDIT('ab cd;g', 'dwd')
			'abce'
			'cd;g'

FINFO(*filename, option*)	file information (6)	FINFO('x.pub',0)	TRUE

Symbol	Function	Example	Result
FQUALIFY(*string*)	returns a fully qualified filename	CALC fqualify(‘a’) CALC fqualify(‘a.b’) CALC fqualify(‘a.b.c’) CALC fqualify(‘./a’) CALC fqualify(‘./A’) CALC fqualify(‘/a/b/c’) CALC fqualify(‘*a’) CALC fqualify(‘$null’) CALC dirname (fqualify(‘./a’))	A.GROUP.ACCOUNT # when the CWD is your logon group or /CWD/A #when the CWD is s a directory A.B.ACCOUNT A.B.C /ACCOUNT/GROUP/a A.GROUP.ACCOUNT # when the CWD is your logon group or /CWD #when the CWD is a directory /a/b/c *A $NULL /ACCOUNT GROUP # when the CWD is your logon group or CWD # when the CWD is a directory

FSYNTAX()(*string*)	returns the syntax of the passed filename argument	fsyntax(‘a.b.c’)
		fsyntax(‘/a/b/c’)
		fsyntax(‘./ab@/c’)
		fsyntax($null’)
		fsyntax(‘a.b.c.d’)
			MPE
			POSIX
			POSIX;WILD
			MPE;$FILE MPE
			ERROR=426

HEX(*integer*)	convert to hexadecimal string	hex(329)	$149
INPUT([*prompt*] [,*wait*])	accept user input (10)	input('Enter choice:',20)	Enter choice: Y **Return** "Y"
LEN(*string*)	string length	len(“abc‘)	3
LFT(*string, # chars*)	left string extraction	lft('abc',2)	ab
LSL	logical shift left	7 lsl 1	14
LSR	logical shift right	-7 lsr 1	2,147,483,644
LTRIM(*string* [,*trimstr*])	trim left end of string (11)	'X'+ltrim(' abc') "X"+ltrim('...abc', '.')	Xabc Xabc
MAX(*num1*[,*num2...*])	find largest of several integers	max(5,4-3,70,0)	70
MIN(*num1*[,*num2...*])	find smallest of several integers	min(5,4,-3,70,0)	-3
MOD	modulo (4)	25 mod 2	1

Symbol	**Function**	**Example**	**Result**
NOT	logical not	not(2>1)	FALSE

NUMERIC (*string*)	check if a string is all digits	numeric('12345')
		numeric('$a234ef')
			TRUE
			FALSE

OCTAL(*integer*)	convert to octal string	octal(329)	%511

ODD(integer)	determine if integer is odd	odd(233)
		odd(-2)
			TRUE
			FALSE

OR	logical or	5=5 or 2=3	TRUE
ORD(*string*)	ordinal (8)	ord('AbcD')	65

POS(*find str,source str*[,*n*])	find *nth* occurrence of *find str* in *source str*; positive value for *n* begins search at left; negative value for *n* begins search at right) (12)	pos('ab','cgabd')
		pos('.','file.grp.acct',2)
		pos('.','file.grp.acct',-1)
			3
			9
			9

PMATCH(*pattern,str* [,*start*])	searches for *pattern* in a given string (*str*) starting at *start*; *pattern* may contain wildcards; default *start* is 1	PMATCH('f@','fread')
		PMATCH('abc','abcd')
			TRUE
			FALSE

Symbol	Function	Example	Result

REPL(*str,oldstr,newstr* [,*cnt*][,*start*])	in a given string (*str*), replaces *cnt* occurrences of *oldstr* with *newstr*, beginning at *start*; if *cnt* is positive, replacement begins at the left end of *str*; if negative, replacement begins at the right end of *str*; default *start* is 1; default *cnt* is zero (meaning all occurences)	REPL('aaabcaab','aa','X')
		REPL('aaabcaab','ab','' ,-1)
			'XabcXb'
			'aaabca'

RHT(*string, # chars*)	right string extraction	rht(“abc‘,2)	bc

RPT(*string,count*)	repeat a string (-count reverses string)	rpt('aBc',3)
		rpt('aBc',-3)
			aBcaBcaBc
			cBacBacBa

RTRIM(*string* [,*trimstr*])	trim right end of string (11)	rtrim('abc ')+'X' rtrim('abc...','.')+"X"	abcX abc X
SETVAR (*varname,expr*)	return result of *expr* and set *varname* to result (13)	setvar(*myvar*,2*3+5)	sets variable *myvar* to 11 and returns 11
STR(*string,start pos, # chars*)	general string extraction	str('abcde',2,3)	bcd
TYPEOF(*expression*)	type of variable or expression (5)	typeof(HPPATH)	2 (string)
UPS(*string*)	shift string to uppercase (7)	ups('aBc5d')	ABC5D

Symbol	Function	Example	Result

WORD(*string*,[,*delims*] [,*nth*][,*end_var*][,*start*])	performs general word extraction; default *delims* are a space, a comma, a semicolon, an equals sign, left and right parentheses, left and right brackets, single quote, double quote, and Tab; default *nth* is 1; the default *end_var* is no variable; the default *start* is 1	WORD('file a=bb,old; rec=40,,f,ascii')
		WORD('file a=bb,old; rec=40,,f,ascii',,-4,j)
			'file'
			'40', j=18

XOR	logical exclusive or	7=7 xor 5=5	TRUE

XWORDstring)	returns a string less ‘word’	xword(‘file a=bb, old; rec=40, ,f, ascii’)
		xword(‘file a=bb, old; rec=40, ,f, ascii’ , ,2)
		xword(‘file a=bb, old; rec=40, ,f, ascii’ , “ ; , “ , , j, 8)
		xword(‘file a=bb, old; rec=40, ,f, ascii’ , , -4, j)
			‘a=bb,old;rec=40, , f, ascii’
			‘file bb, old; rec=40 , , f, ascii’
			‘file a=old; rec=40 , ,f , ascii’ and J=10
			‘file a=bb, old;rec=, f, ascii ‘ and J=18

References

The following references apply to the numbers that appear in parentheses in table B-1.

1. Special rules apply when you use the comparison operators with strings. The strings are compared, character by character, until an inequality is found. This becomes the inequality of the strings. For example: 'ba' > 'abcd' and 'abcc' < 'abdc'. If string1 is longer than string2, and if string1 and string2 are equal up to the length of the string2, then string1 > string2 evaluates as TRUE.

2. The BOUND(*varname*) function returns the value TRUE if *varname* has been defined (assigned a value) and FALSE if it has not been defined. The BOUND function is defined as follows:

   - BOUND (name of a defined variable) = TRUE
   - BOUND (name of an undefined variable) = FALSE
   - BOUND (numeric value of expression) = TRUE
   - BOUND (string value of expressions) = TRUE
   - BOUND (Boolean value of expression) = TRUE

   For example

   ```
   setvar a 6
   calc bound(a)           TRUE
   deletevar a
   calc bound(A)           FALSE
   calc bound(1+2)         TRUE
   calc bound('a'+'b')     TRUE
   calc bound(5<4)      TRUE
   ```

   In BOUND (*expression*), if *expression* is not a valid expression, an error message is displayed

3. The circular shift operators, CSL and CSR, shift the specified number of bits in a 32-bit word in the specified direction. When 1 or 0 is shifted off one end, it comes back onto the other end. The logical shift operators, LSL and LSR, perform the same shifting as the circular shift operators, but when 1 or 0 is shifted off one end, a 0 comes back at the other end.

4. The modulo operation functions as it is defined by Donald E. Knuth, *The Art of Computer Programming,* Addison-Wesley Publishing Co., Reading, MA; Second ed., 1973; Volume I, p. 38.

5. The TYPEOF(*expression*) returns one of the following integer values:

   - 0 if expression is invalid.
   - 1 if expression evaluates to an integer.
   - 2 if expression evaluates to a string.
   - 3 if expression evaluates to a Boolean value.

6. The FINFO function returns a string, Boolean, or an integer value. The result depends upon the option specified.

   The first parameter, *filename,* is a string, the name of the file for which you want the information. This must be a fully or partly qualified file name, or a string expression that yields such a file name.

   This parameter can also be a string that specifies a file equation by backreference, for example, FINFO("*XIN', 1), which references the equation `FILE XIN=....`

   The second parameter, *option*, may be an integer (or integer expression) corresponding to the `FLABELINFO` intrinsic item numbers. Options 0 and 1 and the negative options are exceptions. The negative options provide the same information as their positive counterparts, except the format of the data is different.

   The option parameter may also be a string mnenomic which corresponds to an integer value. The string value is often easier to remember than the integer. Table B-2, which follows, summarizes all of the `FINFO` options.

   Users with system manager (SM) capabilities may use options 4 and 33 on any file within the system. Users with account manager (AM) capabilities may use those options only on files within their account.

7. The DWNS() and UPS() functions operate only on ASCII characters in the ranges "a" through "z" and "A" through "Z".

8. The ORD() and CHR() functions operate only on ASCII characters in the range 0 through 255.

9. 0^0 (zero to the zero power) yields 1.

10. The INPUT() function is different from other evaluator functions in that the execution of the command in which the function appears stops while input is taken from the user. The syntax is as follows:

    `INPUT([`*prompt*`][,`*wait*`])`

    INPUT reads from `$STDIN`. If a prompt is specified, it is written to `$STDLIST` before reading. If a wait is specified, the read is a timed read. The duration of the timed read is the lesser of wait seconds or the value of the `HPTIMEOUT` variable in minutes. The result of the read is returned as the string value of the function. If the user gave no input, but just pressed **Return**, the empty string is returned. If the timeout specified in the function itself (as opposed to the `HPTIMEOUT` timeout) expires, the empty string is returned. If the `HPTIMEOUT` timeout expires, the session is terminated.

NOTE This function should be used carefully since it interrupts execution of a command. It is not executed if it is skipped as a result of evaluation of a previous clause of a Boolean expression. This is the right side of an AND where the left is FALSE or the right side of an OR where the left is TRUE. For example:

```
IF "!filename" = " " AND SETVAR (filename,input &
('Enter filename:'))<>" " THEN
comment If filename is not empty, the left
```

```
                 comment side of the AND is FALSE and so
                 comment the right side is not executed.  This
                 comment means no INPUT() will be performed.
```

For LTRIM and RTRIM if *trimstr* is not given, then a space is used as the default

POS (*findstr,sourcestr*[,*N*]). If *N* is specified, the Nth occurrence of *findstr* is searched for in *sourcestr*. If *N* is negative, the ABS(*N*)th occurrence of *findstr* is searched for in *sourcestr* from the right. A value of zero for *N* results in a zero being returned. This is the same value which is returned if the requested occurrence of *findstr* is not found in *sourcestr*. For example:

```
POS('.','FILE.GRP.ACCT') WILL RETURN 5
POS('.','FILE.GRP.ACCT',-1) WILL RETURN 9
```

The SETVAR() function is different from other evaluator functions in that it is the first function that modifies its environment. The syntax is as follows:

SETVAR (*varname, expression*)

The *expression* is evaluated. If it evaluates with no errors, the value is returned and the variable with the name given as the first parameter is set to that value. Normal rules on setting variables apply: if it does not exist, it is created; if it does exist, its type is set to the type of the result of expression. Please refer to the SETVAR command for additional information. The Table B-2 shows the FINFO Specifications

NOTE The SETVAR() function is not executed in a partial evaluation skip state. See the INPUT() function above for an example.

Table B-2 FINFO Specifications

Number	Alias	Data Type	Item Description
0	EXIST	Boolean	Existence of file
1	FILENAME ONLY FNAME FULL FILENAME FULLFNAME FULLY QUALIFIED FILENAME	String	File name
2	GROUP GROUPNAME	String	Group name
3	ACCOUNT ACCT ACCOUNTNAME	String	Account name
4	CREATOR	String	File creator name
5	FMTSECURITY FORMATTED SECURITY MATRIX	String	Security matrix for access
-5	SECURITY MATRIX INTSECURITY	Integer	Security matrix for access
6	CREATED CREATION DATE FMTCREATED	String	File creation date
-6	CREATION DATE INTEGER INTCREATED	Integer	File creation date

Num ber	Alias	Data Type	Item Description
7	ACCESSED FMTACCESSED LAST ACCESS DATE	String	Last access date
-7	LAST ACCESS DATE INTEGER INTACCESSED	Integer	Last access date
8	MODIFIED LAST MOD DATE FMTMODDATE	String	Last modification date
-8	LAST MOD DATE INTEGER INTMODDATE	Integer	Last modification date
9	FILE CODE MNEMONIC FMTFCODE	String	File code of disk file
-9	FCODE INTFCODE FILE CODE	Integer	File code of disk file
10	USER LABELS WRITTEN	Integer	Number of user labels written
11	USER LABELS AVAIL	Integer	Number of user labels available
12	FILE LIMIT LIMIT	Integer	Total number of logical records possible in the file
13	FORMATTED FOPTIONS FMTFOPT	String	File options
-13	FOPTIONS INTFOPT	Integer	File options
14	RECORD SIZE RECSIZE	Integer	Record size
15	BLOCK SIZE BLKSIZE	Integer	Block size
16	MAX EXTENTS MAXEXT	Integer	Maximum number of extents
17	LAST EXTENT SIZE LASTEXTSIZE	Integer	Last extent size
18	EXTENT SIZE EXTSIZE	Integer	Extent size
19	END OF FILE EOF	Integer	Number of logical records in file
20	ALLOC TIME FMTALLOCTIME	String	File allocation time
-20	ALLOC TIME INTEGER INTALLOCTIME	Integer	File allocation time
21	ALLOC DATE FMTALLOCDATE ALLOCATED	String	File allocation date
-21	ALLOC DATE INTEGER INTALLOCDATE	Integer	File allocation date
22	NUM OPEN CLOSE RECS	Integer	Number of open/close records
23	DEVICE NAME DEV NAME	String	Device name (8 bytes

Number	Alias	Data Type	Item Description
24	FMTMODTIME LAST MOD TIME	String	Last modification time
-24	INTMODTIME LAST MOD TIME	Integer	Last modification time
25	FIRST USER LABEL	String	First user label (user label 0)
27	UNIQUE FILE ID UFID	String	Unique file identifier (UFID)
28	BYTE FILE SIZE BYTEFILESIZE	Integer	Total number of bytes allowed in file
29	BYTE DATA OFFSET DATASTART	Integer	Start of file offset
30	BYTE RECORD SIZE BYTERECSIZE	Integer	Record size (indicates bytes)
31	BYTE BLOCK SIZE BYTEBLKSIZE	Integer	Block size (indicates bytes)
32	BYTE EXTENT SIZE BYTEEXTSIZE	Integer	Extent size (indicates bytes)
33	LOCKWORD	String	File lockword
34	VOLUME RESTRICTION VOLRESTR	String	Volume restriction
35	VOLUME SET NAME	String	Volume set names
36	LOG SET ID	String	Transaction management log set id
37	LDEV LOGICAL DEVICE NUMBER	Integer	Logical device number
38	POSIX FULL FILE NAME POSIXFULLFNAME	String	Terminated HFS-syntax system absolute pathname
39	NUM HARD LINKS NUMHARDLINKS	Integer	The current number of hard links to the file
40	ACCESS TIME FMTACCESSTIME LAST ACCESS TIME	String	Time of last file access (clock format)
-40	LAST ACCESS TIME INTEGER INTACCESSTIME	Integer	Time of last file access (clock format)
41	STATUS CHANGE TIME FMTSTATUSCHANGETIME	String	Time of last file status change (clock format)
-41	INTSTATUSCHANGETIME CHANGE TIME INTEGER	Integer	Change Time Integer
42	STATUS CHANGE DATE FMTSTATUSCHANGEDATE	String	Date of the last file status change (calendar format)
-42	CHANGE DATE INTEGER INTSTATUSCHANGEDATE	Integer	Date of the last file status change (calendar format)

Number	Alias	Data Type	Item Description
43	FILE OWNER NAME OWNER	String	File owner
44	FILE OWNER ID UID	Integer	File owner identifier
45	FILE GROUP NAME FILEGROUP	String	File group
46	FILE GROUP ID GID	Integer	File group identifier
47	FILE TYPE FILETYPE	String	File type
-47	FILE TYPE INTEGER INTFILETYPE	Integer	File type
48	RECORD TYPE RECTYPE	Integer	Record type
49	BYTE FILE SIZE BYTEFILESIZE	Integer	Current file size (in bytes)
50	KSAM VERSION KSAMVERS	Integer	KSAM XL file version
51	KSAM LABEL KSAMPARAM	String	KSAM XL parameters
52	DEVICE TYPE DEVTYPE	String	MPE/iX device type
-52	DEVICE TYPE INTEGER INTDEVTYPE	Integer	MPE/iX device type
53	RELEASED	Boolean	Secured/Released
56	COMPRESSED	Boolean	Compressed/un-compressed (HSM)
57	MIGRATED	Boolean	Migrated/Not migrated (HSM)
58	SECTORS NUM SECTORS	Integer	Number of sectors occupied by the file
59	ESTENTS NUM EXTENTS	Integer	Number of extents occupied by the file
60	CREATETIME FMTCREATETIME INTEGER	String	File creation time (CLOCK format).
-60	INTCREATETIME CREATION TIME INTEGER	Integer	File creation time (CLOCK format).
61	ACCESSORS NUM ACCESSORS	Integer	Number of accessors of the file

Expression Evaluator Features

The two main types of expressions, which can be processed by the expression evaluator, are numeric and string. In addition, Boolean expressions may be constructed using numeric and string expressions (involving the comparison operators), Boolean operators, Boolean functions, and Boolean variables.

A numeric expression may contain the following:

- Variables containing numeric values or expressions
- Unary operators: +, -
- Bit manipulation operators: CSL, CSR, LSL, LSR, BOR, BAND, BNOT, BXOR
- Exponentiation operator: ^
- Algebraic operators: +, -,
- , /, MOD
- Comparison operators: >, <, =, >=, <=, <>
- Parentheses: ( )
- Functions returning numeric values: ABS(), LEN(), ORD(), POS()
- Decimal digits, optionally preceded by #, consisting of 0..9
- Hexadecimal digits, preceded by $, consisting of 0 .. 9, a .. f, A .. F
- Octal digits, preceded by %, consisting of 0 .. 7

String expressions are comprised of the following:

- Variables containing string values or expressions
- Algebraic operators: +, -
- Comparison operators: >, <, =, >=, <=, <>
- Parentheses: ( )
- Functions returning string values: STR(), LFT(), RHT(), CHR()
- Quoted strings of the form 'string', "string", '', or "" (the last two refer to an empty string)

String and numeric expressions resulting in Boolean values may be combined with each other and with Boolean functions and variables using the logical operators AND, OR, XOR, and NOT. Their Boolean values may also be compared with the equality, =, and inequality, <>, operators. For example:

```
setvar a 1
setvar str2 'b'
setvar boolvar1 1=0
if (a=1)=('a'=str2) or boolvar1 then
EXPRESSION FALSE
endif
```

Functions may be nested and mixed. String, numeric, and Boolean operations may not, however, be mixed. For example:

```
calc 1+'abc'
    ERROR
calc 'a' +len('abc')
    ERROR

if bound(a) + 3>2 then
    ERROR

setvar bool1 true
if bool1 and (str('abc',2,len('ab'))='bc') then
EXPRESSION IS TRUE
endif

calc 'a'+chr(65)
aA
calc 1=3 or 'a'  <>  'b'
TRUE

calc chr(ord('A'))
A
calc 2+len(str(lft('abcdefg',2*2),5-3,ord('A')-63))
4, $4, %4
```

Variables may be used in expressions either through explicit dereferencing or implicit dereferencing. To explicitly dereference a variable, precede the variable name with an exclamation point (!). This is passed through string substitution the same as any other CI command. Explicit dereferencing is recursive, meaning that if the contents of the variable references another variable (introduced with an exclamation point) the value of the included variable is also retrieved.

To implicitly dereference a variable, simply use its name in any expression. If a variable with this name has not been defined an error results. Implicit dereferencing is not recursive. This means that if the contents of an implicitly dereferenced variable contains a string which might be a variable name, preceded by an exclamation point the evaluator does not attempt to dereference that variable. Instead, the string value is used in the expression.

For example:

```
setvar a 'x'
showvar a
A = x
calc a+'b'
xb

setvar a ''
showvar a
A =
if a = '' then
EXPRESSION IS TRUE
endif

setvar a 'x'
setvar b a
calc a+b
xx

setvar exp 'a+b*c/d'
setvar a  1
setvar b  2
setvar c  3
setvar d  4
setvar e  5
calc exp
```

```
a+b*c/d
calc !exp
2, $2, %2
setvar exp2 exp+'*e'
calc !exp2
6,$6,%6

setvar a hptimef
showvar a
A = 8:26 AM              ** the time when var was set **
calc a + 'in the morning!!! '
8:26 AM in the morning!!!'
calc 'a' + 'in the morning!!!'
a in the morning!!!'
calc '!a' + 'in the morning!!!
8:26 AM in the morning!!!'

deletevar a
calc a + 'x'
    ERROR

setvar hppath '!!hpgroup,pub,pub.sys'
showvar hppath
HPPATH = !hpgroup,pub,pub.sys
calc hppath - ',pub'
!hpgroup,pub.sys
calc !hppath-',pub'
calc UI,pub,pub.sys-',pub'
     ^
ERROR
comment         ** variable dereferenced before call **
comment         ** to evaluator and content does not **
comment         ** make valid expression.            **
calc '!hppath' - ',pub'
UI,pub.sys

setvar a 6+2
setvar b 7
setvar c b
calc b*c
49, $31, %61
calc hpresult/a
6, $6, %6

setvar a '2'
setvar b 6
calc a+b
    ERROR       ** variables of different types      **
calc len(b)
    ERROR       ** expected string or string variable**
calc ord(a)
50, $32, %62

setvar a -6
calc 18/(3^2^3/3^6/3+6)/-(a+3)-1
-1, $FFFFFFFF, %37777777777
```

The rules of precedence determine which operations are performed before others. Their order, from highest to lowest priority, is:

- Variable dereferencing
- Unary operators: + -
- Bit manipulation: CSL, CSR, LSL, LSR, BOR, BAND, BNOT, BXOR
- Exponentiation
- Multiplication, division, modulo
- Addition, subtraction
- Comparison: > < = + <= <>

- Logical operations: AND, NOT
- Logical operations: OR, XOR

The evaluation of string expressions follows a similar hierarchy. However, bit manipulation, multiplication, division, and modulo operations do not apply to string expressions. If you attempt to use them with a string expression, an error occurs.

Evaluation is left to right until the evaluation is complete, or until a fatal error has been detected. If a fatal error is detected, evaluation terminates.

Completion of evaluation in this case means either end of expression or partial evaluation of expression.

In the latter case (partial evaluation), the result of the evaluation can be determined without examining the rest of the expressions. For example, when part of an expression that is evaluated to FALSE is followed by an AND, or is evaluated to TRUE and is followed by an OR:

```
(1=2) And (2=2 or 3=4)
FALSE And (whatever) -> FALSE

(1=1) or (2=3 and x=y)
TRUE or (whatever) -> TRUE
```

NOTE Exponentiation is the one exception to the left-to-right evaluation pattern. Exponentiation evaluates right to left. For example, 3^2^3 is resolved as 3^8 (=6561) and not as 9^3 (=729).

The logical operators operate only on Boolean expressions, Boolean functions, or Boolean variables. Boolean expressions are those which contain a comparison operation (< > <= >= <> =) or a logical operation (AND, OR, NOT, XOR).

Examples:

```
if 6-5>2 and 'abc'-'a'<=rht('cdbc',2) then
     EXPRESSION IS FALSE
endif

if not(1=1 and 'a'<>'b') or 6>7 then
     EXPRESSION IS FALSE
endif

calc 6+(7>2)
     ERROR   ** Invalid Expression:       **
             ** Mixed Numeric and Boolean **

if 1 then
     ERROR   ** Bad Boolean Expression    **

setvar errorflag true
if errorflag then
     EXPRESSION IS TRUE
endif
```

The expression evaluator is sensitive to the position of expression tokens. If an operator is expected, then an operator must be obtained in that position or a fatal error occurs. If a number is expected and a valid numeric string is not found, variable management is called to determine if this token is actually a variable. If the token is a variable with a numeric value, the variable value is used in the expression. If the token is not a variable or the variable is not an integer variable, the expression is not valid and an error is returned.

If a string is expected and a valid quoted string is not found, variable management is called to determine if the token is a variable. If it is not a variable, an error is returned. If it is a variable containing a string value, its contents is used in the expression. If the variable contains something other than a string, an error is returned.

Provided below is information on other facts you should be aware of concerning evaluator functions.

Ord

If the length of *string*> 1, then the value returned from ORD(*string*) is the ordinal value of the first character in the *string*.

Strings

A “string‘ of characters must be surrounded with quotation marks (“ or ‘) in order to be treated as a string. For example, `a + 'a'` is treated as the contents of the string variable `a` concatenated to the string `'a'` .

Evaluating a string that contains a string operator returns an error unless the string itself is surrounded by quotation marks (“ or ‘). You may include quotation marks within a string in this fashion: `“a‘b“` is evaluated as `a‘b`, but `a‘b` by itself produces an error.

You may also use quote folding, for example, two adjacent quotes of the same type that began the string. They are folded to one and the string is not terminated, for example:

```
setvar a "a quote is here""!"
```

This would put the string `a quote is here"!` into the string variable A.

Variables

Variables that are dereferenced by an `!` are dereferenced to complete resolution or to the limits of dereferencing (default is 30 levels). Variables may be used in expressions without the `!`, of course. This is called implicit dereferencing, and these variables are dereferenced to only one level.

For example, if variable `A` has a value of `B`, it is implicitly and explicitly dereferenced as `B`. If this variable has a value of `!B`, implicit dereferencing yields `!B`. If you want `A` to be fully dereferenced, you must use `!A` (explicit dereferencing) in the expression you want evaluated.

Variables and Strings

Explicitly dereferenced variables should be placed within quotation marks if you want the variable's value treated as a string. Doing this also eliminates problems that might arise if the variable contains delimiters or operators. Refer to the discussion on "Strings" above. For example:

```
SETVAR X 3
CALC “AB‘ + “!X‘
AB3
CALC “AB‘ + X
  error
CALC “AB“ + “X‘
```

```
ABX

SETVAR Z “foo‘
CALC “AB‘ + Z
ABfoo

CALC “AB‘ + “!Z‘
ABfoo
CALC “AB‘ + !Z
  error  variable foo not found

SETVAR A “X‘+“Y‘
CALC A +“B‘
XYB
CALC “!HPTIMEF‘
8:26 AM
CALC HPTIMEF
8:26 AM
CALC !HPTIMEF
  error
```

The error in the last example occurs because the dereferenced value of HPTIMEF is not a valid expression.

Dereferencing of either kind is performed before any evaluation is carried out. The following examples illustrate the consequences:

```
SETVAR B 2
SETVAR A B
  error
```

The first command causes no problem. A variable, B, is created and its value is set to 2. Because 2 is not surrounded by quotes, it is taken as an integer.

String Substitution assumes that an exclamation point introduces a variable name. However, there are occasions when the user wants String Substitution to ignore an exclamation point. Doubling the exclamation point will cause String Substitution to reduce the two exclamation points to one, and ignore them as dereferencing characters.

Dereferencing takes place first, and B yields !B For additional information on variables and dereferencing, refer to the *Using the 900 Series HP 3000 Fundamental Skills* (32650-60039). Because B is not surrounded by quotes, it is not taken as a string, integer, or a Boolean. Therefore, SETVAR A B produces an error.

The problem is corrected by changing the second command:

```
SETVAR B 2
SETVAR A “B‘   ** second command changed **
```

Now the variable A is given the *string* representation of !B.

Consequently,

```
SHOWVAR B
B=2
SHOWVAR A
A=!B
```

But,

```
CALC A + 2
```

produces an error. A has been assigned a string value (the result of B, which is the string “!B‘).

However,

```
CALC !A + 2
```

works. !A is really !B. That in turn yields a value of 2. The result is 2 + 2, which equals 4.

```
CALC “HPTIMEF‘
HPTIMEF
```

But,

```
CALC !HPTIMEF

CALC 8:26 AM
```

which produces an error.

On the other hand,

```
CALC “!HPTIMEF‘
```

is the same as

```
CALC “8:26 AM‘
```

which produces

```
8:26 AM
```

C Terminal and Printer Types

The terminal types supported on the advanced terminal processor (ATP) and asynchronous data communications controller (ADCC) terminal/printer controllers for MPE V/E T-MIT or later (MPE V/E version G.01.00 or later) are 6, 9, 10, 12, 13, 15, 16, 18, 19, 20, 21, TTPCL18, TTPCL19, and TTPCL22

The data and terminal subsystems (DTS) on MPE/iX systems supports terminal types 10 and 18.

The DTS on MPE/iX systems supports printer types 18, 21, and 22.

Table C-1 through Table C-4provide comparative information for MPE V/E and MPE/iX terminal and printer types.

Table C-1 MPE/iX Terminal Types and Similar MPE V/E Terminal Types

MPE/iX Terminal	MPE V/E Terminal	Differences
10	10	Enhanced XON/OFF protocol on MPE/iX. No ENQ/ACK protocol on MPE/iX.
18	18	Enhanced XON/OFF protocol on MPE/iX.

Table C-2 MPE/iX Printer Types and Similar MPE V/E Terminal Types

MPE/iX Printer	MPE V/E Terminal	Differences
18	18	Enhanced XON/OFF protocol on MPE/iX.
21,22	21,22	Enhanced XON/OFF protocol on MPE/iX. No ENQ/ACK protocol on MPE/iX. Printer initialization string of the MPE V/E terminal type PCL22 is used. Printer status checking is done less frequently on MPE/iX ; status requests are not sent after each printed line.

Table C-3 MPE V/E Terminal and Similar MPE/iX Terminal Types

MPE V/E\Terminal	Description	MPE/iX \Terminal	Comments
6,9	Non-HP hardcopy device needing delays after linefeed or formfeed.	None	Devices that need delays are not supported on MPE/iX.
10	General HP CRT terminal using both ENQ/ACK and XON/XOFF.	10	Only XON/XOFF protocol is used on MPE/iX.
12	8-bit character version of terminal type 10. Used with languages that need extended character sets.	10	It is necessary to programmatically set parity to NONE.
13	Terminal type 10 with no echo or ENQ/ACK protocol. Used for plotters or the HP 2601 printer.	None	These devices are not supported on MPE XL.
15,16	8-bit and 7-bit HP 2635 hardcopy terminal.	None	The HP 2635 is not supported on MPE/iX.
18 (Terminal)	Terminal type 10 without ENQ/ACK protocol or a READ trigger. Used with non-HP devices.	18	Terminal type 18 is the same on MPE V/E and MPE/iX , except that an enhanced XON/XOFF protocol is used on MPE/iX.
18 (Printer)	Non-HP devices or application printers.	18	The MPE V/E terminal type 18 and MPE/iX printer type 18 are the same except that an enhanced XON/XOFF protocol is used on MPE/iX.

Table C-4 MPE V/E Terminal and Similar MPE/iX Printer Types

MPE V/E\Terminal	Description	MPE/iX \Printer	Comments
Terminal Type PCL18	Terminal type 18 with a printer initialization string and an XOFF timer. Used with the HP 2687A.	None	The HP 2678A is not supported on MPE/iX.
19	Remote serial spooled printer.	21	Status checking is done less frequently on MPE/iX. The printer initialization string is the same as MPE V/E terminal type PCL22.
Terminal Type PCL19	PCL Remote serial spooled printer.	21	Status checking is done less frequently on MPE/iX.

MPE V/E\Terminal	Description	MPE/iX \Printer	Comments
20	8-bit Serial spooled printer.	22	Status checking is done less frequently on MPE/iX. The Initialization string is the same as MPE V/E terminal type PCL22.
21	8-bit Serial spooled printer with no status checking after XOFF.	21	Status checking is done less frequently on MPE/iX. Initialization string is the same on MPE V/E terminal type PCL22.
22	Serial spooled printer with no status checking after XOFF.	22	Status checking is done less frequently on MPE/iX. Initialization string is the same as MPE V/E terminal type PCL22.
Terminal Type PCL22	PCL 8-bit serial spooled printer with no status checking after XOFF.	22	Status checking is done less frequently on MPE/iX.

D Subsystem Formal File Designators

Table D-1 lists the formal file designator associated with specific command parameters.

Table D-1 Formal File Designators

Command	Parameter	Formal File Designator
BASIC	*commandfile*	BASCOM
	inputfile	BASIN
	listfile	BASLIST
BASICOMP	*textfile*	BSCTEXT
	uslfile	BSCUSL
	listfile	BSCLIST
BASICGO	*textfile*	BSCTEXT
BASICPREP	*listfile*	BSCLIST
BBASIC	*commandfile*	BASCOM
	inputfile	BASIN
	listfile	BASOUT
BBASICOMP	*infile*	BBCIN
	uslfile	BBSCUSL
	listfile	BBSCLIST
BBASICGO	*infile*	BBCIN
BBASICPREP	*listfile*	BBCLIST
BBXL	*commandfile*	BASCOM
	inputfile	BASIN
	listfile	BASOUT
BBXLCOMP	*textfile*	BBCIN
	objectfile	BBCOBJ
	listfile	BBCLIST
BBXLGO	*textfile*	BBCIN
BBXLLK	*listfile*	BBCLIST

Command	**Parameter**	**Formal File Designator**
CCXL	*textfile* *objectfile* *listfile*	CCTEXT CCOBJ CCLIST
CCXLGO CCXLLK	*textfile* *listfile*	CCTEXT CCLIST
COB74XL COB85XL	*textfile* *objectfile* *listfile* *masterfile* *newfile* *workspacename* *xdbfilename*	COBTEXT COBOBJ COBLIST COBMAST COBNEW COBWKSP COBXDB
COB74XLG COB74XLK COB85XLG COB85XLK	*textfile* *listfile* *masterfile* *newfile* *workspacename* *xdbfilename*	COBTEXT COBLIST COBMAST COBNEW COBWKSP COBXDB
COBOLII	*textfile* *uslfile* *listfile* *masterfile* *newfile* *workspacename*	COBTEXT COBUSL COBLIST COBMAST COBNEW COBWKSP
COBOLIIGO COBOLIIPREP	*textfile* *listfile* *masterfile* *newfile* *workspacename*	COBTEXT COBLIST COBMAST COBNEW COBWKSP
EDITOR	*listfile*	EDTLIST

Command	Parameter	Formal File Designator
FORTRAN	*textfile*	FTNTEXT
	uslfile	FTNUSL
	listfile	FTNLIST
	masterfile	FTNMAST
	newfile	FTNNEW
FORTGO	*textfile*	FTNTEXT
FORTPREP	*listfile*	FTNLIST
	masterfile	FTNMAST
	newfile	FTNNEW
FTN	*textfile*	FTNTEXT
	uslfile	FTNUSL
	listfile	FTNLIST
FTNGO	*textfile*	FTNTEXT
FTNXLGO	*listfile*	FTNLIST
FTNPREP	*textfile*	FTNTEXT
	progfile	FTNLIST
	listfile	FTNPROG
FTNXL	*textfile*	FTNTEXT
	objectfile	FTNOBJ
	listfile	FTNLIST
FTNXLLK	*textfile*	FTNTEXT
	progfile	FTNOBJ
	listfile	FTNLIST
PASCAL	*textfile*	PASTEXT
	uslfile	PASUSL
	listfile	PASLIST
PASCALGO	*textfile*	PASTEXT
PASCALPREP	*listfile*	PASLIST
PASXL	*textfile*	PASTEXT
	objectfile	PASOBJ

Command	Parameter	Formal File Designator
	listfile	PASLIST
	libfile	PASLIB
PASXLGO	*textfile*	PASTEXT
PASXLLK	*listfile*	PASLIST
	libfile	PASLIB
PREP	PMAP	SEGLIST
PREPRUN	LMAP	LOADLIST
RESTORE	SHOW	SYSLIST
RPG	*textfile*	RPGTEXT
	uslfile	RPGUSL
	listfile	RPGLIST
	masterfile	RPGMAST
	newfile	RPGNEW
RPGGO	*textfile*	RPGTEXT
RPGPREP	*listfile*	RPGLIST
	masterfile	RPGMAST
	newfile	RPGNEW
RPGXL	*textfile*	RPGTEXT
	objectfile	RPGOBJ
	listfile	RPGLIST
RPGXLGO	*textfile*	RPGTEXT
RPGXLLK	*listfile*	RPGLIST
SEGMENTER	*listfile*	SEGLIST
SPL	*textfile*	SPLTEXT
	uslfile	SPLUSL
	listfile	SPLLIST
	masterfile	SPLMAST
	newfile	SPLNEW
SPLGO	*textfile*	SPLTEXT
SPLPREP	*listfile*	SPLLIST

Command	Parameter	Formal File Designator
	masterfile *newfile*	SPLMAST SPLNEW
STORE RESTORE	SHOW	SYSLIST
SYSGEN	*inputfile* *outputfile*	SYSGIN SYSGOUT

E MPE/iX File Codes

File codes are recorded in the file label and are 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 the mnemonic name for this parameter, certain reserved integers and mnemonics have particular system-defined meanings. defines the MPE/iX reserved integer and mnemonic values Table E-1

.

Table E-1 File Codes

Integer	Mnemonic	Meaning
1024	USL	User Subprogram Library
1025	BASD	Basic Data
1026	BASP	Basic Program
1027	BASFP	Basic Fast Program
1028	RL	Compatibility Mode Relocatable Library
1029	PROG	Compatibility Mode Program File
1031	SL	Segmented Library
1035	VFORM	VPLUS Forms File
1036	VFAST	VPLUS Fast Forms File
1037	VREF	VPLUS 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 Work File

Integer	Mnemonic	Meaning
1059	TDPXQ	TDP Work File (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 Log File
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
1112	PENV	IFS/3000 Environment File
1113	PCCMP	IFS/3000 Compiled Character Cell File
1114	RASTR	Graphics Image in RASTER 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
Integer	Mnemonic	Meaning
1149	D48	Reserved
1152	SLATE	Compressed SLATE File
1153	SLATW	Expanded SLATE Work File

Integer	Mnemonic	Meaning
1156	DSTOR	RAPID/3000 DICTDBU Utility Store File
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 Messages File
1169	MARPD	ARPA Distribution List
1170	MCMND	HPDESK Abbreviated Commands File
1171	MFRTM	HPDESK Diary Free Time List
1172	None	Reserved
1173	MEFT	HPDESK External File Transfer Messages File
1174	MCRPT	HPDESK Encrypted Item
1175	MSERL	HPDESK Serialized (Composite) Item
1176	VCSF	Version Control System File
1177	TTYPE	Terminal Type File
1178	TVFC	Terminal Vertical Format Control File
1192	NCONF	Network Configuration File
1193	NTRAC	Network Trace File
1194	NTLOG	Network Log File
1195	MIDAS	Reserved
1211	NDIR	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

Integer	Mnemonic	Meaning
1228	LANGD	Language Definition File
1229	CHARD	Character Set Definition File
1230	MGCAT	Formatted Application Message Catalog
1236	BMAP	Base Map Specification File
1242	BDATA	HP Business BASIC/V Data File
1243	BFORM	HP Business BASIC/V Field Order File for VPLUS
1244	BSAVE	HP Business BASIC/V SAVE Program File
1245	BCNFG	Configuration File for Default Options for HP Business BASIC Programs
1246	BKEY	Function Key Definition File for Terminal
1258	PFSTA	Pathflow STATIC File
1259	PFDYN	Pathflow Dynamic File
1270	RFDCA	Revisable Form DCA Data Stream
1271	FFDCA	Final Form DCA Data Stream
1272	DIU	Document Interchange Unit File
1273	PDOC	HPWORD/150 Document
1275	DFI	DISOSS Filing Information File
1276	SRI	Search Restart Information File
1401	CWPTX	Chinese Word Processor Text File
1421	MAP	HPMAP/3000 Map Specification File
1422	GAL	Reserved
1425	TTX	Reserved
1428	RDIL	HP Business Report Writer (BRW) Dictionary File CM
1429	RSPEC	BRW Specification File
1430	RSPCF	BRW Specification File
1431	REXCL	BRW Execution File
1432	RJOB	BRW Report 509 File
1433	ROUT1	BRW Intermediate Report File
1434	ROUTD	BRW Dictionary Output

Integer	Mnemonic	Meaning
1435	PRINT	BRW Print File
1436	RCONF	BRW Configuration File
1437	RDICN	BRW NM Dictionary File
1438	REXNUM	BRW NM Execution File
1441	PIF	Reserved
1476	TIFF	Tag Image File Format
1477	RDF	Revisable Document Format
1478	SOF	Serial Object File
1479	GPF	Chart File for Charting Gallery Chart
1480	GPD	Data File for Charting Gallery Chart
1483	VCGPM	Virtuoso Core Generator Processed Macro File
1484	FRMAT	Formatter
1485	DUMP	Dump Files Created and Used by IDAT and DPAN
1486	NNMD0	New Wave Mail Distribution List
1491	X4HDR	X.400 Header for HPDesk Manager
1500	WP1	Reserved
1501	WP2	Reserved
1502	LO123	Lotus 123 Spread Sheet
1514	FPCF	Form Tester Command Spec File
1515	INSP	Spooler XL Input Spoolfile
1516	OUTSP	Spooler XL Output Spoolfile
1517	CHKSP	Spooler XL Checkpoint Spoolfile
1521	DSKIT	HPDesk Intrinsics Transaction File
1526	MSACK	Man Server Acknowledgement
1527	MSNON	Man Server Non-Delivery Notification
1528	MSTRC	Man Server Trace File
3333		Reserved

NOTE Default is the unreserved file code of 0.

Using 1090 (LOG) as a designated file code 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.

F Wildcard Characters

In some commands, you may substitute wildcard characters for certain parameters, or parts of parameters, in the list. The wildcard characters count toward the eight character limit for user, group, account, and file names. These wildcard characters are defined in Table F-1.

Table F-1 Table F-1. Wildcard Character Definitions

Character	Function
@	Specifies zero or more characters. When used by itself, @ denotes all possible members of the set.
#	Specifies one numeric character.
?	Specifies one 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 as many as seven digits, where each digit is represented by a single number sign (#).
=?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".
[ ]	A range of characters (only with the `LISTFILE`, `SHOWVAR`, `DELETEVAR`, `STORE`, and `RESTORE` commands).

The `LISTFILE`, `SHOWVAR`, `DELETEVAR`, `STORE`, and `RESTORE` commands provide for a range or set of characters. Refer to chapter 2 of this manual for additional information on these commands.

Index

Index

Index

Index

Index

Index