RSTS/E
System User's Guide

Order No. AA-EZ12A-TC

digital software

RSTS/E
System User's Guide

Order No. AA-EZ12A-TC

June 1985

This manual contains general information about RSTS/E, provides rules for using DCL (DIGITAL Command Language) on RSTS/E, and describes commands for file, system and programming operations.

OPERATING SYSTEM AND VERSION:	RSTS/E	V9.0
SOFTWARE VERSION:	RSTS/E	V9.0

digital equipment corporation, maynard, massachusetts

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

The postage-paid READER'S COMMENTS form on the last page of this document requests your critical evaluation to assist us in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

digital™	DIBOL	ReGIS
	FMS-11	RSTS
DEC	LA	RSX
DECmail	MASSBUS	RT
DECmate	PDP	UNIBUS
DECnet	P/OS	VAX
DECtape	Professional	VMS
DECUS	Q-BUS	VT
DECwriter	Rainbow	Work Processor

Contents

4 System and Account Operations

5 Terminal Status and Operations

6 Working with Devices

Tables

Objectives

This manual describes the RSTS/E operating system and the use of the DIGITAL Command Language (DCL) on RSTS/E systems.

This manual describes commands for operations such as working with files, getting information about the system and its devices, printing files and running batch jobs, and developing programs. Note that this manual does not describe commands which require privileges for you to use them; see the *RSTS/E System Manager's Guide* for descriptions of privileged DCL commands.

Audience

Although some familiarity with computers is helpful, you do not need to be an experienced computer user or programmer to use this manual.

Document Structure

The manual is divided into eight chapters and two appendixes:

Chapter 1	Describes the basics of the RSTS/E system and how to use it. It includes descriptions of the command environments, running a program, and using DECnet/E. (DECnet/E allows you to communicate with other computers.)
Chapter 2	Describes how to use commands interactively and introduces how to use command procedures.
Chapter 3	Describes the RSTS/E commands used to create, display, edit, copy, and compare files.
Chapter 4	Describes how to use DCL in day-to-day RSTS/E operations, such as displaying system status and changing your password.
Chapter 5	Describes how to display and set your terminal characteristics, and how to enable and disable terminal logging.
Chapter 6	Describes the devices available for your use on a RSTS/E system.
Chapter 7	Describes the commands in the RSTS/E Print/Batch Services (PBS) facility, used to submit files for print or batch processing.

Chapter 8	Describes the commands available for you to develop and run programs.
Appendix A	Lists error messages that RSTS/E displays when you enter a command that RSTS/E cannot execute.
Appendix B	Contains information about the BASIC-PLUS, RSX, and RT11 command environments. It also contains information about RSTS/E file specification.

The manual also contains a glossary of RSTS/E and DCL terms.

Related Documents

See the *RSTS/E Guide to Writing Command Procedures* for more information on command procedures and their use.

For complete details on how to define and control SORT and MERGE operations, see the *PDP-11 SORT/MERGE User's Guide.*

To get started with BASIC-PLUS-2 programming, read the *Introduction to BASIC.*

See the *RSTS/E Documentation Directory* for a description of the other manuals in the RSTS/E documentation set.

Conventions

This manual uses the following symbols and conventions:

[ ]	Square brackets show the optional parts of a command in format statements. For example: DIRECTORY [file-spec[,...]] The square brackets in this example indicate that you can include a file specification ([file-spec]), or more than one ([,...]), if you choose. Square brackets also indicate the choice you have in using a command. For example, /[NO]DELETE This means you can type either /DELETE or /NODELETE, depending on the form of the qualifier you select. Do not confuse the square brackets in command formats with the square brackets in Project-Programmer numbers (PPNs), as in [52,20]. See Chapter 1 for a description.
< >	Angle brackets surrounding text indicate a place holder for what the system inserts when an error message occurs.

CTRL/x	The control key, which you use in combination with another key. For example, enter CTRL/U by holding down the CTRL key and pressing the keyboard key labeled "U." RSTS/E displays, or echoes, CTRL/U at your terminal, as ^U.
RET	The key labeled RETURN on your terminal. You press the RETURN key to complete lines and commands that the system will process.
`dot matrix color`	Information in this typeface indicates an example of computer/user dialogue. In examples, black characters are data produced by the computer. Red characters indicate information that you type.

Summary of Technical Changes

RSTS/E V9.0 is a major release of the RSTS/E PDP-11 operating system. This manual contains support for the following new features:

- Privileges
- Login command files
- Command procedures
- Relative date and time formats
- New file manipulation commands: SORT, MERGE, SET FILE
- New SHOW commands for system and account status
- New SET PASSWORD command for changing your password
- New qualifiers for SET TERMINAL
- Terminal logging feature
- New SHOW commands for devices
- Print/Batch Services (PBS): new facility for print and batch processing

Getting Started With RSTS/E 1

This chapter introduces you to RSTS/E and contains information to help you become familiar with using the system.

Using Your Account

In order for you to use the computer, your system manager must assign you an account number and password. The account number identifies you to the system. The password is a precaution to keep uninvited users from accessing the system.

When you log in, the system displays a message at your terminal to confirm that you are logged in to your account. The system may also display informational messages.

Privileges

After you log in, you have access to all the RSTS/E facilities that your account allows. Your system manager decides what privileges to give your account.

Like most users of your system, you will probably have sufficient privileges to create, modify, copy, and delete your own files, as well as to change your own password. Other users of your system may be granted additional privileges, allowing them to perform operations such as:

- Creating and deleting accounts
- Adding commands to the system
- Controlling the number of users on the system
- Saving user files in case of accidental damage
- Shutting down the system
- Reading and modifying other users' files

This manual describes only those commands, command qualifiers, and functions that can be used by most system users, that is, users who do not have privilege to affect

accounts other than their own, or to affect the system itself. See the *RSTS/E System Manager's Guide* for detailed descriptions of commands, qualifiers, and functions requiring additional privileges.

Your Project-Programmer Number

Account numbers on RSTS/E are called project-programmer numbers, or PPNs. Your PPN corresponds to your account and identifies you as a system user.

A PPN consists of two numbers enclosed in square brackets and separated by a comma, for example, [52,20]. The numbers are:

- A project number (the number 52 in [52,20]), which corresponds to a group of users. For example, all employees in a particular work group may be assigned the same project number.
- A programmer number (the number 20 in [52,20]), which is unique for each user in the group.

The brackets around the PPN show that the two numbers are used together. Do not type the brackets when you log in, although you will need to use the brackets in many operations dealing with files. Also, do not confuse brackets around a PPN with brackets used to indicate optional parts of a command (see "Conventions" in the Preface).

If you have more than one account on the system, you must use different PPNs and passwords to log in to your different accounts.

Your Password

Your password protects your account from unauthorized use. Passwords on RSTS/E systems can consist of 6 to 14 alphanumeric characters. (The system manager may restrict some users' passwords to a maximum length of 6 characters, while other users may use up to 14 characters.)

Your system manager may assign you a password or let you choose your own. The method of assigning a password depends on the policy at your site. However, most users are granted sufficient privilege to change their own passwords at will. (Frequent password changes help to protect the system from unauthorized users.)

To minimize the chances that unauthorized users can discover your password, DIGITAL recommends the following guidelines:

- Do not use names or words that could be readily associated with any user (for example, WRITER, GUEST, or your first name).
- Choose a password that contains both letters and digits.
- Change your password at least once a month.

See Chapter 4 for details on how to use the SET PASSWORD command to change your password.

Using the Terminal (CTRL, DELETE, RETURN Keys)

You use a terminal to communicate with the system. Terminals can be grouped into two main categories: hard-copy and video. A hard-copy terminal, such as an LA120, prints the characters you type and data from the system onto lineprinter paper. A video terminal, such as a VT100 or VT220, displays characters on a Cathode Ray Tube (CRT) screen.

Most of the keys on a terminal match keys on a typewriter keyboard. However, several keys on your terminal's keyboard have special functions. Three of the most commonly used function keys are labeled CTRL, DELETE (also known as RUBOUT), and RETURN.

The key labeled CTRL is called the control key. You always use the control key in combination with another key. For example, you can use CTRL/U to delete all the characters on a command line if you have not yet pressed the RETURN key. Enter CTRL/U by holding down the control key and then pressing the "U" key on the keyboard. RSTS/E displays, or "echoes," this at your terminal as ^U. For example:

```
$ EDIT/EDT RACER.PDQ (CTRL/U)
```

On your terminal, this looks like:

```
$ EDIT/EDT RACER.PDQ ^U
```

The line is now deleted. Press the RETURN key for the DCL prompt:

```
(RET)
$
```

Use the DELETE key to erase one letter at a time, or hold the key down to erase words and lines. The DELETE key has the same function on a hard-copy terminal and on a video terminal, but the effects look different. (On some hard-copy terminals, the key labeled RUBOUT performs the delete function.) For example, suppose you misspell a command and use the DELETE key twice:

```
$ COPIE (DEL)(DEL)
```

On a video terminal, the resulting line reads:

```
$ COP
```

By contrast, a hard-copy terminal displays deleted characters between backslashes (\), because a hard-copy terminal is not designed to "erase" characters. For example:

```
$ COPIE\EI\
```

The RETURN key allows you to:

- End commands — When you enter the name of a command, pressing the RETURN key tells the system to execute the command. For example:

```
$ DIRECTORY (RET)
```

- End lines — When you enter text in a file, the RETURN key ends each line and begins a new one. For example:

```
Each year about 500 (RET)
runners enter Littleton's (RET)
annual road race. (RET)
```

- Respond to a prompt — Sometimes pressing the RETURN key causes the system or a program to assume a response, or "default." For example, the following system prompt allows you to enter a line width, but lets you know that, if you press RETURN without entering a number, the system assumes a width of 50 characters per line:

```
Width of line <50>? (RET)
```

- Verify that the system is in operation — When the system is down, or not operational, it does not respond when you press RETURN. Otherwise, the system displays another DCL prompt:

```
$ (RET)
$
```

Note

A common mistake is to press the NO SCROLL key (on VT100s), HOLD SCREEN key (on VT200 series), or CTRL/S (on VT52s) to stop output to the terminal, and then forget to press NO SCROLL, HOLD SCREEN, or CTRL/Q to resume output. (Nothing is printed at the terminal when you press NO SCROLL, HOLD SCREEN, or CTRL/S.) In either case, the system does not display another DCL prompt when you press RETURN. You must press NO SCROLL or HOLD SCREEN again, or CTRL/Q, to resume output.

Table 1-1 lists the RSTS/E special function keys.

Table 1-1: Special Function Keys on RSTS/E

Key	Function
CTRL/C	Halts execution of the current command or program and returns control to the job keyboard monitor. Echoes as "^C" on the terminal.
CTRL/O	Stops and restarts terminal output while a program is running. The stopped terminal output is lost. Echoes as "^O" on the terminal.
CTRL/Q	Resumes terminal output suspended by CTRL/S while a program is running. You can only use CTRL/Q if the TTSYNC characteristic is set for the terminal. (See Chapter 5 for more information on terminal characteristics.)
CTRL/R	Redisplays the current terminal line.
CTRL/S	Suspends terminal output while a program is running. You can use CTRL/S only if the TTSYNC characteristic is set for the terminal. (See Chapter 5 for more information on terminal characteristics.)
CTRL/T	Displays a one-line status report for your job. The report includes your job number, keyboard number, job state, and other information. See Chapter 4 for more information. (Note that CTRL/T is an optional feature, so it may not be available on your system.)
CTRL/U	Deletes the current terminal line. CTRL/U does not erase characters from the screen. Instead, it echoes "^U" and moves the cursor to the next line.
CTRL/X	Deletes unprocessed terminal input, including the current line. Echoes as "^X" on the terminal.
CTRL/Z	Is an end-of-file (EOF) marker. CTRL/Z can also be used to end a DCL command and return to the $ prompt. Echoes as "^Z" on the terminal.
DELETE or RUBOUT	Erases the last character typed. Hard-copy terminals display erased characters between backslashes.
ESCAPE or ALTMODE or CTRL/[	Sends a typed line to the system for processing. Echoes as a "$" on your terminal.
FORM FEED or CTRL/L	Sends a typed line to the system for processing. Performs four line feed operations at a video terminal. Advances paper to the top of the next page on hard-copy terminals. (See Chapter 5 for information on the FORMFEED terminal characteristic.)
HOLD SCREEN	Performs the same function as CTRL/S and CTRL/Q on VT200-series terminals. The TTSYNC characteristic must be set for the terminal.
LINE FEED or CTRL/J	Ends the current line. (Same as RETURN.)
NO SCROLL	Performs the same function as CTRL/S and CTRL/Q on VT100, VT101, VT125, and other VT100-series terminals. The TTSYNC characteristic must be set for the terminal.
RETURN	Sends a typed line to the system for processing. Performs a carriage return/line feed operation at the terminal.
TAB or CTRL/I	Moves the cursor to the next tab stop on the terminal line. (By default, tab stops are eight spaces apart.)

Beginning a Terminal Session: HELLO or LOGIN

You can log in to your account on the system as soon as you have a PPN and password. RSTS/E provides two commands for logging in to your account: HELLO and LOGIN. To get started, type HELLO (or LOGIN) and press RETURN:

```
HELLO (RET)
```

The system signals that it is in operation by displaying a message similar to:

```
RSTS/E V9.0    15-Jun-85      05:52 PM

User:
```

This message gives you the following information:

- The name and version number of the computer's software.
- The date and time when you log in. RSTS/E keeps track of the length of time you use the system and its resources, starting with the date and time of login.
- The prompt for your PPN. RSTS/E displays the User: prompt.

Enter your PPN and password after the prompts as shown. Note that the system does not display your password as you type it. This prevents anyone who may be watching from discovering your password.

```
User: 52,20 (RET)
Password: (RET)
```

You will generally be able to log in this way with no problems. However, it may help you to know the following constraints:

- If you wait too long before answering the User: and Password: prompts (the limit is 30 seconds) or if your response to either prompt is incorrect, the system prints ?Invalid entry — try again, and reprints the User:prompt. If you type in the wrong PPN, the system does not tell you until after you have typed your password.
- After two invalid login attempts, RSTS/E waits 5 seconds before displaying the User:prompt; this wait time increases with each successive invalid login attempt.
- After five invalid login attempts, the system prints ??Access denied. If you are trying to use the system over a dial-up line when this happens, the system disconnects the phone line. Verify your PPN and password before trying again.

When you log in successfully, you may see a message showing the date and time you last logged in to the system. This message confirms that you now have access to RSTS/E. For example:

```
Last logged in on 14-Jun-85, 12:32 PM at KB2:
```

Your system manager may choose to display other messages of general system interest:

```
15-Jun-85    The system will be down Saturday, 17-Jun-85,
             from 8:00 AM to 1:00 PM for maintenance.

$
```

RSTS/E is now at command level, which means that it is ready to accept the commands you enter. The dollar sign ($) prompt indicates the DCL command environment, which is the default for RSTS/E.

You can also use the LOGIN command when you are already logged in to the system if you want to "reset" your current job to the state it was when you originally logged in. Typing LOGIN while you are logged in causes RSTS/E to execute the login procedure again, as if you had logged out and then logged back in.

The System Command Environment

The default keyboard monitor on your system is the command environment that is present when you log in. DCL is the default keyboard monitor for RSTS/E systems. The dollar sign ($) prompt means that RSTS/E is ready for you to enter DCL commands.

Other Command Environments

From DCL, you can switch to one of three other command environments, all of which have different prompts, as shown:

Ready	The BASIC-PLUS environment
>	The RSX environment
.	The RT11 environment

Some commands — such as RUN to run a computer program — are common to all environments. Other commands are available in only one environment, although the actions you perform are similar throughout the command environments. For example, you use the DELETE command in the DCL environment to delete files you have created; you use the UNSAVE command in the BASIC-PLUS environment.

RSTS/E uses DCL as its default command environment because DCL commands are used on other DIGITAL systems, including VAX/VMS. So if you change from a RSTS/E to a VAX/VMS system, or back again, you will be able to use many of the same commands.

You may want to switch from the DCL command environment, depending on your work needs or your particular preference. For example, the BASIC-PLUS environment is useful if you do a great deal of programming in the BASIC-PLUS language. By the same token, the RSX and RT11 environments emulate the commands available on DIGITAL's RSX and RT-11 operating systems for the PDP-11 computer; if you have worked with either of these systems, you may feel more "at home" in one of these environments.

Appendix B provides more detailed information about switching between the command environments. In addition, it lists some of the commonly used commands in the BASIC-PLUS, RSX, RT11 environments.

Using Login Command Files

At login, the RSTS/E system attempts to execute the system-wide command file LOGIN.COM located in account [0,1] on the system disk. This file contains commands defined by the system manager that are executed for all users. In addition, the system-wide login command file normally contains a command that invokes a private LOGIN.COM file located in your directory on the system disk. If your system manager allows the use of private LOGIN.COM files, the system automatically executes these commands every time you log in. This feature lets you tailor the system for your everyday use.

For example, if you enter the same sequence of commands every time you log in, place these commands in a command file (also called a command procedure) named LOGIN.COM. You can also define synonyms for often-used DCL commands in your LOGIN.COM file, making them easier to type and remember. The LOGIN.COM file could also contain commands to assign logical names, run programs, execute other command procedures, or display message files. For example:

```
$ TYPE $NEWS.TXT                ! Display system messages
$ @TERM.COM                     ! Execute a command procedure
$ RUN DB1:[2,214]COOKIE.EXE     ! Run a program
$ ASSIGN DR3:[4,214] WORK:      ! Assign a logical name
$ TODO == "@REMIND"             ! Define a command to execute a procedure
$ S == "SHOW SYSTEM"            ! Abbreviate a command
$ WH-O == "SHOW USERS"          ! Nickname a command
```

Use the at sign (@) command to execute and test your LOGIN.COM file. For detailed information on creating and executing various types of command procedures, including LOGIN.COM files, see the *RSTS/E Guide to Writing Command Procedures.*

Figure 1-1 shows the difference between the user-defined and the system-defined login command files.

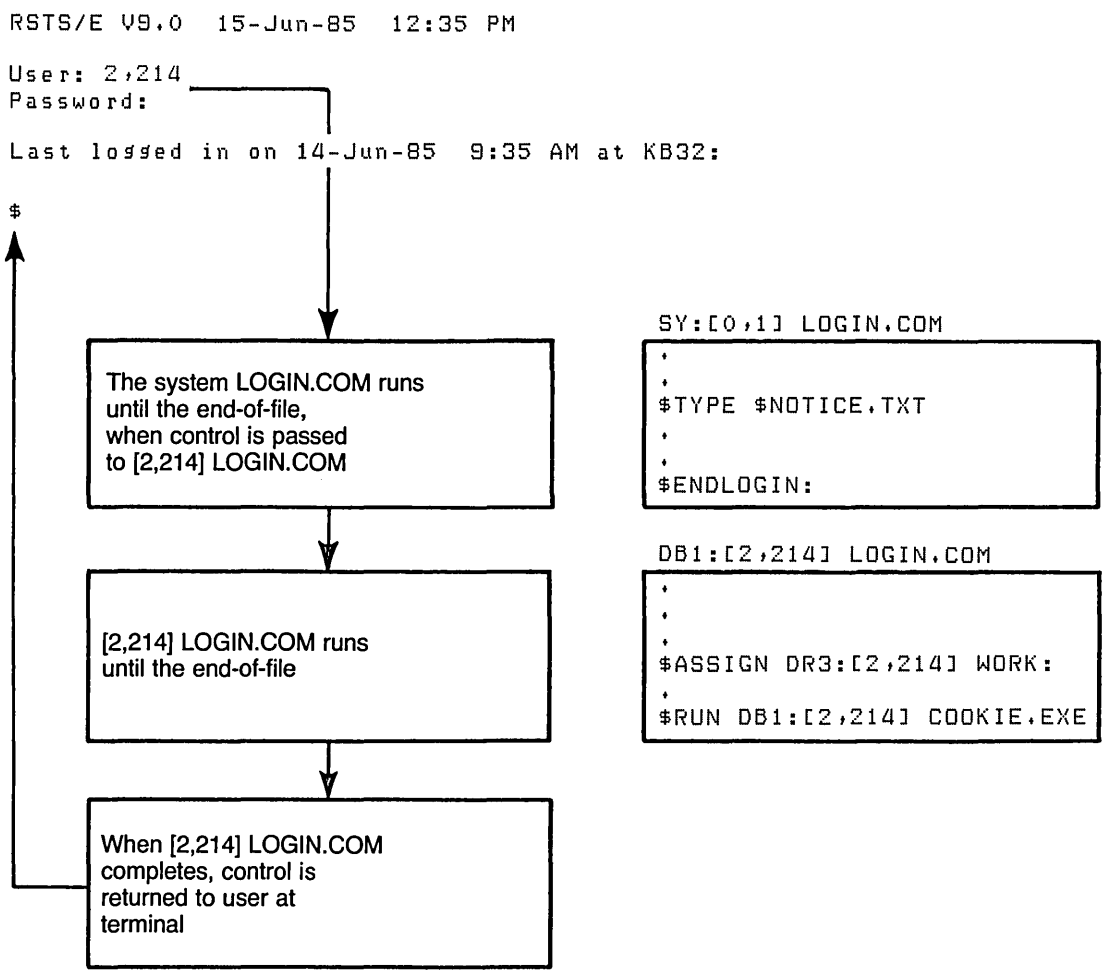

Figure 1-1: Executing System and User Login Files

In Figure 1-1, the file specification for the system-defined login command file is SY:[0,1]LOGIN.COM. The file specification for the user-defined login file is DB1:[2,214]LOGIN.COM. When user [2,214] logs in, the system-defined login file executes first. After the system-defined login file completes, the command interpreter locates and executes the user-defined login file on the default disk and directory for user [2,214].

Getting Information About Commands: HELP

You can use the help facility to learn about the commands that are available on RSTS/E. Each time you use HELP, RSTS/E displays the information you request and then displays the DCL prompt.

To use HELP, type:

```
$ HELP RET
```

The system displays the list of DCL topics on which you can get information:

```
DCL

The RSTS/E System Users Guide contains descriptions of the DCL
commands that you use in file, system, and programming operations.

The RSTS/E System Managers Guide contains descriptions of the DCL
commands used in system management operations.

See the RSTS/E Quick Reference Guide for the syntax requirements
for all DCL commands on RSTS/E.

For instructions on using this HELP facility, type HELP HELP.


Additional help is available on:

@               ALLOCATE        APPEND          ASSIGN          ATTACH
BACKUP          BASIC           BROADCAST       BYE             CCL
CLOSE           COBOL           COPY            CREATE          DEALLOCATE
DEASSIGN        DEFINE          DELETE          DETACH          DIBOL
DIFFERENCES     DIRECTORY       DISMOUNT        DUMP/SYSTM      EDIT
EOD             EXIT            Expressions     FORCE           FORTRAN
Functions       GOTO            HANGUP          HELP            IF
INITIALIZE      INQUIRE         INSTALL         Labels          LINK
LOAD            LOGIN           LOGOUT          MACRO           MAIL
MERGE           MOUNT           ON              OPEN            Operators
PRINT           READ            REMOVE          RENAME          REQUEST
RESTORE         RSTS            RUN             SET             SHOW
SORT            START           STOP            SUBMIT          Symbols
TYPE            UNLOAD          WRITE
```

The help display on your system may be different from what is shown here because the system manager can change the help information.

The display lists topics on which there is help available. For example, if you want to read about the LOGIN command, you type:

```
$ HELP LOGIN (RET)
```

RSTS/E displays information about the LOGIN command and lists its restrictions and qualifiers. The display on your terminal reads:

```
$ HELP LOGIN

LOGIN

  The LOGIN command is used to:

      o Create a new job by logging into an account at a terminal
      o Log in your current job under a different account
      o Reset your current job to its initial logged-in state

  Format

      LOGIN [account]

  Privilege Required

      DEVICE to log in an account at a restricted terminal
      GACNT or WACNT to log in a new job under a different account

  Command Qualifiers                    Defaults

      /[NO]OVERRIDE[=NOLOGINS]          /NOOVERRIDE
      /TERMINAL=KBn[:]

  Prompts

  Password:  (if logging into different account and no WACNT or
              GACNT privilege)

  For more information, see the RSTS/E System User's Guide and the
  RSTS/E System Manager's Guide.
```

The RSTS/E topic provides you with the RSTS/E system help facility, similar to the following display:

```
$ HELP RSTS (RET)

RSTS

  Help can be obtained on a particular topic by typing:
          HELP  topic  subtopic  subsubtopic  ...
  A topic can have the following format:
     1) an alphanumeric string (e.g., a command name, option, etc.)
     2) same preceded by a "/" (=> interpreted as a switch)
     3) the match-all symbol "*"
  Examples:
      HELP DIRECTORY /S
      HELP SET STALL
  Abbreviations result in all matches being displayed.

  Additional help is available on:

  /OUTPUT         /PROMPT
  ADVANCED        ASSIGN          ATTACH          BASIC
  BYE             DCL             DEASSIGN        DISMOUNT
  DIRECTORY       EXIT            FILENAMES       FIT
  HELP            HELLO           KEYBOARD        LOGIN
  MOUNT           PIP             PLEASE          QUE
  REASSIGN        RT11            RSX             RUN
  SWITCH          SYSTAT          TECO            TYPE
  VTEDIT
```

RSTS/E help messages are available in a "hierarchy." This means that the help you request is available in levels of increasing detail. You can type HELP RSTS and a topic, such as RSX, for information about the topic:

```
$ HELP RSTS RSX (RET)
```

Each topic may have a subtopic, which you can include in the command line:

```
$ HELP RSTS RSX DISMOUNT (RET)
```

Ending a Terminal Session: LOGOUT or BYE

RSTS/E provides two commands for ending, or logging out of, your terminal session when you are finished using the system: LOGOUT and BYE. (Use whichever of these two commands you find most convenient; their effect is identical.)

You can include one of two qualifiers when using the LOGOUT command: /BRIEF or /FULL.

If you simply type LOGOUT (or BYE), the /FULL qualifier is the default:

```
$ LOGOUT (RET)
```

RSTS/E displays information about the status of your account at the time of logout:

```
Saved all disk files on SY: 1568 blocks in use
Job 13 User 52,20 logged off KB2 at 15-Jun-85 02:36 PM
System RSTS V9.0
Run time was 3 minutes, 2.4 seconds
Elapsed time was 21 minutes
Good afternoon
```

The log-out information tells you:

- The status of your permanent files (Saved all disk files). RSTS/E also shows the use of your file storage space (1568 blocks in use).
- Your job number (13), PPN (52,20), terminal number (KB2), the date (15-Jun-85) and time of logout (02:36 PM).
- The name and version of the computer's software (RSTS/E V9.0).
- The time you spent at the terminal (run time and elapsed time).
- Any log-out messages (Good afternoon).

Note the difference between run time (3 minutes and 2.4 seconds) and elapsed time (21 minutes). Run time is the amount of CPU time you used. The commands you type require the use of the system's central processing unit (CPU). Elapsed time shows how long you were logged in to the system. Although you were logged in for 21 minutes, you used the computer's CPU for only a fraction of that time.

If you include the /BRIEF qualifier after the the LOGOUT command, RSTS/E ends your session at the terminal by displaying a one-line message. The BYE command has a corresponding qualifier, /F (for "fast"), which produces the same message as LOGOUT/BRIEF:

```
$ BYE/F (RET)

Job 13 User 52,20 logged off KB2 at 15-Jun-85 02:36 PM
```

Exceeding Your Disk Quota

On occasion, you may create more files than your RSTS/E disk quota allows. A disk quota is the amount of file storage space allocated to your account.

RSTS/E enforces two disk quotas for your account: a "logged-in" quota which is enforced all the time you are using the system; and a "logged-out" quota, which is enforced at log-out time. You cannot log out if you have exceeded your quota; you must first delete one or more files and then try again.

For example, if you created a file named BIRDS.TXT to practice editing text, but you no longer need the file, you can delete it by typing:

```
$ DELETE BIRDS.TXT (RET)
```

If you try to log out and find you have exceeded your disk quota, the system displays a message similar to:

```
$ LOGOUT (RET)
Disk quota of 1000 exceeded by 128 blocks
Some file(s) must be deleted before logging out

$
```

In the previous example, RSTS/E will not let you log out until you free at least 128 blocks of storage space. The Size column in the DIRECTORY command (see Chapter 3) includes the blocks of storage space each file requires:

```
$ DIRECTORY (RET)

Name .Type   Size    Prot     Name.Typ    Size    Prot    SY:[52,20]
VANISH.COB     89   < 60>   PATTYC.RNO     130   < 60>
STRIDR.BAS    120   < 60>   ROAD  .MAP     280   < 60>
FLAGS .DAT    159   < 60>   TEST  .MAC     350   < 60>

Total of 1128 blocks in 6 files in SY:[52,20]
```

In general, it is a good idea to delete any files you do not need before attempting to log out. This practice saves space on the system. If you need all your files, you can copy some of them onto magnetic tape to store them before deleting them from your directory. See Chapter 6 for information about copying files to magnetic tape.

Using DCL Command Qualifiers

In the DCL environment, you use commands by typing a command name and pressing the RETURN key. You can also use qualifiers, which modify the effects of a command, by typing a slash (/) followed by a word the computer recognizes as a qualifier. For example, the LOGOUT command has two possible qualifiers: /BRIEF and /FULL.

You type a qualifier after the command name but before pressing the RETURN key. If you do not use qualifiers, DCL makes assumptions known as defaults.

For example, if you use the LOGOUT command, the /FULL qualifier is the default. You could type the /BRIEF qualifier instead, which modifies the effect of LOGOUT by displaying a one-line message instead of the full log-out display.

Chapter 2 describes the use of qualifiers in more detail. In addition, the command descriptions in this manual list the qualifiers and defaults available for each command.

Maintaining Files

Many of the DCL commands in this manual affect files. It is helpful to understand the relationship between accounts and directories when you work with files.

Accounts, Directories, and Files

When you begin a RSTS/E session, you log in to an account. Your PPN is an account number, which identifies you to the system.

The system uses your account to keep track of the system resources you use, such as the amount of time you access the computer's memory (run time) or are logged in (elapsed time), and the amount of storage space your files require. If you have more than one account on the system, RSTS/E sees each account as belonging to a separate user, who is identified by a PPN.

A directory is a collection of files in your account that are kept in a specific location on a disk. An account may have several directories on different disks. Each directory stores information, such as the name and size of each file.

You identify a file by specifying its location and its name. A file's location consists of:

- The node on the network — A node is a computer system with DECnet/E, which connects two or more systems together. See the section "Using DECnet" for more information.
- The device — Files are usually kept on disks or magnetic tapes. Your files are located on a set of disks called the public structure, unless you specify otherwise.
- The directory — A device can be divided into directories, in which you can store files. You specify your directory like you specify your account, by using your PPN.

A file's name, chosen by the person who created the file, consists of:

- The file name — One- to six-alphanumeric characters (MYFILE)
- The file type — Zero- to three-alphanumeric characters, preceded by a period (.TXT)

Putting Files into Your Directory

You can put files into your directory either by moving them from another directory or account or by creating them. The following example creates a file with the CREATE command (see Chapter 3) to show the placement of a file in a directory.

To use the CREATE command, type:

```
$ CREATE (RET)
```

The command prompts you for a file specification:

```
File:
```

You can assign the file almost any name you choose. For this example, name the file ROAD.MAP:

```
File:  ROAD.MAP (RET)
```

Enter the text as follows, and press CTRL/Z when you are finished. (CTRL/Z is displayed on the terminal as ^Z, but the characters ^Z do not become part of the file.) If you make a mistake on a line before you press the RETURN key, you can press the DELETE key to correct it.

```
Although I have been a jogger for (RET)
more than eight years, I did not (RET)
enter any road races until 1985. (RET)
(CTRL/Z)
$
```

The newly created file named ROAD.MAP is now in your directory.

Displaying Your Files

If you are new to a RSTS/E system, you may or may not have files in your directory. Files exist in your directory only if someone put them there; it is possible that the system manager or another user gave you some files. If you created ROAD.MAP in the previous section, then information about the file appears in a directory listing.

You can check for files in your directory with the command:

```
$ DIRECTORY (RET)
```

If you have no files in your directory, RSTS/E displays a message similar to:

```
%No files matching [52,20]??????.??? - continuing
```

The PPN ([52,20]) identifies your directory, and the question marks (??????.???) show that no files are found.

If there are files in your directory, the system displays information about each file. For example:

```
Name.Type    Size    Prot     Name.Typ     Size     Prot    Sy:[52,20]
FINISH.COB    89    < 60>    TENK  .RNO      130    < 60>
START .BAS    12    < 60>    ROAD  .MAP        3    < 60>
TEMP16.TMP   258    < 60>

Total of 492 blocks in 5 files in SY:[52,20]
```

Your directory provides the following information about each file in your account:

Name.Type	The names and types of your files.
Size	The size of your files in numbers of blocks. (A block is 512 characters.)
Prot	The protection code assigned to your files. This determines whether or not other users can read, write, or execute the files.

SY:[52,20] identifies your directory as being on the system disk (SY:) in your account ([52,20]).

Certain operations (such as compiling and linking programs) cause RSTS/E to create a temporary file with a .TMP file type. Temporary files are scratch files that programs use for work space. The file named TEMP16.TMP in the preceding listing is a temporary file.

Do not display or work with temporary files, because they often contain codes that can make your terminal work abnormally. Temporary files are mentioned only because they use disk space and are displayed in the directory listing; the system automatically deletes them when you log out.

The TYPE command displays the contents of a file. The following example shows the contents of the file ROAD.MAP:

```
$ TYPE ROAD.MAP (RET)
Although I have been a jogger for
more than eight years, I did not
enter any road races until 1985.
```

File Specifications

A file specification is the full name and location of a file. In its complete form, the file specification includes:

- A node name on the DECnet/E network, if applicable
- The device name or logical name (see Chapter 6 for descriptions)
- The directory
- The file name
- The file type

The format of a complete file specification is:

node::device:[directory]filename.typ

The delimiters (special characters) in a file specification are brackets, commas, and colons. Brackets ([ ]) surround a PPN. A comma (,) separates the two numbers in a PPN. Double colons (::) follow node names, while a single colon (:) follows the device name.

A dollar sign ($) in place of a PPN indicates the directory [1,2], which is the system library. (The system library stores most of the system files and programs.) For example, the following command displays a system file named NOTICE.TXT:

```
$ TYPE $NOTICE.TXT RET
```

A file specification can be as simple as:

```
ROAD.MAP
```

In this case, RSTS/E assumes your own directory, the local node, and the system disk by default. See the section "Using DECnet" for an explanation of host node.

An example of a file specification that does not assume any defaults is:

```
GARP::DR0:[52,20]ROAD.MAP
```

The parts of the file specification are:

GARP:: - The node (system) on the network

DR0: - The device at node GARP:: on which the file is located

[52,20] - The directory on the disk at node GARP::

ROAD.MAP - The file name and type

File Names and Types

When creating a file, you can assign it any name with up to six alphanumeric characters. You can also change the name of a file with the RENAME command (see Chapter 3).

A file type consists of one to three alphanumeric characters that follow a file name. There is a period between the file name and file type. File types provide you with a shorthand way to identify a file's contents. For example, the file STATUS.DOC has a file name of STATUS and a file type of .DOC.

Note

If you assign a file name of more than six characters or a file type of more than three characters, RSTS/E accepts your input then truncates the name to the first six characters and the type to the first three characters. For example, if you name a file LONGNAME.LIST, RSTS/E truncates it to LONGNA.LIS.

You can assign any alphanumeric file name and type in a file specification, although it is a good idea to use a "mnemonic" name. This means that you should assign the file a name that reminds you of its contents. For example, TEST.DAT could be a data file (.DAT) that you create to test a program (TEST). If you have a file containing a BASIC-PLUS program to run your payroll, you might name it PAYROL.BAS. The file type .BAS indicates a BASIC-PLUS source file. (A "source" file contains the program you typed in.)

An advantage of using a mnemonic file type is that RSTS/E recognizes several file types by default (for example, .DOC, .DAT, .RNO, .COM, and so on). If you assign a file one of these types, then you do not always have to type it explicitly. For example, if you are using the COBOL command to compile a program, you can type:

```
$ COBOL PROG1 (RET)
```

RSTS/E assumes the resulting source file to be PROG1.CBL, because the file type .CBL is the default for a COBOL source file.

Table 1-2 lists some common file types on RSTS/E.

Table 1-2: RSTS/E File Types

File	Meaning
.B2S	BASIC-PLUS-2 source file
.BAC	BASIC-PLUS compiled file
.BAK	Backup file created by several types of utility programs
.BAS	BASIC-PLUS source file
.CBL	COBOL source file
.COM	DCL comand procedure
.CRF	Cross-reference listing file
.CTL	Batch control file
.DAT	Data file
.DBL	DIBOL source file
.DIF	DIFFERENCES output file

(continued on next page)

Table 1-2: RSTS/E File Types (Cont.)

File	Meaning
.DIR	File produced by the DIRECTORY command
.DOC	RUNOFF output file
.EDT	EDT initialization file
.ESC	Terminal set-up file
.FLB	FMS-11 form library file
.FOR	FORTRAN-IV source file
.FIN	FORTRAN-77 source file
.HLP	System program help text file
.LIB	Resident library file
.LNK	Input file used by the DCL LINK command
.LOG	Batch output log file
.LST	Listing file
.MAC	MACRO source file
.MAI	DECmail/RSTS mailbox file
.MAP	Map file created by the LINK command or Task Builder
.MLB	MACRO library file used by the RSX-based MACRO assembler
.OBJ	Object file, which is a compiled BASIC-PLUS-2, COBOL, DIBOL, FORTRAN, or MACRO program
.ODL	Overlay Description Language input file used by the Task Builder
.OLB	Object module library file used by the Task Builder
.PMD	Post-Mortem Dump file
.RNO	RUNOFF input file
.RTS	Run-time system
.SAV	Executable FORTRAN-IV or MACRO program produced by the LINK command or the RT11-based LINK program
.SIL	Save Image Library (RSTS/E monitor or system program)
.SKL	Skeleton file produced by the COBOL-81 compiler
.SRT	SORT-11 file
.STB	Symbol table file produced by the Task Builder
.SYS	System file, usually for internal use
.TMP	Temporary file created by a system program (deleted when you log out or if the disk is rebuilt)

(continued on next page)

Table 1-2: RSTS/E File Types (Cont.)

File	Meaning
.TSK	Executable BASIC-PLUS-2, COBOL, DIBOL, FORTRAN-77, or MACRO program produced by the LINK command or the Task Builder
.TXT	ASCII text file

Protection Codes

You determine who, if anyone, can have access to files you create by using protection codes. RSTS/E displays the protection codes of files between angle brackets (< >) in your directory listing. For example:

```
Name .Type   Size    Prot      Name.Typ    Size    Prot     SY:[52,20]
FINISH.COB     89   < 60>    TENK  .RNO     130   < 60>
START .BAS     12   < 60>    ROAD  .MAP       3   < 60>
```

In the example, 60 is the default protection code that RSTS/E assigns. (On some systems, the system manager may have chosen a different default protection code.) If one of your files has a protection code of 60, only you (or a user with additional privileges) can read, edit, or delete it. A protection code remains fixed until you change it. See the section "Allowing Access to Files" in Chapter 3 for a list of available protection codes and a description of how to assign them.

Wildcards

Wildcards let you refer to a group of files all at once by specifying a pattern of identification. You can use wildcards to specify all files that have a common element in their names.

For example, if you have the files TIMING.COB, TIMING.BAS, and TIMING.DAT is your directory, you can display information about them by typing a wildcard (in this case an asterisk) in place of the file type:

```
$ DIRECTORY TIMING.* RET
```

RSTS/E provides the following two wildcard characters:

- Asterisk (*) — Use the * as a wildcard in specifying a PPN (for example, [*,20], [52,*], or [*,*]) or in specifying file names or types (for example, *.BAS, or MYFILE.*).
- Question mark (?) — Use the ? as a wildcard to replace a single character in a file name or type. The question mark wildcard will display all file names containing the alphanumeric string, regardless of whether the single character is present. (For example, specifying PROG?.COB will display PROG.COB, as well as PROG1.COB, PROG3.COB, and PROGB.COB, if those files are present in the directory.)

Running a Program

You run most programs by typing RUN followed by the name of the program. A program that can be run is called an "executable" program. Chapter 8 describes the DCL commands for running, compiling, and linking programs on RSTS/E. For more information about each programming language, refer to the appropriate language manual and user's guide.

There are two types of executable programs: user programs and system programs.

Some programs are referred to as user programs because they are created by a system user. For example, if you create a BASIC-PLUS program named BANKER that balances a checkbook, you can run the user program BANKER.BAC by typing:

```
$ RUN BANKER.BAC (RET)
```

Other programs are referred to as system programs, because they are installed as part of the system software. You can run system programs with one of the following:

- A DCL command — For example, you run the system's editor program (EDT) by typing EDIT and pressing the RETURN key. EDIT is the DCL command that runs the EDT program.
- The RUN command — For example, you can switch to DCL from another keyboard monitor by typing:

```
RUN $SWITCH (RET)
Keyboard Monitor to switch to?  DCL (RET)
$
```

Using DECnet/E

The DECnet facility links two or more DIGITAL computer systems as a network. DECnet refers to both the hardware (machinery) and software (programs) involved in network operations. The DECnet software available on RSTS/E is called DECnet/E.

If your system has DECnet/E, then your system is part of a network that allows you to perform file, system, and programming operations on another system. To find out if your system has DECnet/E, type the following command:

```
$ SHOW NETWORK (RET)
```

You should get one of the following responses:

- If the system displays network information (see the next section), you can use DECnet/E on your system.
- If DECnet/E is on your system but is not currently in operation, the system displays:

```
$ SHOW NETWORK (RET)

No information
```

- If your system does not have DECnet/E, the system displays the error message:

```
?Command not installed
```

Note

If your system does not have DECnet/E, go on to Chapter 2.

You can use some of the commands in this manual over the network. The commands and their qualifiers that you can use in network operations are labeled with the symbol (N) in the command descriptions.

DECnet/E also lets you log in to another computer system from your account. You can then use that system as though your terminal were connected to it directly. Use the SET HOST command (see the next section) to log in to another system, or node, on the network.

To log into another system on the network, you need to have an account and password for that system. In addition, the network between your system and the other system must be working. (You use the SHOW NETWORK command, described in the next section, to check for available nodes on the network.)

You should know the following terms to become familiar with DECnet/E:

Network — The family of software modules, files, hardware components, and facilities that ties different DIGITAL systems together.

Node — A computer system that is connected to other computers by a network. Node names are assigned by the system manager. There are three types of nodes:

- Host — The node you access with the SET HOST command.
- Local — The node you logged in to before using the network; your original system.
- Remote — Any node in the network other than the local node.

To summarize, the node of the network you first log in to is local, and the system you reach by the network is remote. You connect to another system by specifying the host and then logging in. However, if you want to communicate with a system on the network from your local node, you can use DCL commands over the network without logging in to the remote system.

For a complete description of the capabilities that DECnet/E has to offer, see the *Introduction to DECnet* and the manuals in the DECnet/E documentation set.

Displaying Network Status: SHOW NETWORK

The SHOW NETWORK command displays the systems you can connect to on the network. To use the SHOW NETWORK command, type:

```
$ SHOW NETWORK (RET)
```

If the network is in operation, RSTS/E displays the names of the different nodes your system can access. For example:

```
Active Node Volatile Summary as of 11-JUN-85 14:31:11

Executor Node = 1.135 (GARP)

State                     = On
Identification            = GARP DECnet/E V2.0
Active Links              = 3

Remote Node = 1.13 (SNOOPY)

State                     = Reachable
Circuit                   = DMC-1

      .
      .
      .
      .

Remote Node = 1.20 (CLOUD9)

State                     = Reachable
Circuit                   = DMC-1
```

For a complete description of the elements in the SHOW NETWORK display, see the *DECnet/E System Manager's Guide.*

Using the Network: SET HOST

The SET HOST command lets you log in to another computer from the system you are currently logged in to. To use the SET HOST command, type:

```
$ SET HOST (RET)
```

RSTS/E prompts you for the node name with:

```
Node:
```

Enter the name of a node that is available on the network. For example, you connect to a node named SPEEDY by typing:

```
Node: SPEEDY (RET)
```

A faster method is to type SET HOST and follow it directly by a node name, then press RETURN. For example:

```
$ SET HOST SPEEDY (RET)
```

After the connection is made, the local node displays a message stating that the connection has been established. You must have an account on the remote node to log in. When the connection is established, you use the normal log-in dialogue for that system. When you successfully log in, you can use the remote node essentially as if your terminal were connected to it directly.

When you are finished, use the LOGOUT command to log out of the remote node. After about five seconds, control returns to your local node.

There are two ways you can end your session at the remote node and return directly to the local node:

- The first way, which DIGITAL recommends for normal use, is to follow the host system's normal log-out procedure. This deletes any temporary files you may be using.
- The second way is to type CTRL/P, which prompts you for a network command. For example:

```
(CTRL/P) (RET)
GARP::NET> EXIT (RET)
```

For more information on using the network, see the *DECnet/E Guide to User Utilities.*

Using RSTS/E Commands and Command Procedures 2

This chapter shows how to use RSTS/E commands interactively. It also introduces special groups of commands, called command procedures or command files, that the system executes automatically for you.

This chapter explains how to enter:

- Commands
- File specifications
- Qualifiers
- Character string data
- Numeric values
- Date and time values
- Command procedures

Understanding Command Formats

The commands in this manual are grouped by related functions. (For example, Chapter 3 describes all commands that manipulate files.) Each command description includes:

- A brief description of the command's function
- Syntax information (command parameters)
- The command's qualifiers and defaults, if any
- The command's prompts, if any
- Detailed information about each command parameter and qualifier

The command format descriptions in this manual use the terms in Table 2-1.

Table 2-1: Terms Used in Command Formats

Term	Meaning
Format	The use of the elements in a command string.
Qualifiers	The elements that modify the effect of the command string in some way.
Command Qualifiers	Qualifiers that modify the command itself.
File Qualifiers	Qualifiers that modify the treatment of a file included in the command string.
Defaults	The assumptions RSTS/E makes when you enter the command string and do not use all the available parts of the format.
Prompts	The text that RSTS/E displays to request your input.
Command Parameters	The parts of a command string that specify what the command is operating on. (For example, what file is affected by the command.)
file-spec	A file specification.
input file-spec	The specification of a file to be used as input in the command string.
output file-spec	The specification of a file to be used as output in the command string.

For example, the command format in Figure 2-1 lists the syntax elements for using the COPY command. (See Chapter 3 for the complete COPY command description.)

Duplicating, Renaming, and Appending Files

The commands described in this section let you duplicate, rename, and append the contents of files.

COPY

The COPY command duplicates one or more existing files, or concatenates two or more files. You can use this command for local and network file specifications.

Format

COPY input-file-spec[,...] output-file-spec

Command Qualifiers	**Defaults**
/Allocation=n	
/BEFORE=date	
/BLOCK_SIZE=n	/BLOCK_SIZE=512
CLUSTER_SIZE=n	
/CREATE	
/[NO]CONTIGUOUS (N)	
/[NO]LOG (N)	/LOG
/MODIFIED	
/[NO]OVERLAY	/NOOVERLAY
/POSITION[=n]	/POSITION=0
[=MIDDLE]	
[=INDEX]	
/PROTECTION=n	
/[NO]QUERY (N)	/NOQUERY
/[NO]REPLACE (N)	
/SINCE=date	

Prompts

From: input-file-spec[,...]
TO: output-file-spec
OK to replace existing file file-spec ?

Use COPY to:

- Copy one file to another file
- Merge more than one file into a single file
- Copy a group of files to another group of files

For example, if you have a file named SMITH.MAR and you want to make a copy of it as file JONES.JOE, you type:

```
$ COPY SMITH.MAR JONES.JOE (RET)
```

Figure 2-1: The COPY Command Format

Suppose you want to use COPY to copy all the files named JOKES and GAMES in your account to one file named PARTY.TYM on disk DB2:.

In the following example, you specify that the files should not replace existing files. Therefore, you include the following information in the command line:

- A command qualifier (/NOREPLACE)
- Two input file-specs (JOKES.* and GAMES.*)
- One output file-spec (DB2:PARTY.TYM)

The dialogue at your terminal looks like:

```
$ COPY/NOREPLACE (RET)
From:    JOKES.*,GAMES.* (RET)
To:      DB2:PARTY.TYM (RET)
File JOKES.1 copied to DB2:[52,20]PARTY.TYM
FIle JOKES.2 copied to DB2:[52,20]PARTY.TYM
File GAMES.TXT copied to DB2:[52,20]PARTY.TYM
$
```

Note that RSTS/E prompts you for your input. This is called "interactive" mode, meaning that you and the system are communicating directly with each other. RSTS/E can also operate in "batch" mode, meaning that the system is executing commands in a command procedure. (A command procedure is a series of commands that RSTS/E processes automatically without requiring input from you.)

RSTS/E displays its DCL dollar sign ($) prompt when the COPY operation is complete.

Entering Commands

The complete specification of a command, including the command name, command qualifiers, parameters, and file qualifiers (if any), is called a command string. Because you can continue a command on more than one line, RSTS/E uses the term command string to define the entire command that is passed to the system. By contrast, the term command line describes the part of a command string that you type on one line.

The general format of a command string is:

$ command-name[/qualifiers....]parameter[/qualifiers...] [...]

Each item in a command must follow these rules:

- At least one blank or tab character must separate the command name from the first parameter; at least one blank must separate each additional parameter from the previous parameter. Note that you can use multiple blanks and tabs in all cases where a single blank is required.
- Each qualifier must be preceded by a slash (/). You can enter any number of blanks or tabs before or after the slash.

Continuing Commands on More than One Line

The maximum number of characters you can enter on one line is 132. However, you can enter a command string on more than one line by using the continuation character, a hyphen (-), as the last element on a command line.

Command line continuation is especially useful when you enter a command and want to specify many qualifiers, or when you place a command in a command procedure and want to make the procedure more readable. For example:

```
$ PRINT MYFILE - RET
  /AFTER=17:00 - RET
  /COPIES=20 - RET
  /NAME=GUIDO RET
```

There is no restriction on the number of continued lines you can use to enter a command. However, the total number of characters in a command string must not exceed 255.

Entering Comments

Comments explain or document commands or files. You indicate a comment by preceding it with an exclamation mark (!). For example, the following line in a command procedure uses a comment to explain the use of the DIRECTORY command in a control file:

```
$ DIRECTORY W:[*,*] !W: is a logical name for disk DB2:
```

All the text (and spaces) starting at the exclamation point to the end of the line is considered a comment. The comment does not affect the action of the DIRECTORY command.

Comments are valid in the following positions:

- As the first item on a command line; in this case, the entire line is considered a comment and is not processed.
- Following the last character in a command string, or after a hyphen that signals continuation in a command line.

For example:

```
$ !THIS ENTIRE LINE IS A COMMENT RET
$ PRINT MYFILE - !  PRINT COMMAND COMMENT RET
Continue:  /COPIES=3    !  3 COPIES, PLEASE RET
```

The system prints the "Continue:" prompt in the last line to show that what follows is a continuation line.

When you continue a command on more than one line, RSTS/E uses the Continue: prompt to indicate that it is still accepting the command. Note that the hyphen to

continue the command must come before the comment. Hyphens after the exclamation point (!) are ignored. This means that you cannot continue comments, although the next line could read:

```
$ - ! Comment included. RET
```

In this last line, you have no command, just a comment.

Abbreviating Keywords

Keywords are the command names, qualifiers, or options that RSTS/E recognizes. You can abbreviate all RSTS/E keywords by truncating (abbreviating) them to the first four characters. You can also abbreviate many of these keywords to two or three characters, if the keyword remains unique. You can abbreviate:

- Command names
- Command keyword parameters
- Qualifiers
- Qualifier keyword values

The following two sections describe the rules for abbreviating keywords.

Abbreviating Command Names

You can abbreviate command names to two characters if the abbreviation is unique. For example, the PRINT command is the only command that begins with the characters "PR." Therefore, you can abbreviate the PRINT command to two characters. The DEALLOCATE and DEASSIGN commands, however, have the same first three characters, so you cannot abbreviate these commands to fewer than four characters.

Abbreviating Parameters, Qualifiers, and Qualifier Arguments

You can also abbreviate parameters, qualifiers, and qualifier arguments to two characters if the abbreviation is unique. For example, /SIZ is the minimum abbreviation of the /SIZE qualifier; fewer than three characters conflicts with /SINCE. (Note that the slash (/) character is not considered when counting characters.)

Some qualifiers permit a negative form. For example, /NOJOURNAL is the negative form of the /JOURNAL qualifier. In applying the minimum four-character abbreviation rule, do not count the NO prefix as the first two of the four characters. In this case, the minimum abbreviation that guarantees uniqueness is /NOJOUR.

The underscore (_), as in the / [NO]D_LINES qualifier, is considered optional syntax. This means that the minimum abbreviation for guaranteed uniqueness is /D_LIN and /NOD_LIN.

Entering File Specification Lists

An input file parameter for many commands has the format:

file-spec[,...]

This format indicates that you can enter more than one file specification. You can separate the file specifications with commas (,) or plus signs (+). See Table 2-2 for further discussion.

Any number of blanks or tab characters can come before or follow the commas or plus signs. RSTS/E treats the list of file specifications as a single parameter.

Entering Entry Specifications

An entry specification parameter, often used in Print/Batch Services (PBS) commands, has the format:

queue-name:[ppn]entry-spec

This format indicates:

- The queue in which the print or batch entry is to be placed.
- The project-programmer number (PPN) assigned to the entry.

  If you do not specify a PPN, the default is your own. (Unless you have sufficient privileges, you cannot specify a PPN different from your own.)
- The name of the print or batch entry.

See Chapter 7 for more information on entry specification.

Entering Qualifiers

Commands can take one or both of the following types of qualifier:

- Command qualifiers
- File qualifiers

Command qualifiers have the same meaning regardless of whether they appear after the command name or after a parameter. For example:

```
$ PRINT/QUEUE=LP1: SPRING,SUM,FALL,SUM (RET)
$ PRINT SPRING,SUM,FALL,SUM/QUEUE=LP1: (RET)
```

In this example, the /QUEUE qualifier is a command qualifier; therefore, the two PRINT commands are the same.

Unlike command qualifiers, file qualifiers can have different meanings depending on where you place them in the command line. If you specify a file qualifier immediately after a file specification parameter, it affects only the file it follows. If you specify a file qualifier after the command name, it affects all the files specified as parameters.

For example:

```
$ PRINT/COPIES=2 SPRING.SUM,FALL.SUM (RET)
$ PRINT SPRING.SUM/COPIES=2,FALL.SUM (RET)
```

The first PRINT command requests two copies of each of the files SPRING.SUM and FALL.SUM. The second PRINT command requests two copies of the file SPRING.SUM, but only one copy of FALL.SUM.

In some cases, you can specify file qualifiers only after a parameter — not after the command name. (For example, the /LIBRARY qualifier used with the MACRO command.) In this case, the qualifier description indicates that the qualifier can apply only to a file parameter. In general, however, if you specify a file qualifier following the command name, the qualifier applies to all files specified.

Determining Qualifier Defaults

Many of the command formats in this manual show defaults. When you do not explicitly include a qualifier in a command line, RSTS/E assumes a default qualifier. For example, when you copy a file, RSTS/E normally displays a message confirming that the COPY operation was successful. If you type /NOLOG in the command line, you suppress the message. Otherwise, RSTS/E assumes the /LOG qualifier and displays the message.

Entering Qualifier Arguments

Many qualifiers can be followed by a keyword, file specification, character string, or numeric value.

You must separate a qualifier and value with either an equal sign (=) or colon (:). For example, the following specifications are the same:

```
/OUTPUT=DB1:NEW.DAT          /OUTPUT:DB1:NEW.DAT
```

Many qualifiers accept one keyword or variable value. This manual presents this format as:

/qualifier=value

For example:

```
/COPIES=3
/DATE=MODIFIED
```

Entering Output File Qualifiers

Some qualifiers request output from a command and optionally accept a file specification value. For example, the /LIST and /OBJECT qualifiers for the compilers, as well as the /EXECUTABLE and /MAP qualifiers for the LINK command, are output file qualifiers that fit into this category.

The following rules apply to output file qualifiers:

- If the qualifier is present by default, the output file specification defaults to your directory on the public disk structure and the name of the first input file. The qualifier provides a default file type. For example:

Command	Output File
LINK A	A.TSK
LINK A,B	A.TSK
LINK [52,20]A,[52,20]B	A.TSK
LINK A.OBJ	A.TSK

- If the qualifier does not specify an output file specification, the output file specification defaults to your account on the public disk structure and the name of the first input file. The qualifier provides a default file type. For example:

Command	Output File
LINK/EXECUTABLE A	A.TSK
LINK/EXECUTABLE A,B	A.TSK
LINK/EXECUTABLE A.OBJ	A.TSK

- If you specify /LIST without a file specification, the listing goes where the object file goes, but it has an .LST file type. If you do not specify an object file, then the listing has the same name as the first input file, an .LST file type, and is in your directory on SY:.
- If you specify /MAP without a file specification, the map goes where the executable file goes, but it has a .MAP file type. If you do not specify an executable file, then the map has the same name as the first input file, a .MAP file type, and is in your directory on SY:.
- If the qualifier indicates a file specification for the output file, then any parts entered in the file specification are used to name the output file, and no default file name is supplied. For example:

Command	Output File
LINK A,B/EXECUTABLE=C	C.TSK
FORTRAN/LIST=A B+C	A.LST, B.OBJ
LINK/EXE=[52,20]A	[52,20]A.TSK

Entering Uppercase, Lowercase, and Nonalphanumeric Characters

You should remember the following rules when you enter RSTS/E commands:

- You can use any combination of uppercase and lowercase letters; both are treated the same.

- RSTS/E treats multiple blanks and tabs the same as a single blank.

  RSTS/E ignores blanks and tabs at the beginning or end of the command string, around commas and plus signs, around equal signs, and around colons that delimit qualifier values. RSTS/E does not permit blanks or tabs in file specifications, dates and times, keywords, or numbers.

- You can use quoted, or literal, strings in some RSTS/E commands (such as REQUEST) and in network file specifications. A quoted string consists of characters enclosed in quotation marks.

Table 2-2 lists nonalphanumeric characters that have special meanings in RSTS/E.

Table 2-2: Nonalphanumeric Characters

Symbol	Name	Meaning
:	Colon	A colon can serve one of the following functions: • Device name delimiter in a file specification. • Node name delimiter; used as a double colon (::). • Qualifier value delimiter; separates a qualifier name from its value.
/	Slash	Qualifier delimiter.
+	Plus sign	List element separator for parameters.
,	Comma	List element separator for parameters.
-	Hyphen	Continuation character. As the last nonblank character in a command string, indicates that a continuation line follows. If the statement to be continued contains a comment, the hyphen must be the last nonblank character before the exclamation point.
[ ]	Square brackets	Directory delimiters in a file specification.
?	Question mark	Wildcard character in a file specification.
=	Equal sign	Qualifier value delimiter; separates a qualifier name from its value.
*	Asterisk	Wildcard character in a file specification.
.	Period	File type delimiter in file specifications.
!	Exclamation point	Comment delimiter.
"	Quotation mark	Literal string delimiter.
'	Apostrophe	Symbol substitution delimiter.
$	Dollar sign	Command file character. Used as the first character in first position of a command file statement. In a file specification, denotes the directory [1.2], which is the system library.
@	At sign	Executes command procedure interactively.

(continued on next page)

Table 2-2: Nonalphanumeric Characters (Cont.)

Symbol	Name	Meaning
<SPACE>	Space	Separates fields in a command string. Otherwise ignored unless embedded in a string delimited by quotation marks.
<TAB>	Tab	Separates fields in a command string. (Equivalent to one space [blank] character; otherwise ignored.)

Entering Dates and Times

Some RSTS/E commands let you specify a date or time when the command will take effect. For example, you can print 10 copies of a file to be printed after 5:00 PM on a weekend, when fewer people need the line printer, by typing a line such as:

```
$ PRINT NOTIN.NOW/COPIES=10/AFTER=15-JUN-85:5:00PM (RET)
```

In RSTS/E, you can specify the following date and time formats:

- Absolute — A specific date and time of day. For example:

  ```
  23-May-1985:10:53:22
  ```

- Relative — A date or time calculated from the current date or time. For example:

  ```
  +3DAYS or +2HOURS
  ```

- Combinations of absolute and relative dates and times. For example:

  ```
  TOMORROW+3DAYS-1HOUR
  ```

The /BEFORE, /SINCE, and /AFTER Qualifiers

You can specify a date with any command that allows the /SINCE or /BEFORE qualifier. (Generally these commands are used for working with files.) The /SINCE=date qualifier applies to files that are created or modified on or since the specified date, depending on whether or not you also use the /CREATED or /MODIFIED qualifier.

The /BEFORE=date qualifier applies to files that are created or modified before the specified date, depending on whether or not you also use the /CREATED or /MODIFIED qualifier.

You can use the /SINCE and /BEFORE qualifiers together to provide a range of dates for file operations.

The /AFTER qualifier lets you specify a date, a time, or both. Use /AFTER with commands that affect jobs in a batch or print queue, to indicate that the system should not begin executing the batch job, or print the file, until after a specified time.

Absolute Date and Time Formats

There are three formats for absolute dates:

- The alphanumeric format is dd-mmm-yy, as in 23-MAR-85. In this format, dd is the date of the month; mmm is the first three letters of the month; and yy is the last two digits of the year, or the entire year (as in 1985). The yy is optional — you can just type dd-mm. If you omit the year, RSTS/E assumes the current year.
- The numeric format is yy.mm.dd, as in 85.3.23 (for March 23, 1985). In this format, yy is the last two digits of the year or the entire year (as in 1985); mm is the month; and dd is the date of the month.
- Use these keywords to specify a date: TODAY, TOMORROW, or YESTERDAY.

Note

You can specify any date from 1-JAN-70 to 31-DEC-99, inclusive. If you specify a date outside that range, you get the message:

```
?Invalid date
```

There are two formats for absolute times: 24-hour time and AM/PM time.

- A 24-hour time has the form hh:mm. In this format, hh is the hour, in the range 0 to 24, and mm is the minute, in the range 0 to 59. The minute is optional. The default minute is 00. For example, 11 is equivalent to 11:00. The limits of time formats are:
  - 00:00 is midnight, and is considered the beginning of the specified date.
  - 12:00 is noon.
  - 24:00 is midnight, and is considered the end of the specified date.
- An AM/PM time can be in the format hh:mmAM, hh:mmPM, or 12:00M. In this format, hh is the hour, in the range 1 to 12; mm is the minute, in the range 0 to 59; and M (for meridian) stands for noon. You can use either 00:00 or 24:00 to specify midnight.

Combinations of Absolute Dates and Times

The order for specifying both a date and time is date:time. For example:

```
23-JUN-85:11:00AM

TOMORROW:1:00AM
```

If you specify just a time, RSTS/E assumes TODAY. If you specify just a date when using the /AFTER qualifier, then RSTS/E assumes 11:59PM. (In other words, the job processes after the end of the specified day.)

Syntax

The punctuation marks (syntax) in a time specification indicate the time value you enter. If you specify both the date (dd-mmm-yy) and the time (hh:mm), you must type the colon between the date and the time.

A line ending with a hyphen (-) is continued, rather than recognized as a part of the date. For example:

```
$ PRINT/AFTER=15-
  CONTINUE: RET
```

This PRINT command specifies 3:00 PM today, according to 24-hour time.

Relative Date and Time Formats

In addition to absolute dates and times, you can also specify relative dates and times. There are three formats for relative dates and times:

- + or −nD[AYS]. For example:

  +23D or -2DAYS

- + or −nH[OURS]. For example:

  +23H or -8HOURS

- + or −nM[INUTES]. For example:

  +30M or -15MINUTES

You may specify these formats in any order or in any combination you choose.

If you specify a relative date, you may or may not want to also specify either an absolute or relative time, with the following results:

- If you do specify an absolute time, then that time is used. For example, PRINT/AFTER=3PM+2DAYS means 2 days from today at 3:00 PM.
- If you do not specify an absolute time, but specify only a relative time, then the time is calculated from the current time. For example (if it is currently 1:45 PM), PRINT/AFTER=TOMORROW+2HOURS means tomorrow at 3:45 PM.
- If you do not specify a time at all, then time is not considered. For example, PRINT/AFTER=05-AUG+3DAYS means first thing on August 9th; this has the same result as specifying /AFTER=08-AUG.

Some other examples of specifying relative dates and times are:

- To print out a listing three days from today starting at 3:00 PM, type:

```
PRINT/AFTER=3:00PM+3DAYS
```

- To print a file tomorrow at the same time as today, type:

```
PRINT/AFTER=TODAY+0HOURS
```

or

```
PRINT/AFTER=+24HOURS
```

- To set up a batch job named WEEKLY that resubmits itself every seven days starting at 2 PM, type the following command in WEEKLY.COM:

```
SUBMIT/AFTER=2:00PM+7DAYS WEEKLY.COM
```

Using Command Procedures

Command procedures are files that contain DCL commands. You execute these commands by running the procedure. Use command procedures to execute sequences of commands you use during interactive terminal sessions or commands you submit for batch processing. To fully understand how to use command procedures on RSTS/E, see the *RSTS/E Guide to Writing Command Procedures.*

Command procedures can range from simple to complex. A simple command procedure consists of one or more command lines for the DCL command interpreter to execute. For example, the following command procedure deletes all temporary (.TMP) files and then shows a directory listing:

DIRECT.COM

```
$ ! Delete .TMP files and show directory
$ DELETE *.TMP
$ DIRECTORY
```

A more complex command procedure performs program-like functions. It can:

- Contain loops and error checking routines
- Perform arithmetic calculations and input/output (I/O) operations
- Manipulate character string data
- Call or pass parameters to other command procedures

For example, the following command procedure displays all occurrences of files with a .B2S file type in a user's account:

B2S.COM

```
$ ! Display all .B2S files
$ NEXT_FILE = F$SEARCH("_SY:_*.B2S")
$ LOOP:
$ IF NEXT_FILE .EQS. "" THEN EXIT
$ WRITE 0 NEXT_FILE
$ NEXT_FILE = F$SEARCH()
$ GOTO LOOP
```

Creating Command Procedures

To create a command procedure, use either a text editor (such as EDT) or the DCL CREATE command. The following example shows how to create a simple command procedure using EDT:

```
$ EDIT/EDT RUN.COM (RET)
Input file does not exist
[EOB]
$ !Run three programs
$ RUN PROG1 (RET)
$ RUN PROG2 (RET)
$ RUN PROG3 (RET)
(CTRL/Z)
*EXIT
RUN.COM 4 lines
$
```

The next example shows how to create the same procedure using the DCL CREATE command:

```
$ CREATE RUN.COM (RET)
$ !Run three programs
$ RUN PROG1 (RET)
$ RUN PROG2 (RET)
$ RUN PROG3 (RET)
(CTRL/Z)
$
```

Note that the examples use the file type .COM. When you enter the at (@) command at the DCL prompt, the command interpreter assumes that the at sign (@) character is followed by the name of a file with the file type .COM. In addition, if you enter the @ command without specifying a file, RSTS/E prompts you for the file specification.

Formatting Command Procedures

When you create a command procedure, start each command line with a dollar sign ($) character. The use of the $ character ensures that DCL will process the command when you execute the procedure from another keyboard monitor such as BASIC-PLUS. Omit the $ character for data lines.

Executing Command Procedures

You can execute command procedures in many different ways on RSTS/E systems. For example:

- At login time, using LOGIN.COM files
- At DCL command level, using the at sign (@) command (or from any other keyboard monitor, using the $ character before the @ command)

- In batch processing, using the SUBMIT command
- From inside another command procedure (nesting)
- With the RUN command from any keyboard monitor, providing the command file is executable

See the *RSTS/E Guide to Writing Command Procedures* for a complete description of each method of executing command procedures.

3 Working With Files

This chapter describes DCL commands that are used in file operations.

Table 3-1 lists the commands as they appear in this chapter.

Table 3-1: Commands for File Operations

Command	Meaning
Creating and Modifying Text Files	
CREATE	Lets you enter text that you can save as a file
EDIT	Starts the EDT text editor
Displaying File Names and Files	
DIRECTORY	Displays information about files in a directory
TYPE	Displays the contents of individual text files at the terminal
Deleting Files	
DELETE	Deletes files from a directory
Copying, Renaming, and Appending Files	
COPY	Duplicates a file or concatenates files
RENAME	Assigns a new name to a file
APPEND	Copies a file to the end of another file
Sorting and Merging Files	
SORT	Invokes the PDP-11 SORT utility to reorder the records in one through ten files into a defined sequence and to create a single new file containing the reordered records
MERGE	Invokes the PDP-11 MERGE utility to combine two through ten similarly sorted input files and create a single output file

(continued on next page)

Table 3-1 Commands for File Operations (Cont.)

Command	Meaning
Comparing Files	
DIFFERENCES	Creates a new file that lists differences in the text of two other files. Use DIFFERENCES to compare earlier versions of a file with its later versions.
Allowing Access to Your Files	
SET PROTECTION	Specifies the protection code of a file to determine who, if anyone, can read, modify, or delete a file. The /DEFAULT qualifier assigns the same protection code to all the files you create in a session at the terminal.
Setting File Characteristics	
SET FILE	Sets various characteristics of a specified field

Creating and Modifying Text Files

The CREATE and EDIT commands let you create files in DCL.

CREATE

The CREATE command lets you enter text and save it as a file.

Although you will use the EDIT command more often than CREATE to create files, the CREATE command is useful when you want to:

- Preallocate (set) the size of a file
- Create a file from a command procedure

There is less overhead when you preallocate the size of a file, because the system assigns enough space for the file only once, rather than having to assign more space several times as the file grows larger.

Because preallocation causes less overhead, access to the file is faster. Allocating the size of a file is not necessary, but it is occasionally useful if a program will be using the file. Preallocation avoids possible overhead caused when the file size is extended by the program's use.

You also preallocate the size of a file if you need a contiguous file of a specific size. You must specify /ALLOCATION with the /CONTIGUOUS qualifier.

While you are creating a file with the CREATE command, you can use the DELETE and CTRL/U keys to delete text on the current line. When you are finished entering text, you press CTRL/Z to save the text as a file.

Format

CREATE file-spec

Command Qualifiers	**Defaults**
/ALLOCATION=n	/ALLOCATION=0
/CLUSTER_SIZE=n	
/[NO]CONTIGUOUS	/NOCONTIGUOUS
/POSITION[=n]	/POSITION=0
[=MIDDLE]	
[=INDEX]	
/PROTECTION=n	
/[NO]REPLACE	

Prompts

File: file-spec
OK to replace existing file filespec ?

Command Parameters

File-spec
Specifies the name of the file to be created. You cannot use wildcard characters in the file specification.

Command Qualifiers

/ALLOCATION=n
Creates space on the disk by forcing the initial allocation of the file to a number of 512-character blocks, which you specify with n. The following example provides three blocks of space to create a file named BOSTON.MAR:

```
$ CREATE BOSTON.MAR/ALLOCATION=3 RET
```

/CLUSTER_SIZE=n
Establishes the cluster size for a disk file. A cluster is a number of contiguous blocks taken together as a unit. The cluster size is the minimum unit of allocation for the file.

This qualifier is useful for large files. By specifying a large cluster size, you can speed up random access to the data. See Appendix B for more information on specifying cluster size.

/CONTIGUOUS
/NOCONTIGUOUS
Indicates whether or not the file is to be contiguous, which means that the file occupies consecutive physical disk blocks.

If you do not specify an allocation, or if the allocation is too small, you get the following error message when the system tries to increase the size of the contiguous output file:

```
?Protection violation
```

/PROTECTION=n
Specifies the protection code of the file.

RSTS/E normally assigns files a protection code of 60, unless your system manager has changed the default. If you use the SET PROTECTION/DEFAULT command before creating the file, the protection code becomes the value you specified as the default. See the SET PROTECTION command at the end of this chapter.

/POSITION=n
/POSITION=MIDDLE
/POSITION=INDEX
Specifies the starting position of the target file on disk. This qualifier is ignored for files created on devices other than disks.

If you specify /POSITION with an argument, then that value indicates the starting disk cluster at which to place the file. The minimum value is 0 (beginning of disk) and the maximum value is the disk's maximum cluster number. If you specify 0 (the default), the file is created at the first available cluster on the disk.

If you specify /POSITION=MIDDLE, the file is created at the first available cluster past the middle of the disk.

If you specify /POSITION=INDEX, the file is created at the first available cluster following the storage allocation table (SATT.SYS) on the disk.

/REPLACE
/NOREPLACE

/REPLACE requests that if a file already exists with the same file specification as the one you are creating, the existing file is to be deleted.

For example, if there is a file named BLACK.CAT in your directory, and you want the file you are creating to replace the existing file, type:

```
$ CREATE BLACK.CAT/REPLACE (RET)
```

If the CREATE command is going to replace an existing file, but you do not specify /REPLACE or /NOREPLACE, DCL displays the prompt:

```
OK to replace existing filespec ?
```

You can reply with one of the following:

Y	Yes, replace the file
N	No, do not replace the file
(RET)	No, do not replace the file

/NOREPLACE forces the CREATE command to fail if the file being created already exists. In addition, it suppresses the "OK to replace existing filespec?" prompt.

If you use the CREATE command in a command procedure, you should explicitly specify either /REPLACE or /NOREPLACE. This prevents DCL from issuing the "OK to replace existing filespec?" prompt.

EDIT

The EDIT command starts the EDT editor program, which lets you create and edit text files.

This section provides basic command and syntax information for the EDIT command. To learn about editing text with the EDT editor, see the *Introduction to the EDT Editor*, which is a tutorial manual that you can use at your terminal. For details about each command and feature of EDT, see the *EDT Editor Manual*.

You can also get information about EDT while using it. EDT's help facility provides a description about each of its editing commands. After you use the EDIT command to start EDT, type HELP after the asterisk (*) prompt for help text. (In keypad editing, you press the HELP key for help text. See the *EDT Editor Manual* for details.)

Format

EDIT file-spec

Command Qualifiers	**Defaults**
/COMMAND = file-spec	/COMMAND = EDTINI.EDT
/NOCOMMAND	
/EDT	/EDT
/FORMAT = STREAM	/FORMAT = STREAM
/FORMAT = VARIABLE	
/JOURNAL = [file-spec]	/JOURNAL
/NOJOURNAL	
/OUTPUT = file-spec	/OUTPUT
/NOOUTPUT	
/[NO]READ_ONLY	/NOREAD_ONLY
/[NO]RECOVER	/NORECOVER

Prompts

File: file-spec

Command Parameters

file-spec

Specifies the file to be created or edited using the EDT editor. If the file does not exist, it is created.

The only part of the file specification that you must supply is the file name. EDT does not provide a default file type when creating files; if you do not include a file type, no file type is assigned.

You cannot use wildcard characters in the file specification.

Command Qualifiers

/COMMAND = file-spec
/NOCOMMAND

Controls whether or not EDT reads a command file before prompting at the terminal. If you specify a command file, EDT executes all commands in the file before beginning the editing session. For example, if you have a file named CHANGE.INI containing line editing commands, you use it when editing the file RUN.DAT by typing:

```
$ EDIT RUN.DAT/COMMAND=CHANGE.INI RET
```

By default, EDT looks for a command file named EDTINI.EDT when you begin an editing session. If the file EDTINI.EDT is in your directory on the public structure, EDT uses the commands it contains when the session begins. If you specify the /NOCOMMAND qualifier, EDT does not read a command file before beginning the editing session. To suppress the effects of the command file EDTINI.EDT when you start an editing session with file RUN.DAT, type:

```
$ EDIT RUN.DAT/NOCOMMAND RET
```

Otherwise, the commands in the EDTINI.EDT file automatically affect your editing session.

You cannot use wildcard characters in the command file specification.

/EDT

Makes sure that the EDIT command starts EDT, if your system manager has changed the default on your system from EDT to another editor. Otherwise, you do not need to use this qualifier.

For example:

```
$ EDIT/EDT NEWFIL.RNO RET
```

The EDT editor is invoked, so you can produce a file named NEWFIL.RNO.

/FORMAT = STREAM
/FORMAT = VARIABLE

Determines the format of the output file.

/FORMAT = STREAM creates the file in stream ASCII format. The default is /FORMAT = STREAM. /FORMAT = VARIABLE creates the file in variable-length format. See the *RMS-11 User's Guide* for information about stream and variable file formats.

/JOURNAL[= file-spec]
/NOJOURNAL

Controls whether or not a journal file is created for the editing session. A journal file contains a copy of the edits you make to another file, from the beginning to the end of the editing session. If your editing session ends abnormally (as in a

system failure or "crash"), you can restart EDT (using the /RECOVER qualifier) and reinstate all commands from the aborted session.

When you begin an editing session, EDT automatically creates a journal file with the same name as the input file and a .JOU file type. If you are editing a file named RUN.DAT, for example, the journal file is named RUN.JOU.

The command line to state explicitly that you want a journal file when you edit RUN.DAT, and to name the journal file RUNDAT.V01 is:

```
$ EDIT RUN.DAT/JOURNAL=RUNDAT.V01 (RET)
```

If you specify the /NOJOURNAL qualifier, no journal file is created.

If you omit the /JOURNAL qualifier or if you specify the qualifier without a file specification, the editor creates a journal file with the same file name as your input file and a default file type of .JOU. EDT also assumes a default file type of .JOU if you specify a file name but not a file type when you use the /JOURNAL qualifier.

Your directory contains the journal file when the system resumes operation after a system crash. By default, EDT places the journal file in the same directory and device as the output file, if you specified /OUTPUT when creating the file. Otherwise, EDT uses the same directory and device as the input file.

For example, suppose you start a session to edit the existing file MARY.DAT, but you want the finished output file to become file BETH.DAT:

```
$ EDIT MARY.DAT/OUTPUT=BETH.DAT (RET)
```

If the system crashes, the journal file is named BETH.JOU. You can then recover from a system crash by typing:

```
$ EDIT MARY.DAT/OUTPUT=BETH.DAT/RECOVER (RET)
```

If you used the /OUTPUT qualifier to invoke EDT, you must also use it when you recover the file. (The journal file does not "remember" the command line you typed to begin the editing session.)

You cannot use wildcard characters in the journal file specification.

You cannot use /NOJOURNAL if you specify /NOREAD_ONLY.

/OUTPUT = file-spec
/NOOUTPUT

Defines the file specification of the file created during the editing session. If you do not specify the /OUTPUT qualifier, the output file replaces the input file, and the previous version of the input file is saved as filename.BAK. EDT deletes any previous file named filename.BAK.

For example, suppose you type:

```
$ EDIT PAPER.NUM RET
```

If you continue the editing session and then exit from EDT by saving your edits, EDT automatically creates a new output file named PAPER.NUM. The file you began the session with is renamed to PAPER.BAK.

If you want the output file to be named something other than the original file name, you can specify the output file name explicitly. For example, the following command line tells EDT to put the contents of the file you are about to update into a file named SUPPLY.DAT:

```
$ EDIT PAPER.NUM/OUTPUT=SUPPLY.DAT RET
```

You cannot use wildcard characters in the file specification.

You can suppress the creation of the output file with the /NOOUTPUT qualifier. For example:

```
$ EDIT PAPER.NUM/NOOUTPUT RET
```

Use of the /NOOUTPUT qualifier does not suppress creation of the journal file.

You cannot use /OUTPUT if you specify /READ_ONLY.

/READ_ONLY
/NOREAD_ONLY
Controls whether EDT disables journaling and the creation of an output file when you view a file. The /READ_ONLY qualifier is equivalent to specifying /NOOUTPUT and /NOJOURNAL. You can use the /READ_ONLY qualifier to examine files without modifying them.

The default is /NOREAD_ONLY, which allows journaling and output.

/RECOVER
/NORECOVER
Determines whether EDT reads commands from a journal file before starting the editing session. The /RECOVER qualifier is useful if you are editing a file and the system crashes, as in the case of a power failure. The /RECOVER qualifier lets you restore your work on the file. The default is /NORECOVER, which tells EDT not to recover the edits you made to the file.

For example, if you are editing file PAPER.NUM when the system ceases operations, or crashes, you can use the /RECOVER qualifier to restore your work when the system is back in operation:

```
$ EDIT PAPER.NUM/RECOVER RET
```

If you named the journal file something other than PAPER.JOU (for example, to REAMS.DAT) when you began the previous editing session, use the following command line when restoring your work:

```
$ EDIT PAPER.NUM/RECOVER/JOURNAL=REAMS.DAT RET
```

Displaying File Names and Files

The DIRECTORY command displays information about files. Use the TYPE command to display the contents of individual files.

DIRECTORY

The DIRECTORY command displays information about files.

Format

DIRECTORY [file-spec[,...]]

Command Qualifiers	**Defaults**
/BEFORE=date	
/BRIEF	
/CREATED	/CREATED
/DATE[=CREATED]	/NODATE
[=MODIFIED]	
[=ALL]	
/NODATE	
/FULL	
/MODIFIED	/CREATED
/OUTPUT=file-spec	
/[NO]PROTECTION	/PROTECTION
/SINCE=date	
/SIZE[=ALLOCATION]	/SIZE=USED
[=USED]	
/NOSIZE	/SIZE
/TOTAL	

If you type the DIRECTORY command and press the RETURN key, DCL lists the files in your directory on the public structure. For example:

```
$ DIRECTORY (RET)

 Name .Typ    Size     Prot     Name .Typ    Size     Prot   SY:[52,20]
FINISH.COB     130    < 60>   TEN    .K        4    < 60>
START .BAS      89    < 60>   ROAD   .MAP      3    < 60>
TEMP16.TMP     258    < 60>

Total of 484 blocks in 5 files in SY:[52,20]
```

You can see directory listings only of files to which you have access (read, write, or execute). If you specify a file or directory which is not yours, and to which you do not have access, one of the following error messages is displayed:

```
?File does not exist
?Can't find file or account
```

Command Parameters

file-spec[,...]

Specifies one or more files to be listed. If your installation has DECnet/E, you can also specify the node on which the files are located. The default is the local node.

The rules for using the file specification are:

- If you do not enter a file specification, the DIRECTORY command lists all the files in your directory on the system disk.
- If you specify only a device name, the DIRECTORY command lists the files in your directory on that device. For example, if you have files on a disk named DB1:, you can type:

  ```
  $ DIR DB1: RET
  ```

- If you use wildcards for a file name or type, RSTS/E displays all files with that name or type. For example, if your account contains files named RACE.1, RACE.2, and RACE.3, you can type:

  ```
  DIRECTORY RACE.*
  ```

  To get information about all the files in the directories you have access to, type:

  ```
  $ DIRECTORY [*,*]*.* RET
  ```

  This lists all files in all directories on the public disk structure. The first set of asterisks (in [*,*]) corresponds to every account number. The second set of asterisks (*.*) represents every file name and type.

- If a file specification contains a file name and a wildcard for the file type, the DIRECTORY command assumes all file types. Similarly, you can specify a file type and a wildcard for the file name. Therefore, both of the following are valid command lines:

  ```
  $ DIRECTORY RACE.* RET

  $ DIRECTORY *.1 RET
  ```

  If you provide more than one file specification, separate the file specifications with commas (,) or plus signs (+). You can use wildcard characters in the directory specification, file name, or file type of a file specification to list all files that satisfy the elements you specify. For example:

  ```
  $ DIRECTORY RACE.1,**.DAT,DB1:*.* RET
  ```

Command Qualifiers

/BEFORE=date

Specifies that only those files created or modified earlier than a given date be listed. See Chapter 2 for the syntax to specify dates.

You can use the /BEFORE qualifier with either the /CREATED or /MODIFIED qualifier. For example:

```
$ DIRECTORY/CREATED/BEFORE=28-JUN-85 (RET)

$ DIRECTORY/MODIFIED/BEFORE=TODAY (RET)
```

If you omit the /BEFORE qualifier, RSTS/E displays the files without regard to their dates.

You can use /BEFORE only when listing files on the local (your own) node.

/BRIEF

Lists only the file name and type of each file. If you specify /BRIEF, you do not see the number of blocks for the files or their protection codes. For example:

```
$ DIRECTORY/BRIEF (RET)

Name .Typ     Name .Typ     Name .Typ    Name .Typ
SY:[52,20]
ROAD  .MAP   TEMP25.TMP   STRIDE.TXT    PACE  .DAT

4 files 25 blocks
```

If you do not specify /BRIEF, the sizes and protection codes are also included in the display.

/CREATED

Selects the files according to their dates of creation. Use the /CREATED qualifier only with the /BEFORE or /SINCE qualifiers. For example:

```
$ DIRECTORY/BEFORE=10-JUN-85/CREATED (RET)

Name .Typ   Size    Prot    Name .Typ   Size     Prot  SY:[52,20]
ROAD  .MAP    4    < 60>    STRIDE.TXT   21     < 40>
PACE  .DAT    8    < 60>

Total of 33 blocks in 3 files in SY:[52,20]
```

Do not use /CREATED with the /MODIFIED qualifier. (You can select files either according to their date of creation or date of modification, but not both.) After a file is created, any later updates cause the file to be modified.

By default, the selection of files according to a date uses the creation date.

You can use /CREATED only when listing files on the local (your own) node.

/DATE=CREATED
/DATE=MODIFIED
/DATE[=ALL]

The /DATE qualifier causes the DIRECTORY listing to include the date when a file was created or last accessed, or both. If you specify /DATE without an argument, /DATE=ALL is the default.

/DATE=CREATED requests information about the file's creation date and time:

```
$ DIRECTORY/DATE=CREATED (RET)

Name .Typ   Size   Prot    Date      Time      SY:[52,20]
ROAD .MAP   4    < 60>  14-Jun-85  10:13 AM

Total of 4 blocks in 1 file in SY:[52,20]
```

/DATE=MODIFIED displays the date when the file was last updated or accessed:

```
$ DIRECTORY/DATE=MODIFIED (RET)

Name .Typ   Size  Prot   Access    Name .Typ   Size  Prot    Access
SY:[52,20]
ROAD .MAP   4   < 60>  08-Jun-85  STRIDE.TXT   21  < 40>  14-Jun-85

Total of 25 blocks in 2 files in SY:[52,20]
```

/DATE=ALL provides date and time information about the creation and access of a file:

```
$ DIRECTORY/DATE=ALL (RET)

Name .Typ   Size   Prot    Access    Date      Time      SY:[52,20]
ROAD .MAP    4   < 60>  08-Jun-85 14-Jun-85  04:25 PM

Total of 4 blocks in 1 file in SY:[52,20]
```

/NODATE

Lists information about files but does not include dates. Unless you specify /DATE, the /NODATE qualifier is assumed. For example:

```
$ DIRECTORY/NODATE (RET)

Name .Typ    Size   Prot     Name .Typ   Size   Prot     SY:[52,20]
ROAD .MAP     4   < 60>    STRIDE.TXT    21   < 40>

Total of 25 blocks in 2 files in SY:[52,20]
```

You can use /DATE and /NODATE only when listing files on the local (your own) node.

/FULL

Lists the following items for each file:

- Name
- Type
- Number of blocks used
- Protection
- Date last accessed or modified
- Creation date and time
- Cluster size
- Run-time system name
- Position of first block in file
- File attributes, if any

The /FULL qualifier implies the qualifiers /DATE=ALL and /PROTECTION as defaults. The /FULL qualifier also includes information about the file's cluster size (in the Clu column) and position on the disk (in the Pos column). Often the /FULL qualifier is used with a file specification. For example:

```
$ DIRECTORY/FULL RACE.* RET

  Name.Typ     Size  Prot    Access     Date      Time      Clu     RTS  Pos
  SY:[52,20]
  RACE  .1     4     <  60> 12-Sep-85 12-Sep-85 01:02 PM    4 ...RSX 22699
  RACE  .2     3     <  60> 25-Sep-85 25-Sep-85 09:55 AM    4 ...RSX 23284

Total of 7 blocks in 2 files on SY:[52,20]
```

/MODIFIED

Selects files according to the last date the file was modified. Use this qualifier only with the /BEFORE or /SINCE qualifiers. For example:

```
$ DIRECTORY/MODIFIED/BEFORE=12-JUN-85 RET

Name .Typ     Size    Prot     Name .Typ  Size    Prot    SY:[52,20]
ROAD  .MAP      4    <  60>   STRIDE.TXT    21   <  40>

Total of 25 blocks in 2 files in SY:[52,20]

$ DIRECTORY/MODIFIED/SINCE=18-JUN-85

Name .Typ     Size    Prot     Name .Typ  Size    Prot    SY:[52,20]
ROAD  .MAP      4    <  60>   STRIDE.TXT    21   <  40>
RACE  .1        3    <  60>

Total of 28 blocks in 3 files in SY:[52,20]
```

Note that this display lists only the file name and type, file size, and protection code. Use /DATE=MODIFIED to display the dates of file access.

Do not use /MODIFIED with the /CREATED qualifier. (The date you specify can apply only to the creation or modified date; thus, you cannot specify both /CREATED and /MODIFIED.) After a file is created, any later updates cause the file to be modified.

You can use /MODIFIED only when listing files on the local (your own) node.

/OUTPUT=file-spec

Creates a file containing the output of a DIRECTORY listing, instead of displaying the output at your terminal. You must include a file name with the /OUTPUT qualifier. If you omit the file type in the file specification, the default type is .DIR.

For example, the following command line puts the directory listing into a file named RACES.DIR:

```
$ DIRECTORY/OUTPUT=RACES RET

$ TYPE RACES.DIR RET


Name .Typ   Size   Prot   Name .Typ  Size   Prot   SY:[52,20]
ROAD  .MAP    4   < 60>   STRIDE.TXT  21   < 40>
RACE  .1      3   < 60>

Total of 28 blocks in 3 files in SY:[52,20]

$ DIRECTORY RACES RET

Name .Typ   Size   Prot   Name .Typ  Size  Prot   SY:[52,20]
RACES.DIR     2   < 60>

Total of 2 blocks in 1 file in SY:[52,20]
```

Note that the output file has the type .DIR.

You cannot use wildcard characters in the output file specification.

/PROTECTION
/NOPROTECTION

Specifies the protection code of files to be in the directory listing.

You can use this qualifier only when listing files on the local (your own) node.

/SINCE=date

Specifies that only those files created or modified on or after a specified date be displayed.

You normally use the /SINCE qualifier with either the /CREATED or /MODIFIED qualifier. If you omit the /SINCE qualifier, you obtain all the files, regardless of creation date. The date can be either TODAY or YESTERDAY, or a date in the format dd-mmm-yy or yy.mm.dd (as in 12-MAY-85 or 85.5.12). For example:

```
$ DIRECTORY/CREATED/SINCE=06-APR-85 (RET)

Name .Typ   Size   Prot    Name .Typ  Size   Prot    SY:[52,20]
ROAD .MAP    4   < 60>    STRIDE.TXT  21   < 40>
RACE .1      3   < 60>

Total of 28 blocks in 3 files in SY:[52,20]
```

You can use /SINCE only when listing files on the local (your own) node.

/SIZE=ALLOCATION
/SIZE=USED

The /SIZE qualifier lists the number of blocks a file either uses or is allocated. Compare the following two examples:

```
$ DIR *.LOG/SIZE=USED (RET)

Name .Typ  Size   Prot    Name .Typ  Size   Prot    SY:[52,20]
BAR  .LOG   3   < 60>    FOO  .LOG   6   < 40>
LINK .LOG  12   < 60>    HELP .LOG   6   < 60>

Total of 27 blocks in 4 files in SY:[52,20]

$ DIR *.LOG/SIZE=ALLOCATION (RET)

Name .Typ  Size   Prot    Name .Typ  Size   Prot    SY:[52,20]
BAR  .LOG   4   < 60>    FOO  .LOG   8   < 40>
LINK .LOG  12   < 60>    HELP .LOG   8   < 60>

Total of 32 blocks allocated in 4 files in SY:[52,20]
```

The first example shows /SIZE=USED, which is the default. The Size column shows the number of blocks used by each file. By contrast, the Size column in the second example shows the number of blocks allocated to store each file. This may be larger because disk space is allocated in multiples of the file's cluster size.

You can use /SIZE only when listing files on the local (your own) node.

/NOSIZE

The /NOSIZE qualifier suppresses information on file size. The default is /SIZE.

/TOTAL

The /TOTAL qualifier displays the total amount of space your files require, without listing information about each file. For example:

```
$ DIRECTORY/TOTAL (RET)

SY:[52,20]

Total of 647 blocks in 24 files in SY:[52,20]
```

You can use /TOTAL only when listing files on the local (your own) node.

TYPE

The TYPE command displays the contents of a text file (as opposed to a binary or temporary file) at the terminal.

Format

TYPE file-spec[,...]

Command Qualifiers	Defaults
/BEFORE=date	
/CREATED	
/MODIFIED	
/[NO]QUERY (N)	/NOQUERY
/SINCE=date	

Prompts

File: file-spec[,...]

For example, you can display the contents of the file TRAIN.TXT:

```
$ TYPE TRAIN.TXT (RET)

Even though it was difficult to
undertake a new running regime,
I was amazed at how soon I was
able to increase my daily mileage.
```

You can control the display of a file in any of the following ways:

- To temporarily halt the output, use CTRL/S. To resume output where it was interrupted, use CTRL/Q. This works only if your terminal has the TTSYNC characteristic set. (See Chapter 5 for more information on setting terminal characteristics.) On VT100 terminals you can also press the NO SCROLL key to stop and restart output. On VT200 terminals you can press the HOLD SCREEN key to stop and restart output.
- To suppress the display but continue command processing, use CTRL/O. If you press CTRL/O again before processing is completed, output resumes at the current point in command processing.
- To stop command execution entirely, press CTRL/C. The use of CTRL/C returns you to DCL command level.

Command Parameters

file-spec[,...]
Specifies one or more files to be displayed. If your installation has DECnet/E, you can specify the node where the files are located. You can also use wildcards.

If you want to display more than one file, separate the file specifications with commas (,) or plus signs (+). In either case, RSTS/E displays the files in the order you specify. The TYPE command allows wildcards in place of the directory, file name, and file type fields. The following command line contains wildcards and commas:

```
$ TYPE *.TXT,ROAD.MAP,RACE.* RET
```

This command line displays the contents of all files having the file type .TXT, the file ROAD.MAP, and all files named RACE.

Command Qualifiers

/BEFORE=date
/SINCE=date
/CREATED
/MODIFIED
Displays files created or modified before or since the specified date. See Chapter 2 for information on specifying dates and times.

The following example displays all data files created before 01-Jun-85:

```
$ TYPE/CREATE/BEFORE=01-Jun-85 *.DAT OLDWRK.DAT RET
```

The following example display all files that were modified today:

```
$ TYPE/MODIFIED/SINCE=TODAY *.* NUWORK.WED RET
```

/QUERY
/NOQUERY
When you use the /QUERY qualifier, the system displays the name of each file you request and asks if you want to see the file. (The /QUERY qualifier is useful only when you use wildcard file specifications in the TYPE command line.)

For example, if you have eight files in your directory with the file type .TXT, you can type:

```
$ TYPE *.TXT/QUERY RET
```

DCL prints a file specification followed by a question mark (?) prompt. You can reply by typing one of the following responses:

Y	Yes, display the file
N	No, do not display the file
RET	No, do not display the file
CTRL/Z	Quit; return to command level

For example:

```
[52,20]TRAIN .TXT? N RET
[52,20]STRIDE.TXT? Y RET

Hill training taught me that
short, quick steps are good for
running uphill.  Longer strides
are best for downhill inclines
and straightaways.

[52,20]PACE  .TXT? CTRL/Z
$
```

Deleting Files: DELETE

The DELETE command permanently removes a file from a directory.

Format

DELETE file-spec[,...]

Command Qualifiers	**Defaults**
/BEFORE=date	
/CREATED	/CREATED
/ERASE	
/[NO]LOG (N)	/LOG
/MODIFIED	
/[NO]QUERY (N)	/NOQUERY
/SINCE=date	

Prompts

Files: file-spec[,...]

For example, if you want to delete the file BETH.DAT, type:

```
$ DELETE BETH.DAT (RET)
```

Command Parameters

file-spec[,...]

Specifies the name of one or more files to be deleted. If your installation has DECnet/E, you can specify the node where the files are located.

Each file specification must contain a file name and a file type. You can specify wildcard characters for the directory, file name, or file type. For example:

```
$ DELETE DB2:RACE.* (RET)
```

To delete more than one file, separate the file specifications with commas (,) or plus signs (+). If you omit the account specification or device name, RSTS/E uses your directory and the system disk. For example:

```
$ DELETE ROAD.MAP,*.TEXT+STRIDE.*(RET)
```

Command Qualifiers

/BEFORE=date

Specifies that only the files dated earlier than a particular date be deleted. The /BEFORE qualifier is useful only when you include wildcard characters in the file specification.

Use the /CREATED or /MODIFIED qualifiers in conjunction with /BEFORE to request that RSTS/E delete only files created or modified before the specified date. If you do not specify either of these qualifiers, /CREATED is the default.

For example, suppose you have files named RESULT.A, RESULT.B, and RESULT.C in your directory, and you created file RESULT.C today. To delete the files created before today, type:

```
$ DELETE/BEFORE=TODAY RESULT.* RET
RESULT.A deleted
RESULT.B deleted
```

You can use /BEFORE only when deleting files on the local (your own) node.

/CREATED

Specifies that only files created within a defined time period be deleted. Use the /CREATED qualifier with either the /BEFORE or /SINCE qualifiers.

For example, if you do not want to save any of the files you created today, type:

```
$ DELETE/CREATED/SINCE=TODAY RET
```

If you do not specify /MODIFIED or /CREATED, the default is /CREATED.

You can use /CREATED only when deleting files on the local (your own) node.

/ERASE

Specifies that you want to zero the file prior to deleting it. This qualifier is useful if you want to erase a file for security purposes.

For example, to erase the file JIM.LOG, type:

```
$ DELETE/ERASE JIM.LOG RET
JIM.LOG erased and deleted
```

/LOG
/NOLOG

Controls whether the DELETE command displays the file specification of each file after its deletion.

The default is /LOG; normally the DELETE command displays the names of files after it deletes them. For example:

```
$ DELETE RISTUS.ONE RET
RISTUS.ONE deleted
```

You can request the deletion message explicitly with /LOG:

```
$ DELETE/LOG PETER.WRK RET
PETER .WRK deleted
```

Note that the /NOLOG qualifier suppresses the usual message:

```
$ DELETE/NOLOG ROAD.MAP RET
```

/MODIFIED

Specifies that only files modified within a defined time period be deleted. Use the /MODIFIED qualifier with either the /BEFORE or /SINCE qualifiers. If you do not specify /MODIFIED, the default is /CREATED.

For example, if you want to delete all files that have not been modified since June 11, type:

```
$ DELETE/MODIFIED/BEFORE=11-JUN RET
```

A file's revision date is updated whenever the file is modified. (Your system manager can change the default, so that the file is updated whenever the file is read or modified.)

You can use /MODIFIED only when deleting files on the local (your own) node.

/QUERY
/NOQUERY

Specifies that you want to select files to be deleted. Use the /QUERY qualifier when you use a wildcard in the command line, so you can avoid deleting files unintentionally. The default is /NOQUERY.

For example, suppose you want to delete some of your files with a .DAT file type. The command line is:

```
$ DELETE *.DAT/QUERY RET
```

DCL prints a file specification followed by a question mark (?) prompt. You can reply by typing one of the following responses:

Y	Yes, delete the file
N	No, do not delete the file
RET	No, do not delete the file
CTRL/Z	Quit; return to command level

The display at your terminal looks similar to:

```
[52,20]WRITE .DAT? N RET
[52,20]MANUAL.DAT? Y RET
MANUAL.DAT deleted
[52,20]PACE  .DAT? CTRL/Z
$
```

/SINCE=date

Specifies that only the files dated on or after a particular date be deleted.

A file is implicitly modified on the date it was created. Therefore, a file's date of modification never exceeds its date of creation.

Use the /CREATED or /MODIFIED qualifier to request that only files created or modified on or after the specified date be deleted. If you do not specify /CREATED or /MODIFIED, the DELETE command deletes all files created since the specified date.

For example, the following command line deletes all files created on or after June 11:

```
$ DELETE/SINCE=11-JUN *.* RET
```

To delete all files named RACE that were modified since April 6, type:

```
$ DELETE/MODIFIED/SINCE=06-APR RACE.* RET
```

You can use /SINCE only when deleting files on the local (your own) node.

Copying, Renaming, and Appending Files

This section describes commands that let you copy, rename, and append the contents of files.

COPY

The COPY command duplicates one or more existing files, or concatenates two or more files. You can use this command for local and network file specifications.

Format

COPY input-file-spec[,...] output-file-spec

Command Qualifiers	**Defaults**
/ALLOCATION=n	
/BEFORE=date	
/BLOCK_SIZE=n	/BLOCK_SIZE=512
/CLUSTER_SIZE=n	
/CREATED	
/[NO]CONTIGUOUS (N)	
/[NO]LOG (N)	/LOG
/MODIFIED	
/[NO]OVERLAY	/NOOVERLAY
/POSITION[=n]	/POSITION=0
[=MIDDLE]	
[=INDEX]	
/PROTECTION=n	
/[NO]QUERY (N)	/NOQUERY
/[NO]REPLACE (N)	
/SINCE=date	

Prompts

From: input-file-spec[,...]
To: output-file-spec
OK to replace existing file file-spec ?

Use COPY to:

- Copy one file to another file
- Merge more than one file into a single file
- Copy a group of files to another group of files

For example, if you have a file named SMITH.MAR and you want to make a copy of it as file JONES.JOE, you type:

```
$ COPY SMITH.MAR JONES.JOE (RET)
```

The following command copies all of your files on the public structure (SY:) to a tape (MT0:). (If the tape is in DOS format, the files are copied to your PPN on the tape.) For example:

```
$ COPY SY:*.* MT0: RET
```

To copy every file on a tape (MT0:) into your directory on the public structure (SY:), type:

```
$ COPY MT0:[*,*]*.* *.* RET
```

If you copy files from ANSI tape to disk, you do not need to include the [*,*] directory specification. (ANSI format tapes do not have directories.)

When you specify more than one input file and a single output file, the COPY command merges all the input files into the output file:

```
$ COPY RABBIT.1,RABBIT.2,RABBIT.3 BUNNY.ALL RET
```

The example merges a copy of files RABBIT.1, RABBIT.2, and RABBIT.3 into one file named BUNNY.ALL.

If you specify more than one input file and give an output file specification that contains a wildcard character, the COPY command creates one output file for each input file:

```
$ COPY HARE.1,HARE.2,HARE.3 BIGFUT.* RET
```

The example copies files HARE.1, HARE.2, and HARE.3 into files named BIGFUT.1, BIGFUT.2, and BIGFUT.3, respectively.

Similarly, the following example copies all files named ONE into files named TWO, with the same file types:

```
$ COPY ONE.* TWO.* RET
```

You can specify multiple input files in any of the following ways:

- Separate input file specifications with commas (,) or plus signs (+). For example:

  ```
  $ COPY ROAD.LST,RACE.DAT,ENTRY.LST RESULT.MRG RET
  ```

  The input files ROAD.LST, RACE.DAT, and ENTRY.LST are copied in that order into an output file named RESULT.MRG.

- Specify wildcard characters in place of the directory specification, file name, or file type of an input file specification. The following example copies all COBOL source files on D:[160,*] to the public structure:

  ```
  $ COPY D:[160,*]*.CBL SY:*.* RET
  ```

The expression D:[160,*] means all directories on disk D: with project number 160. For example, directories D:[160,5],D:[160,25], and D:[160,39]. All COBOL source files (*.CBL) on D:[160,*] are copied to the public structure (SY:) into files of the same name and type. You must have access to each of these files to use this command string.

The COPY command creates multiple output files when you specify multiple input files and do one of the following:

- Include asterisk wildcard characters (*) in the output directory specification, file name, or file type fields. For example:

```
$ COPY RACE1.*,ROAD.MAP DB1:*.* (RET)
```

- Omit the output file name, or omit the file type and its preceding period (either of which is equivalent to specifying a wildcard).

When you specify a wildcard for any part of the output file specification, the COPY command uses the corresponding part of the input file specification.

Command Parameters

input-file-spec[,...]

Specifies one or more input files to be copied. If you specify more than one input file, separate them with either commas (,) or plus signs (+). You can use wildcard characters for the directory, file name, or file type.

The following example uses wildcards and commas:

```
$ COPY *.TXT,RACE.1,RACE.2 FINISH.TXT (RET)
```

This command line copies all files with the type .TXT, and files RACE.1 and RACE.2 into a file named FINISH.TXT.

output-file-spec

Specifies the name of the file into which the input files are to be copied.

You must specify at least one part of the output file specification.

If you do not specify a device or directory, the COPY command uses the public structure and your own directory. For other parts that you do not specify, the COPY command uses the corresponding field of the input file specification.

If you specify an asterisk wildcard character (*) in place of the file name or file type of the output file specification, the COPY command creates one or more output files, based on the input file specification. It uses the corresponding part of the input file specification to name the output file.

For example, assume you have files NAME.1, NAME.2, and NAME.3 in your directory. To copy these files into duplicates named NEW.1, NEW.2, and NEW.3, you type:

```
$ COPY NAME.* NEW.* RET
```

You can use wildcard characters in the directory specification of an output file. For example, the following command copies all files in all directories on the public structure onto DB1:, into the corresponding directory, file names, and file types:

```
$ COPY [*,*]*.* DB1:[*,*]
```

The preceding command line copies all files in all directories onto a disk named DB1:. The file names remain the same because you do not specify an output file name.

The ability to copy files depends on the file protection codes and whether or not you have write access to the directory to which you are copying them.

Command Qualifiers

/ALLOCATION=n

Creates space on the disk by forcing the initial allocation of the output file to a number of 512-byte blocks, which you specify with n. The following example provides three blocks of space to copy the file PAPER.NUM into SUPPLY.DAT:

```
$ COPY PAPER.NUM SUPPLY.DAT/ALLOCATION=3 RET
```

If you do not specify the initial allocation of the output file, then RSTS/E determines it by the size of the input file being copied.

You can use /ALLOCATION only if the input and output files are on the local (your own) node.

/BEFORE=date
/SINCE=date
/CREATED
/MODIFIED

Copies files created or modified before or since the specified date. See Chapter 2 for information on specifying dates and times.

The following example copies all data files created before 01-Jun-85 into a file named OLDWRK.DAT:

```
$ COPY/CREATED/BEFORE=01-Jun-85 *.DAT OLDWRK.DAT RET
```

The following example copies all files that were modified today into a file named NUWORK.WED:

```
$ COPY/MODIFIED/SINCE=TODAY *.* NUWORK.WED RET
```

/BLOCK_SIZE=n

Lets you specify a block size for output to magnetic tape. For example:

```
$ COPY MYFILE.TXT  MM0:MYFILE.TXT/BLOCK_SIZE=2048
```

The argument of the /BLOCK_SIZE=n qualifier must be an even integer in the range of 18 to 4096 bytes. The default is 512 bytes.

If you are copying a large file, a larger block size can be useful because it reduces the number of interblock gaps that are needed; therefore, you can fit more data onto a tape. On the other hand, a small block size is often preferable for a small file, since it reduces the amount of space wasted at the end of the file.

If you specify an invalid value for n, the COPY command prints the error message ?Bad block size.

The /BLOCK_SIZE=n qualifier has the following restrictions:

- The qualifier can be used only for magnetic tape. Its use on disk is ignored.
- The qualifier applies only to output files. The COPY command automatically handles input magnetic tape files with block sizes other than 512 bytes.
- If the output tape is in ANSI format and is intended for interchange with another operating system, the block size must be an even integer between 18 and 2048 bytes. Tapes having block sizes greater than 2048 bytes cannot be read by other operating systems.
- If the output tape is intended for use on the RT-11 operating system, the block size must be 512 bytes.

If both the input and output files are on magnetic tape, and both have large block sizes, the COPY command may fail because of insufficient buffer space. If this happens, transfer the file to disk, and then use the COPY command to transfer the file from disk to tape.

/CLUSTER_SIZE=n

Establishes the cluster size for a disk file. A cluster size is a number of contiguous blocks taken together as a unit. The cluster size is the minimum unit of allocation for the file.

This qualifier is especially useful for large files because specifying a large cluster size speeds random access to the data.

In addition, you can avoid filling up your directory by specifying a large cluster size. (In your directory, the system maintains an internal list of the clusters in each file. If your directory fills up, you will receive the error message ?No room for user on device, even though the disk is not full.)

RSTS/E allows cluster sizes of 1, 2, 4, 8, 16, 32, 64, 128, and 256 blocks. With the SHOW DISK command, you can display the minimum cluster size of each disk. If you specify a cluster size below the minimum, the system uses the disk's minimum and does not display an error message.

/CONTIGUOUS
/NOCONTIGUOUS

Indicates whether or not the output file is to be contiguous. A contiguous file occupies consecutive physical disk blocks.

For example, the following command line causes the file MANUAL.RUN to be stored in consecutive blocks on disk DB2:.

```
$ COPY MANUAL.RUN DB2:*.*/CONTIGUOUS RET
```

By default, the COPY command creates an output file in the same format as the corresponding input file. If an input file is contiguous, the COPY command attempts to create a contiguous output file. If there is not enough contiguous disk space, the system copies the file into available empty disk blocks and displays the message:

```
%File file-spec created noncontiguous
```

The /CONTIGUOUS qualifier should not be used if the output file is not on disk.

If you are concatenating several input files, or if the single input file is on a device other than disk, and you are creating a contiguous output file, you must use the /ALLOCATION qualifier to preallocate the size of the output file. This is necessary because the system cannot judge the size of the output file in these cases.

If you do not specify an allocation, or if the allocation is too small, RSTS/E displays the following error message when the system tries to increase the size of the contiguous output file:

```
?Protection violation
```

If you copy a file from a tape and want the file to be contiguous, you can use two COPY commands: one to copy the file from the tape, and another to create a contiguous file.

For example, assume you want to copy file FRIDAY.WOW from tape MT0: to a contiguous file in your account ([52,20]) on the public structure. You rename the file to WORK.END temporarily, because you cannot copy file FRIDAY.WOW to itself when making it contiguous:

```
$ COPY MT0:FRIDAY.WOW SY:[52,20]WORK.END/LOG RET
[File FRIDAY.WOW copied to SY:[52,20]WORK.END]

$ COPY WORK.END FRIDAY.WOW/CONTIGUOUS/LOG RET
[File SY:[52,20]WORK.END copied to SY:[52,20]FRIDAY.WOW]

$ DELETE WORK.END/LOG RET
WORK.END Deleted
```

The first command copies file FRIDAY.WOW from tape MT0: into a file named WORK.END in your account ([52,20] on the public structure.) The second command copies WORK.END to a contiguous file named FRIDAY.WOW. The third command deletes the file WORK.END, because you used it only to move file FRIDAY.WOW into your account as a contiguous file. (Note that the /LOG qualifier displays each completed file operation.)

If you specify /CONTIGUOUS and there is not enough contiguous disk space, the file is not contiguous and the system prints the error message:

```
?No room for user on device
```

If you specified a network node name on either the input or output file, a different error message displays:

```
?Device full:  Can't create or extend file
```

/LOG
/NOLOG

Controls whether the COPY command displays the file specifications of each file copied. /LOG is the default.

Unless you specify /NOLOG, the COPY command displays the file specifications of the input and output files. For example:

```
$ COPY DR3:[52,20]CREATE.BAS [1,248]*.* RET
[File DR3:[52,20]CREATE.BAS copied to [1,248]CREATE.BAS]
```

/OVERLAY
/NOOVERLAY

Requests that data in the input file be copied into an existing output file, replacing the output file's existing data. The physical location of the file on disk does not change.

For example, assume you rewrote the first chapter (CHAP1.TXT) of a manual (MANUAL.RUN). To overlay the existing chapter with the new first chapter, type:

```
$ COPY CHAP1.TXT MANUAL.RUN/OVERLAY RET
```

If the input file is smaller than the output file, the excess data in the output file is not removed. In such a case (unless you use the /NOLOG qualifier), the COPY command prints a message similar to the following:

```
[File CHAP1.TXT copied into prefix of [1,248] MANUAL.RUN]
```

The default is /NOOVERLAY. In addition, RSTS/E ignores the /OVERLAY qualifier if the output file is written to any device other than a disk.

You can use this qualifier only when the input and output files are on the local (your own) node.

/POSITION=n
/POSITION=MIDDLE
/POSITION=INDEX

Specifies the starting position of the target file on disk. This qualifier is ignored for files copied on devices other than disks.

If you specify /POSITION with an argument, then that value indicates the starting disk cluster at which to copy the file. The minimum value is 0 (beginning of disk) and the maximum value is the disk's maximum cluster number. If you specify 0 (the default), the file is copied at the first available cluster on the disk.

If you specify /POSITION=MIDDLE, the file is copied to the first available cluster past the middle of the disk.

If you specify /POSITION=INDEX, the file is copied to the first available cluster following the storage allocation table (SATT.SYS) on the disk.

/PROTECTION=n

Specifies the protection code of the output file.

You cannot use /PROTECTION with /OVERLAY, or in network operations.

By default, RSTS/E assigns files a protection code of 60, unless your system manager changes the default. Furthermore, if you use the SET PROTECTION/DEFAULT command before the COPY operation, the protection code becomes the value you specified as the default. See the SET PROTECTION command description at the end of this chapter.

You can use /PROTECTION only if the input and output files are on the local (your own) node.

/QUERY
/NOQUERY

Lets you select files to be copied when you specify more than one file. The /QUERY qualifier is useful only if you use wildcards. The default is /NOQUERY.

When you use /QUERY, RSTS/E displays each input file name and a question mark prompt. Enter one of the following responses after each prompt:

Y	Yes, copy the file
N	No, do not copy the file
RET	No, do not copy the file
CTRL/Z	Skip remaining files; return to command level

For example, if you have files STRIDE.TXT, RACE.TXT, and FINISH.TXT in your account, you can use the /QUERY qualifier as follows:

```
$ COPY *.TXT MANUAL.RUN/NOOVERLAY/QUERY (RET)
STRIDE.TXT? Y (RET)
File STRIDE.TXT copied to MANUAL.RUN
RACE  .TXT? N (RET)
FINISH.TXT? (CTRL/Z)
$
```

/REPLACE
/NOREPLACE

The /REPLACE qualifier requests that if a file already exists with the same file specification as the one you are creating, the existing file is to be deleted.

For example, if there is a file named ROAD.MAP in your default directory, and you want the file you are copying to replace the existing file, type:

```
$ COPY ROAD.MAP/REPLACE (RET)
```

If the COPY command is going to replace an existing file, but you do not specify /REPLACE or /NOREPLACE, RSTS/E displays the prompt:

```
OK to replace existing filespec ?
```

You can reply with one of the following:

Y	Yes, replace the file
N	No, do not replace the file
(RET)	No, do not replace the file

A YES response allows you to copy the file and replace the existing file. Otherwise, the system returns to command level.

For example, suppose you mistakenly enter the name of an existing file that you do not want replaced. The display at the terminal reads:

```
$ COPY FOO.TXT SORE.FUT (RET)
OK to replace existing file DR1:[52,20]SORE.FUT? N (RET)
```

The /NOREPLACE qualifier forces the COPY command to fail if the file being created already exists. In addition, it suppresses the "OK to replace existing filespec?" prompt.

If you use the COPY command in a command procedure, you should explicitly specify either /REPLACE or /NOREPLACE. This prevents DCL from issuing the "OK to replace existing filespec?" prompt.

RENAME

The RENAME command changes the file name or type of an existing file.

Format

RENAME old-file-spec[,...] new-file-spec

Command Qualifiers	Defaults
/BEFORE=date	
/CREATED	
/[NO]LOG	/LOG
/MODIFIED	
/PROTECTION=n	
/[NO]QUERY	/NOQUERY
/[NO]REPLACE	/NOREPLACE
/SINCE=date	

Prompts

From: old-file-spec[,...]

To: new-file-spec

For example, you can rename the file JIM.DAT to MORGAN.DAT with the command line:

```
$ RENAME JIM.DAT MORGAN.DAT RET
```

Command Parameters

old-file-spec[,...]

Specifies one or more files whose specifications you want to change. The files must be stored on a disk.

You can use wildcard characters in the directory, file name, or file type of the file specification. If you use wildcards, all the files you specify are renamed (assuming that they are your own files or that you have additional privilege to rename other users' files).

new-file-spec

Provides the new file name or type to be applied to the input file. The RENAME command uses the file name and type of the input file specification to provide defaults for fields you do not specify in the output file. (You cannot change the device or directory of the files.)

You can specify an asterisk (*) in place of the file name or type of the output file specification, or you can omit either one; the RENAME command uses the corresponding field in the input file specification to name the output file. The /QUERY qualifier is useful when you use RENAME with wildcards.

Command Qualifiers

/BEFORE=date
/SINCE=date
/CREATED
/MODIFIED

Renames files created or modified before or since the specified date. See Chapter 2 for information on specifying dates and times.

The following example renames all files created before 01-Jun-85 with a file type of .TXT to files with a file type of .DAT:

```
$ RENAME/CREATE/BEFORE=01-JUN-85 *.TXT *.DAT (RET)
```

The following example renames all files that were modified today to files with a file extension of .WED:

```
$ RENAME/MODIFIED/SINCE=TODAY *.* *.WED (RET)
```

/LOG
/NOLOG

When you rename a file, RSTS/E automatically displays a message to confirm that the file was renamed. You can also request this message explicitly with the /LOG qualifier, although /LOG is the default. The /NOLOG qualifier suppresses the message. For example:

```
$ RENAME RACE1.TXT FUN1.RUN (RET)
RACE1.TXT renamed to FUN1.RUN

$ RENAME RACE2.TXT FUN2.RUN/LOG (RET)
RACE2.TXT renamed to FUN2.RUN

$ RENAME RACE3.TXT FUN3.RUN/NOLOG (RET)
```

/PROTECTION=n

Specifies the protection code to assign to the file to be renamed. By default, the protection codes of the files are not changed.

/QUERY
/NOQUERY

Allows you to select which files you want to rename. The /QUERY qualifier is only useful when you use wildcards in the old file specification. The default is /NOQUERY.

When you use /QUERY, RSTS/E displays individual file names and a question mark prompt. You can reply with:

Y	Yes, rename the file
N	No, do not rename the file
(RET)	No, do not rename the file
(CTRL/Z)	Skip remaining files; return to command level

If the file exists and you do not specify /REPLACE, DCL displays the error message:

```
?Name or account now exists - file file-spec - continuing
```

For example, suppose you want to rename some of your files with a .TXT file type to files with a .DAT file type. You can enter the following command line and then answer the prompts as follows:

```
$ RENAME *.TXT *.DAT/QUERY (RET)
[52,20]RACE1.TXT? Y (RET)
RACE1.TXT renamed to RACE1.DAT
[52,20]STRIDE.TXT? Y (RET)
?Name or account now exists - file STRIDE.DAT - continuing
[52,20]ATLAS.TXT? N (RET)
[52,20]TEXT.TXT? (CTRL/Z)
$
```

/REPLACE
/NOREPLACE

Replaces any existing files with a file to which you assign the same name. The default is /NOREPLACE. The /NOREPLACE qualifier (whether by default or by inclusion) is useful when you know that you need all your files and do not want to risk replacing one.

For example, assume you want to rename the file PAVED.DAT to TRACKS.DAT, and that you no longer need the existing file named TRACKS.DAT. The first of the following command lines assumes the /NOREPLACE qualifier, the second explicitly includes /NOREPLACE, and the third command line replaces the file:

```
$ RENAME PAVED.DAT TRACKS.DAT (RET)
?Name or account now exists - file TRACKS.DAT - continuing

$ RENAME PAVED.DAT TRACKS.DAT/NOREPLACE (RET)
?Name or account now exists - file TRACKS.DAT - continuing

$ RENAME PAVED.DAT TRACKS.DAT/REPLACE (RET)
File PAVED.DAT renamed to TRACKS.DAT
```

APPEND

The APPEND command adds the contents of one or more files to the end of the file you specify. APPEND is similar in syntax and function to the COPY command.

Format

APPEND input-file-spec[,...] output-file-spec

Command Qualifiers	**Defaults**
/BEFORE = date	
/CREATED	
/[NO]LOG (N)	/LOG
/MODIFIED	
/[NO]QUERY (N)	/NOQUERY
/SINCE = date	

Prompts

From: input-file-spec[,...]

To: output-file-spec

For example, assume you create two files and then decide to append the first file to the second one:

```
$ CREATE FILE.A (RET)
This is File A. (RET)
(CTRL/Z)

$ CREATE FILE.B (RET)
And this is File B. (RET)
(CTRL/Z)
```

The APPEND command puts the contents of FILE.A at the end of FILE.B:

```
$ APPEND FILE.A FILE.B (RET)
[File FILE  .A appended to [52,20]FILE  .B]

$ TYPE FILE.B (RET)
And this is File B.
This is File A.
```

FILE.B now includes a copy of FILE.A; the original FILE.A remains unchanged.

Note that you cannot append a file to itself. If you try, RSTS/E displays an error message:

```
$ APPEND MOXIE.DAT MOXIE.DAT (RET)
?Protection violation - file MOXIE .DAT
```

Command Parameters

input-file-spec[,...]

Specifies one or more input files to append to the output file. If your installation has DECnet/E, you can specify the node where the input files are located.

If you specify more than one input file, separate the file specifications with commas (,) or plus signs (+). All files are appended, in the order specified, to the end of the output file.

You can use wildcards in any part of the file specification. For example, to combine files STRIDE.REP, STRIDE.COP, and STRIDE.TXT into an existing file named EVENT.DOC, you type:

```
$ APPEND STRIDE.* RET
To:        EVENT.DOC RET
[File STRIDE.REP appended to [52,20]EVENT.DOC]
[File STRIDE.COP appended to [52,20]EVENT.DOC]
[File STRIDE.TXT appended to [52,20]EVENT.DOC]
```

output-file-spec

Specifies the file to which the input files are to be appended. (The output file must be on a disk.) If your installation has DECnet/E, you can specify the node where the output file is located.

If the output file does not exist, RSTS/E displays the following error messages:

```
?Can't find file or account - file  output-file-spec - continuing
 [File input-file-spec copied to output-file-spec]
```

RSTS/E then creates a new output file.

You must specify at least one part of the output file specification. If you do not specify a device and/or directory, the APPEND command uses your directory on the public disk structure. For other fields that you do not specify, the APPEND command uses the corresponding field of the input file specification.

If you specify the asterisk wildcard character (*) in any part of the output file specification, the APPEND command uses the corresponding field of the related input file specification. For example:

```
$ APPEND ONE.* RET
To: TWO.* RET
[File ONE.TXT appended to [52,20]TWO.TXT]
[File ONE.DAT appended to [52,20]TWO.DAT]
```

Command Qualifiers

/BEFORE = date
/SINCE = date
/CREATED
/MODIFIED

Appends files created or modified before or since the specified date. See Chapter 2 for information on specifying dates and times.

The following example appends all data files created before 01-Jun-85 to a file named OLDWRK.DAT:

```
$ APPEND/CREATED/BEFORE=01-Jun-85 *.DAT OLDWRK.DAT (RET)
```

The following example appends all files that were modified today to a file named NUWORK.WED:

```
$ APPEND/MODIFIED/SINCE=TODAY *.* NUWORK.WED (RET)
```

/LOG
/NOLOG

Controls whether the APPEND command displays the file specifications of each file appended. Unless you specify /NOLOG, the APPEND command displays the file specifications of the input and output files after each append operation. For example:

```
$ APPEND ATHENS.TXT OHIO.TXT/LOG (RET)
[File ATHENS.TXT appended to [52,20]OHIO.TXT]

$ APPEND ATHENS.TXT MASS.TXT (RET)
[File ATHENS.TXT appended to [52,20]MASS.TXT]

$ APPEND ATHENS.TXT MAINE.TXT/NOLOG (RET)
```

/QUERY
/NOQUERY

Determines whether to append individual files to other files. The /QUERY qualifier is only useful when you use wildcards in the input file specification. The default is /NOQUERY.

When you use the /QUERY qualifier, DCL displays a file name followed by a question mark prompt. You can reply with either:

Y	Yes, append the file
N	No, do not append the file
(RET)	No, do not append the file
(CTRL/Z)	Skip remaining files; return to command level

For example, suppose you want to append some of your files with the type .DAT to a file named RACES.TXT:

```
$ APPEND *.DAT TO RACES.TXT/QUERY (RET)
RACE1.DAT? Y (RET)
[RACE1.DAT appended to [52,20]RACES.TXT]
RACE2.DAT? N (RET)
RACE3.DAT? (CTRL/Z)
$
```

Sorting Files and Merging Sorted Files

The commands in this section let you reorder the contents of a file or files, and combine several sorted files into a single output file.

SORT

The SORT command invokes the PDP-11 SORT utility to reorder the records in one through ten files into a defined sequence and to create a single new file containing the reordered records.

This section provides basic command and syntax information for the SORT command. For complete details on the qualifiers and additional information on how to define and control SORT operations, see the *PDP-11 SORT/MERGE User's Guide*.

Format

SORT/qualifiers input-file-spec[,...]/qualifiers -
output-file-spec/qualifiers

Command Qualifiers

/KEY
/[NO]STABLE
/[NO]DUPLICATES
/PROCESS
/COLLATING_SEQUENCE
/[NO]STATISTICS
/WORK_FILES
/SPECIFICATION

Input File Qualifiers

/FORMAT
/INDEXED_SEQUENTIAL
/[NO]SHAREABLE
/TREE_SPACE

Output File Qualifiers

/FORMAT
/SEQUENTIAL
/RELATIVE
/INDEXED_SEQUENTIAL
/[NO]OVERLAY
/ALLOCATION
/[NO]CONTIGUOUS
/LOAD_FILL
/BUCKET_SIZE

Prompts

File: input-file-spec[,...]/qualifiers
Output: output-file-spec/qualifiers

Command Parameters

input-file-spec[,...]

Specifies the names of the file(s) whose records are to be sorted. If the file specifications do not include a file type, SORT assumes the default file of .DAT.

output-file-spec

Specifies the name of the file into which the sorted records are to be written. Its qualifiers can request characteristics for the sorted output file. The command string may have only one output file specification.

If the output file specification does not include a file type, SORT assigns the default file type of .DAT to the output file, regardless of the file type(s) of the input file(s).

Command Qualifiers

/KEY = (field[,...])

Defines a SORT key. You can specify this qualifier up to ten times to define ten different key fields to be sorted.

The /KEY qualifier consists of two required arguments that define the position and size of the key field within the record, and several optional arguments that define the type of data within the key field. You must separate the arguments with commas and enclose them in parentheses.

Required Arguments for the /KEY Qualifier

POSITION:n — Specifies the position of the key within each record, where the first character of the record is position 1.

SIZE:n — Specifies the length of the SORT key in characters, bytes, or digits, depending on the key field data type. The total of all key field sizes must be less than 255 bytes. The valid sizes, based on data type, are:

Data Type	Values for n
CHARACTER	1-255
BINARY	1, 2, 4, 8
DECIMAL	1-31

Note

You cannot use the SIZE argument with floating point data types (D_FLOATING, F_FLOATING, or ASCII_FLOATING.

Optional Arguments for the /KEY Qualifier

CHARACTER
BINARY
DECIMAL
ASCII_FLOATING
ASCII_ZONED
DIBOL_ZONED
D_FLOATING
F_FLOATING
PACKED_DECIMAL

Indicates the type of data in the SORT key field. If not specified, SORT assumes that data is CHARACTER.

SIGNED UNSIGNED	Indicates whether a binary or decimal key is to be compared as a signed or an unsigned integer. UNSIGNED causes all numbers to be treated as positive. SIGNED is the default.
LEADING_SIGN TRAILING_SIGN	Indicates whether the sign of a decimal data type key appears at the beginning or end of the key. If you specify the key data type as DECIMAL, but do not specify either of these subqualifiers, the default is TRAILING_SIGN.
OVERPUNCHED_SIGN SEPARATE_SIGN	Indicates whether the sign of a decimal data type key is superimposed on the decimal value or is separated from the decimal value. If you specify the key data type as DECIMAL, but do not specify either of these subqualifiers, the default is OVERPUNCHED_SIGN.
ASCENDING DESCENDING	Indicates whether the key is to be sorted into ascending or descending order. If you specify neither of these subqualifiers, the default is ASCENDING.

/COLLATING_SEQUENCE=sequence

Specifies the collating sequence in which records are to be arranged. The sequence may be ASCII, EBCDIC, or MULTINATIONAL. The default is /COLLATING_SEQUENCE=ASCII.

/STABLE
/NOSTABLE

Specifies how input files containing records with equal keys are to be sorted in the output file. The default is /NOSTABLE, which causes any records with equal keys to be grouped together in the output file, possibly resulting in an unpredictable sort order for those particular records. The /STABLE qualifier specifies that any records with equal keys are to be directed to the output file in the order in which they were input to SORT. See the following /[NO]DUPLICATES qualifier discussion.

/DUPLICATES
/NODUPLICATES
Alternatively specifies how input files containing records with equal keys are to be sorted in the output file (see the previous /[NO]STABLE qualifier discussion). The /NODUPLICATES qualifier specifies that only one record be retained when SORT encounters records with equal keys. The default is /DUPLICATES.

Note

Do not use /NODUPLICATES and /STABLE in the same SORT operation.

/PROCESS = type
Defines the type of sort. You can specify one of the following sort types:

RECORD	Requests SORT to resequence the entire contents of the input file(s) and create an output file containing the reordered records.
TAG	Requests SORT to sort only the record keys, and then reaccess the input files to create an output file containing the resequenced records.
ADDRESS	Requests SORT to produce an address file sorted by record keys. The output file can be read as an index to read the original file in the desired sequence.
INDEX	Requests SORT to produce an address file containing the key field of each data record and a pointer to its location in the input file. The output file can be read to randomly access the data in the original file in the desired sequence.

The default is /PROCESS = RECORD.

/STATISTICS
/NOSTATISTICS
Specifies whether a statistical summary displays after the SORT operation completes. The default is /NOSTATISTICS.

/WORK_FILES = n
Specifies the number of temporary work files to be used during the sort process. You can specify 0 or any value from 3 through 10. A value of 0 indicates that no work files are necessary because the data will fit in physical memory.

/SPECIFICATION[=file-spec]
Indicates that commands and file qualifiers, including key field definitions, are contained in a specification file. See the *PDP-11 SORT/MERGE User's Guide* for complete details on using specification files.

Input File Qualifiers

/FORMAT=(file-attribute[,...])
Specifies attributes of the input file to override the existing data that SORT normally obtains through PDP-11 RMS. You can specify one or both of the following arguments:

FILE_SIZE:n	Defines the size of the file in blocks. This argument is required for files that are not on disk or magnetic tape. If no input file size is specified, SORT uses the default value of 1000 blocks.
RECORD_SIZE:n	Specifies, in bytes, the length of the longest record, overriding the record size defined in the file header or label. This argument is required for files that are not on disk or magnetic tape, or if the longest record size (LRL) is unavailable or known to be inaccurate.

/INDEXED_SEQUENTIAL[=n]
Specifies that the organization of the input file is indexed-sequential. You specify the number of keys (1 through 255) to make sure that SORT allocates sufficient RMS space. The default value is 1.

/SHAREABLE
/NOSHAREABLE
Specifies whether the input files are opened in write-shareable mode. Use this qualifier when you want to sort files that may be updated by another user during the sort operation. The default is /NOSHAREABLE.

TREE_SPACE=n
Specifies the percentage distribution of available work area between SORT/MERGE tree-related data structures and I/O-related data structures. You can specify from 1 to 100 percent (%). For SORT, the default division of work area is 55% to the tree and 45% to I/O.

Output File Qualifiers

/FORMAT=(record-format[,...])

Defines the output file record format. You can specify one or more of the following arguments:

FIXED[:n] VARIABLE[:n] RMS_STREAM[:n] STREAM[:n] CONTROLLED[:n]	Defines the output file record format and size, where n is the length of the longest record in the output file. If n is not specified, the default is a length long enough to hold the longest output record.
BLOCK_SIZE=n	Specifies, when the output file is directed to disk or magtape, the block size in bytes. If one or more input files is a tape file, the output file block size defaults to the maximum of the block sizes of all tape input files. If the input file is a disk file, the default value is 512 bytes.

/SEQUENTIAL

Specifies that the organization of the output file is sequential. This is the default for an ADDRESS or INDEX sort process; for a RECORD or TAG sort process, the output file organization defaults to the organization of the input file.

/RELATIVE

Specifies that the organization of the output file is relative. By default, a RECORD or TAG sort process results in an output file that has the same organization as the input file. Use the /RELATIVE qualifier to create a relative output file from a sequential or indexed-sequential input file

/INDEXED_SEQUENTIAL[=n]

Specifies that the organization of the output file is indexed-sequential. You specify the number of keys (1 through 255) to make sure that SORT allocates sufficient RMS space. The default value is 1.

/OVERLAY
/NOOVERLAY

Specifies whether an existing file is to be overlaid with the sorted records of the input file. The default is /NOOVERLAY, which means a new output file is created and does not overlay an existing file.

/ALLOCATION=n

Specifies the number of 512-byte blocks to be allocated for the output file. By default, SORT allocates blocks based on the number of records sorted.

/CONTIGUOUS
/NOCONTIGUOUS
Specifies whether the allocation of disk space for the output file is to be contiguous. If you specify /CONTIGUOUS, you must also specify /ALLOCATION to define the number of blocks to allocate for the output file. The default is /NOCONTIGUOUS.

/LOAD_FILL
Used only with indexed files, this qualifier loads the buckets according to the fill size established when the file was created.

/BUCKET_SIZE=n
Used with disk files, this qualifier specifies the number of 512-byte blocks per bucket for the output file. The maximum size you can specify is 32 blocks. If you do not specify a bucket size, the bucket size of the output file defaults to that of the input file.

MERGE

The MERGE command invokes the PDP-11 MERGE utility to combine two through ten similarly sorted input files and create a single output file. Note that input files to be merged must be in sorted order.

This section provides basic command and syntax information for the MERGE command. For complete details on the qualifiers and additional information on how to define and control MERGE operations, see the *PDP-11 SORT/MERGE User's Guide.*

Format

MERGE/qualifiers input-file-spec1,input-file-spec2[,...] -
/qualifiers output-file-spec/qualifiers

Command Qualifiers

/KEY
/[NO]STABLE
/[NO]DUPLICATES
/[NO]CHECK_SEQUENCE
/PROCESS
/COLLATING_SEQUENCE
/[NO]STATISTICS
/WORK_FILES
/SPECIFICATION

Input File Qualifiers

/FORMAT
/INDEXED_SEQUENTIAL
/[NO]SHAREABLE
/TREE_SPACE

Output File Qualifiers

/FORMAT
/SEQUENTIAL
/RELATIVE
/INDEXED_SEQUENTIAL
/[NO]OVERLAY
/ALLOCATION
/[NO]CONTIGUOUS
/LOAD_FILL
/BUCKET_SIZE

Prompts

File: input-file-spec1,input-file-spec2[,...]/qualifiers
Output: output-file-spec/qualifiers

Command Parameters

input-file-spec1,input-file-spec2[,...]

Specifies the names of the sorted files whose records are to be merged. At least two files, but not more than ten, must be specified and separated by commas. The keys must be the same in all files.

If the file specifications do not include a file type, MERGE assumes the default file type of .DAT.

output-file-spec

Specifies the name of the merged file to be created. Its qualifiers can request characteristics for the merged output file. The command string may have only one output file specification.

If the output file specification does not include a file type, MERGE assigns the default file type of .DAT to the output file, regardless of the file types of the input files.

Command Qualifiers

/KEY=(field[,...])

Defines a MERGE key. This qualifier can be specified up to ten times to define ten different key fields to be merged.

The /KEY qualifier consists of two required arguments that define the position and size of the key field within the record, and several optional arguments that define the type of data within the key field. The arguments must be separated with commas and enclosed in parentheses.

Required Arguments for the /KEY Qualifier

POSITION:n

Specifies the position of the key within each record, where the first character of the record is position 1.

SIZE:n

Specifies the length of the MERGE key in characters, bytes, or digits, depending on the key field data type. The total of all key field sizes must be less than 255 bytes. The valid sizes, based on data type, are:

Data Type	Values for n
CHARACTER	1–255
BINARY	1, 2, 4, 8
DECIMAL	1–31

Note

You cannot use the SIZE argument with floating point data types (D_FLOATING, F_FLOATING, or ASCII_FLOATING).

Optional Arguments for the /KEY Qualifier

CHARACTER
BINARY
DECIMAL
ASCII_FLOATING
ASCII_ZONED
DIBOL_ZONED
D_FLOATING
F_FLOATING
PACKED_DECIMAL

Indicates the type of data in the MERGE key field. If not specified, MERGE assumes that data is CHARACTER.

SIGNED
UNSIGNED

Indicates whether a binary or decimal key is to be compared as a signed or an unsigned integer. UNSIGNED causes all numbers to be treated as positive. SIGNED is the default.

LEADING_SIGN
TRAILING_SIGN

Indicates whether the sign of a decimal data type key appears at the beginning or end of the key. If you specify the key data type as DECIMAL, but do not specify either of these subqualifiers, the default is TRAILING_SIGN.

OVERPUNCHED_SIGN SEPARATE_SIGN	Indicates whether the sign of a decimal data type key is superimposed on the decimal value or is separated from the decimal value. If you specify the key data type as DECIMAL, but do not specify either of these subqualifiers, the default is OVERPUNCHED_SIGN.
ASCENDING DESCENDING	Indicates whether the key is to be merged into ascending or descending order. If you specify neither of these subqualifiers, the default is ASCENDING.

/CHECK_SEQUENCE
/NOCHECK_SEQUENCE

Examines the input files to be merged to make sure that they are in order. If a record is out of order, MERGE displays the following warning message at the end of the merge operation:

```
%SOR-E-BAD_ORDER, Input file [n] is out of order
```

The default is /CHECK_SEQUENCE.

/COLLATING_SEQUENCE=sequence

Specifies the collating sequence in which records are to be arranged. The sequence may be ASCII, EBCDIC, or MULTINATIONAL. The default is /COLLATING_SEQUENCE=ASCII.

/STABLE
/NOSTABLE

Specifies how input files containing records with equal keys are to be merged in the output file. The default is /NOSTABLE, which causes any records with equal keys to be grouped together in the output file, possibly resulting in an unpredictable merge order for those particular records. The /STABLE qualifier specifies that any records with equal keys are to be directed to the output file in the order in which they were input to MERGE. See the following /[NO]DUPLICATES qualifier discussion.

/DUPLICATES
/NODUPLICATES

Alternatively specifies how input files containing records with equal keys are to be merged in the output file (see the previous /[NO]STABLE qualifier discussion). The /NODUPLICATES qualifier specifies that only one record be retained when MERGE encounters records with equal keys. The default is /DUPLICATES.

Note

Do not use /NODUPLICATES and /STABLE in the same merge operation.

/PROCESS = type

Defines the type of merge. You can specify one of the following merge types:

RECORD	Requests MERGE to resequence the entire contents of the input files and create an output file containing the reordered records.
TAG	Requests MERGE to merge only the record keys, and then reaccess the input files to create an output file containing the resequenced records.
ADDRESS	Requests MERGE to produce an address file merged by record keys. The output file can be read as an index to read the original file in the desired sequence.
INDEX	Requests MERGE to produce an address file containing the key field of each data record and a pointer to its location in the input file. The output file can be read to randomly access the data in the original file in the desired sequence.

The default is /PROCESS = RECORD.

/STATISTICS
/NOSTATISTICS

Specifies whether a statistical summary displays after the MERGE operation completes. The default is /NOSTATISTICS.

/WORK_FILES = n

Specifies the number of temporary work files to be used during the merge process. You can specify 0 or any value from 3 through 10. A value of 0 indicates that no work files are necessary because the data will fit in physical memory.

/SPECIFICATION[= file-spec]

Indicates that commands and file qualifiers, including key field definitions, are contained in a specification file. See the *PDP-11 SORT/MERGE User's Guide* for complete details on using specification files.

Input File Qualifiers

/FORMAT=(file-attribute[,...])

Specifies attributes of the input file to override the existing data that MERGE normally obtains through PDP-11 RMS. You can specify one or both of the following arguments:

FILE_SIZE:n	Defines the size of the file in blocks. This argument is required for files that are not on disk or magnetic tape. If no input file size is specified, MERGE uses the default value of 1000 blocks.
RECORD_SIZE:n	Specifies, in bytes, the length of the longest record, overriding the record size defined in the file header or label. This argument is required for files that are not on disk or magnetic tape, or if the longest record size (LRL) is unavailable or known to be inaccurate.

/INDEXED_SEQUENTIAL[=n]

Specifies that the organization of the input file is indexed-sequential. You specify the number of keys (1 through 255) to make sure that MERGE allocates sufficient RMS space. The default value is 1.

/SHAREABLE
/NOSHAREABLE

Specifies whether the input files are opened in write-shareable mode. Use this qualifier when you want to merge files that may be updated by another user during the merge operation. The default is /NOSHAREABLE.

TREE_SPACE=n

Specifies the percentage distribution of available work area between SORT/MERGE tree-related data structures and I/O-related data structures. You can specify from 1 to 100 percent (%). The default division of work area is 30% to the merge list and 70% to I/O.

Output File Qualifiers

/FORMAT=(record-format[,...])

Defines the output file record format. You can specify one or more of the following arguments:

FIXED[:n] VARIABLE[:n] RMS_STREAM[:n] STREAM[:n] CONTROLLED[:n]	Defines the output file record format and size, where n is the length of the longest record in the output file. If n is not specified, the default is a length long enough to hold the longest output record.

BLOCK_SIZE=n
: Specifies, when the output file is directed to disk or magtape, the block size in bytes. If one or more input files is a tape file, the output file block size defaults to the maximum of the block sizes of all tape input files. If the input file is a disk file, the default value is 512 bytes.

/SEQUENTIAL
: Specifies that the organization of the output file is sequential. This is the default for an ADDRESS or INDEX sort process; for a RECORD or TAG sort process, the output file organization defaults to the organization of the input file.

/RELATIVE
: Specifies that the organization of the output file is relative. By default, a RECORD or TAG sort process results in an output file that has the same organization as the input file. Use the /RELATIVE qualifier to create a relative output file from a sequential or indexed-sequential input file.

/INDEXED_SEQUENTIAL[=n]
: Specifies that the organization of the output file is indexed-sequential. You specify the number of keys (1 through 255) to make sure that MERGE allocates sufficient RMS space. The default value is 1.

/OVERLAY
/NOOVERLAY
: Specifies whether an existing file is to be overlaid with the sorted records of the input file. The default is /NOOVERLAY, which means a new output file is created and does not overlay an existing file.

/ALLOCATION=n
: Specifies the number of 512-byte blocks to be allocated for the output file. By default, MERGE allocates blocks based on the number of records sorted.

/CONTIGUOUS
/NOCONTIGUOUS
: Specifies whether the allocation of disk space for the output file is to be contiguous. If you specify /CONTIGUOUS, you must also specify /ALLOCATION to define the number of blocks to allocate for the output file. The default is /NOCONTIGUOUS.

/LOAD_FILL
: Used only with indexed files, this qualifier loads the buckets according to the fill size established when the file was created.

/BUCKET_SIZE
: Used with disk files, this qualifier specifies the number of 512-byte blocks per bucket for the output file. The maximum size you can specify is 32 blocks. If you do not specify a bucket size, the bucket size of the output file defaults to that of the input file.

Comparing Files: DIFFERENCES

The DIFFERENCES command compares two text files and lists any sections of text that differ between the two files.

Format

DIFFERENCES input-file-spec compare-file-spec

Command Qualifiers	**Defaults**
/IGNORE=BLANKLINES	
/MATCH=size	/MATCH=3
/MAXIMUM_DIFFERENCES=n	/MAXIMUM_DIFFERENCES=300
/OUTPUT=file-spec	/OUTPUT=KB:

Prompts

File 1: input-file-spec

File 2: compare-file-spec

DIFFERENCES compares the two files by groups of lines and produces an output file if the /OUTPUT qualifier is used; otherwise, output is sent to the terminal. This output file lists the differences, if any.

The qualifiers for the DIFFERENCES command can be categorized according to their functions:

- The /IGNORE=BLANKLINES qualifier requests DIFFERENCES to ignore blank lines while comparing files.

  By default, DIFFERENCES compares every line in each file.

- The /MATCH and /MAXIMUM_DIFFERENCES qualifiers control the extent of the comparison.

  By default, DIFFERENCES reads every line in the first input file and looks for a matching line in the second input file. Lines are considered matched only if three identical sequential lines are found in each file.

The DIFFERENCES command normally displays its report at your terminal. Use the /OUTPUT qualifier to request DIFFERENCES to write the output to an alternate file or device. If the two files are significantly different, then the system may print one of the following error messages:

```
?Maximum memory exceeded

?Maximum differences (300) encountered - skipping rest of file
```

For example, you may want to use DIFFERENCES to compare a file (such as HOME.WRK) with its earlier backup version (HOME.BAK). The command line you enter is:

```
$ DIFFERENCES HOME.WRK HOME.BAK (RET)
```

RSTS/E displays a message telling you which files it is comparing:

```
Comparing: 1) SY:[2,214]HOME.WRK to 2) SY:[2,214]HOME.BAK
```

Because you did not include a directory in the command line, the default is your directory.

Next, RSTS/E displays the lines containing text that does not match. The sections in the following example are labeled 1) and 2), corresponding to the first and second file being compared. The sections of text are separated by rows of asterisks (*):

```
******************************
1) SY:[2,214]HOME.WRK
the writings of Milton and
Donne.  Fortunately, I find
their work interesting.
**********
2) SY:[2,214]HOME.BAK
the writings of Marchand and
Hogan.  Fortunately, I find
their work interesting.

?1 Difference Found
```

Notice that the numbers 1) and 2) correspond to HOME.WRK and its backup version, HOME.BAK. DIFFERENCES lists the differences in the two files one group of lines at a time.

Command Parameters

input-file-spec
: Specifies the name of the first input file to be compared.

  You must include a file name. The default file type is null. You cannot use wildcard characters in the file specification.

compare-file-spec
: Specifies the name of the second input file to be compared. You must include a file name. The default file type is null.

Command Qualifiers

/IGNORE=BLANKLINES
: Specifies that blank lines between data lines are to be ignored during the comparison. By default, the DIFFERENCES command compares every line in each file and reports all differences.

For example, if files HOGAN.TOM and HOGAN.SAM have similar contents but have unequal numbers of blank lines, you can disregard the blank lines in their comparison by typing:

```
$ DIFFERENCES HOGAN.TOM HOGAN.SAM/IGNORE=BLANKLINES RET
```

/MATCH=size

Controls the number of lines that must be identical for DIFFERENCES to consider them a match. For example, to make sure that five identical lines are found to be considered a match between MARYB.TXT and BILL.TXT, type:

```
$ DIFFERENCES MARYB.TXT BILL.TXT/MATCH=5 RET
```

By default, after DIFFERENCES finds unmatched lines, it assumes that the files match after it finds 3 sequential lines that match. The match size you specify overrides the default value of 3.

/MAXIMUM_DIFFERENCES=n

If you specify /MAXIMUM_DIFFERENCES, DIFFERENCES terminates after locating n differences. The output file lists differences only on lines compared until the maximum has been reached.

For example, you may need to compare two large data files (MIKE.DAT and EDDIE.DAT), but if more than four differences are found, any further comparison would not be worth your while. In this case you type:

```
$ DIFFERENCES MIKE.DAT EDDIE.DAT/MAXIMUM_DIFFERENCES=4 RET
```

By default, DIFFERENCES aborts after 300 differences are found.

/OUTPUT=file-spec

Stores the result of the comparison in a file rather than displaying it at your terminal. If you omit the /OUTPUT qualifier, the output displays at your terminal.

The /OUTPUT qualifier is especially useful if you are using a video terminal and want a listing of the output. For example, you can request that the output from comparing files RACE.A and RACE.B be stored in a file that you name RESULT.C:

```
$ DIFFERENCES RACE.A RACE.B/OUTPUT=RESULT.C RET
```

If you specify /OUTPUT without a file specification, the output goes to a file with the same file name as the first input file and a file type of .DIF. If you had not assigned the file name RESULT.C in the previous example, it would have been named RACE.DIF.

You cannot use wildcard characters in the file specification.

Allowing Access to Files

This section describes what it means to assign protection codes to your files. Then it explains how to use the SET PROTECTION command and the /DEFAULT qualifier.

Using Protection Codes

A protection code determines who has access to a file.

The protection code consists of one to three decimal digits. This code determines the file's degree of protection on two levels:

- The actions (reading, writing, executing, and deleting) against which it is protected. Anyone who can write in a file can also delete it, and vice versa.
- The user or class of users against whom it is protected.

RSTS/E recognizes three classes of users for protection purposes:

- Owner — The owner of the file, known to the system by a PPN [52,20]).
- Group — The owner's group, known to the system by any PPN whose project number (the 52 in [52,20]) is the same as the owner's project number. Group read or write access also allows the owner read or write access.
- World — All other users on the system, known to the system by any PPN whose project number is different from the owner's. World read or write access also allows both the owner and the owner's group read or write access.

For example, 60 is the default protection code that RSTS/E normally assigns. (The code is something other than 60 if your system manager changed the default, or if you use the SET PROTECTION/DEFAULT command described in the next section.) If one of your files has a protection code of 60, only you or a user with sufficient privileges can read, edit, or delete it. A protection code remains fixed unless you change it.

Note

Some users on your system (for example, the system manager) may have privileges that allow them to access files regardless of the protection codes.

When you create a file, RSTS/E assigns it the default protection code 60:

```
$ CREATE ANNIE.DAT (RET)
Heres a file of only one line.
<CTRL/Z>

$ DIR ANNIE.DAT (RET)

 Name .Typ    Size     Prot      Name .Typ    Size     Prot    SY:[52,20]
ANNIE .DAT       1    < 60>

Total of 1 block in 1 file in SY:[52,20]
```

You assign a protection code with the SET PROTECTION command, as in:

```
$ SET PROTECTION=40 ANNIE.DAT (RET)
ANNIE    .DAT renamed to ANNIE    .DAT<40>

$ DIR ANNIE.DAT (RET)

 Name .Typ    Size     Prot      Name .Typ    Size     Prot    SY:[52,20]
ANNIE .DAT       1    < 40>

Total of 1 block in 1 file in SY:[52,20]
```

A factor in assigning protection codes is whether a file is executable. An executable file is produced by the LINK command or by the COMPILE command of BASIC-PLUS or BASIC-PLUS-2. An executable file contains the protection code 64. See Table 3-3 for executable file protection codes.

A nonexecutable file, on the other hand, is any file that does not contain a protection code of 64.

Table 3-2 lists the file protection codes for nonexecutable files.

Table 3-2: File Protection Codes for Nonexecutable Files

Code	Meaning
1	Protection against reading by owner
2	Protection against writing by owner
4	Protection against reading by owner's group
8	Protection against writing by owner's group
16	Protection against reading by anyone not in the owner's group
32	Protection against writing by anyone not in the owner's group
128	A protected data file; overwritten with zeros when deleted

Individual codes added to the executable protection code 64 have meanings different from those in Table 3-2.

Table 3-3 lists the compiled codes for executable files (64 + code).

Table 3-3: File Protection Codes for Executable Files

64+ Code	Meaning
1	Execute protection against the owner
2	Read and write protection against the owner
4	Execute protection against the owner's group
8	Read and write protection against the owner's group
16	Execute protection against all others not in the owner's group
32	Read and write protection against all others not in the owner's group
128	Program with privileges

The protection code you set for a file is the total of the code numbers for the types of protection you want. For example, the usual system default (60) protects against reading, writing, and deleting by all users except its owner, because it is a sum of 4, 8, 16, and 32.

Table 3-4 lists common codes and their meanings.

Table 3-4: Common File Protection Codes

Code	Sum of Codes	Meaning
232	128+64+32+8	Executable file that is privileged and protected against reading and writing by anyone except owner. However, anyone can execute the file.
124	64+32+16+8+4	Executable file that is protected against reading, writing, and execution by anyone except owner.
104	64+32+8	Executable file that is protected against reading and writing by anyone except owner. However, anyone can execute the file.
62	32+16+8+4+2	Protected against reading by anyone except owner. Protected against accidental deletion or modification by owner or anyone else.
60	32+16+8+4	Protection against reading and writing by anyone except owner.
48	32+16	Protection against reading and writing by anyone not in owner's group. Can be read or written by anyone in owner's group.
42	32+8+2	Protection against writing by anyone including owner; readable by everyone.
40	32+8	Protection against writing by anyone except owner; readable by everyone.
0		No protection (any user can read, write, and delete).

SET PROTECTION

The SET PROTECTION command specifies the protection code of a file. You assign a protection code to determine who else, if anyone, can have access to your files. (To check the protection codes of your files, use the /PROTECTION qualifier with the DIRECTORY command.)

A file specification and protection code are required unless you use the /DEFAULT qualifier. You must include either an equal sign or a space between the command SET PROTECTION and the protection code you specify.

Format

SET PROTECTION[=]n filespec[,...]

Command Qualifiers	**Defaults**
/DEFAULT	
/[NO]QUERY	/NOQUERY
/[NO]LOG	/LOG

For example, you set the protection code of a file named TIME.TRY to 42 by typing:

```
$ SET PROTECTION=42 TIME.TRY (RET)
```

Command Parameters

n

The value you specify for n can be any of the protection codes in Table 3-2, Table 3-3, and Table 3-4, depending on the privileges you have been given by the system manager.

filespec

You can include one or more file specifications in the command string. Wildcard characters are allowed.

Command Qualifiers

/DEFAULT

The /DEFAULT qualifier specifies the default protection code of each file you create. (You can override the default for individual files.) /DEFAULT stays in effect until you log out.

If you use SET PROTECTION/DEFAULT, RSTS/E assigns the protection code you specify to all files you create during the current session. Do not include a file specification when you use the /DEFAULT qualifier. Also, you must include either an equal sign or a space between the command and the protection code:

```
SET PROTECTION[=]n/DEFAULT
```

/QUERY
/NOQUERY

The /QUERY qualifier prompts you to state whether you want to change a file's protection code. The default is /NOQUERY.

The /QUERY qualifier is useful only when you use wildcards in the file specification. You can enter one of the following responses to the question mark prompt:

Y	Yes, change the protection code
N	No, do not change the protection code
RET	No, do not change the protection code
CTRL/Z	Return to command level

For example, if you have several files named MOODY, you can determine their protection codes as follows:

```
$ SET PROTECTION=40/QUERY MOODY.* RET
[52,20]MOODY .DAT? Y RET
MOODY.DAT renamed to MOODY.DAT < 40>
[52,20]MOODY .LIN? N RET
[52,20]MOODY .JON? CTRL/Z
```

Do not use either the /QUERY or /NOQUERY qualifiers with /DEFAULT.

/LOG
/NOLOG

Determines whether RSTS/E displays a message to confirm that a file's protection code was changed. The default is /LOG. To suppress the message when changing the protection code of a file (such as STRIPE.LIS), type:

```
$     PROTECTION=64 STRIPE.LIS/NOLOG RET
```

Unless you specify /NOLOG, a message displays:

```
$ SET PROTECTION=64 STRIPE.LIS/LOG RET
STRIPE.LIS renamed to STRIPE.LIS < 64>
```

Do not use /LOG or /NOLOG with /DEFAULT.

Setting File Characteristics: SET FILE

In addition to the SET PROTECTION command described in the previous section, you can use the SET FILE command to specify protection code, as well as other characteristics of a file.

Format

SET FILE file-spec

Qualifiers	**Defaults**
/NOCONTIGUOUS	
/[NO]DELETABLE	/DELETABLE
/[NO]LOG	See discussion
/[NO]PLACED	
/PROTECTION=n	
/RUNTIME_SYSTEM=name	

Command Parameters

file-spec
Specifies the file whose characteristics you want to set or change.

Command Qualifiers

/NOCONTIGUOUS
Lets you extend a contiguous file by changing it to noncontiguous.

/DELETABLE
/NODELETABLE
Specifies whether a file can be deleted or renamed. The default is /DELETABLE.

/LOG
/NOLOG
Specifies whether the file specification of each modified file is displayed at the terminal. If the file specification contains any wildcards, the default is /LOG. Otherwise, the default is /NOLOG.

/PLACED
/NOPLACED
Specifies whether a file, after modification, is placed at its present position on the disk.

/PROTECTION=n
Specifies the protection code of a file. The protection code must be in the range of 0 to 255. You can also use the SET PROTECTION command, described in the previous section, to specify a file's protection code.

/RUNTIME_SYSTEM=name
Specifies the run-time system associated with the file.

System and Account Operations 4

This chapter describes the commands that let you perform common system and account operations, such as:

- Displaying system status
- Displaying account information
- Changing your password
- Sending a message to the system operator's terminal

SHOW

Use the SHOW command to display information about the system and its resources, as well as the current status of your job or account.

To specify the type of information you want the SHOW command to display, use one of the options listed in Table 4-1.

Note

The SHOW command has additional capabilities for users with greater levels of privilege. See the *RSTS/E System Manager's Guide* for more information.

Format

SHOW option

Table 4-1: SHOW Command Options

Option	Displays
ACCOUNT	Information about accounts. See this chapter for more information.
BUFFERS	Information about available buffers, in-use and maximum job count, hung terminal count, and total logged errors.
CACHE	Current cache settings.
DATE DAYTIME TIME	Current date and time.
DEVICE [dev[:]] DEVICE/ALL	Status of devices in use on the system, or a device you specify. See Chapter 6 for more information.
DEVICE/ALLOCATED	All allocated or open devices, excluding terminals and disks. See Chapter 6 for more information.
DISKS	Status of all mounted disks. See Chapter 6 for more information.
ENTRY [entry] ENTRY/ALL	Print or batch entries that have been queued. See Chapter 7 for more information.
FILE [file-spec] FILE/ALL	Information about installed files on the system. See the *RSTS/E System Manager's Guide* for more information.
JOB [job-number] JOB/ALL	Status of jobs on the system. See this chapter for more information.

(continued on next page)

Table 4-1: SHOW Command Options (Cont.)

Option	Displays
JOB/PRIVILEGES	Privileges assigned to your account.
LIBRARIES	All installed resident libraries and dynamic regions.
LOGICAL [name] LOGICAL/ALL	Information about user-defined logical names.
LOGICAL/SYSTEM [name] LOGICAL/SYSTEM/ALL	Information about system-wide logical names.
MEMORY	Status of system memory allocation.
NETWORK	Names of the nodes that your system can access.
PRINTER [printer-name[:]] PRINTER/ALL	Characteristics of all printers on the system, or the printer you specify.
QUEUE [queue-name[:]] QUEUE/ALL	Characteristics of a print or batch queue. See Chapter 7 for more information.
RECEIVERS	Status of all declared message receivers.
RUNTIME_SYSTEMS	Status of all installed run-time systems.
SERVER [server-name[:]] SERVER/ALL	Characteristics of one or more servers in the Print/Batch Services (PBS) facility. See the *RSTS/E System Manager's Guide* for more information.
SYMBOL [symbol-name] SYMBOL/ALL	Value of a specified symbol or all symbols in a local symbol table or the global symbol table. See Chapter 6 for more information.
SYSTEM	System default characteristics. See this chapter for more information.
TERMINAL	Characteristics of your terminal. See Chapter 5 for more information.
USER [ppn] USER/ALL	Information about attached jobs, or all jobs, on the system. See this chapter for more information.

System and Account Status

The SHOW USER command displays information about users who are logged in and what system resources they are using. The SHOW JOB command displays the status of your current job. The SHOW SYSTEM command displays the default characteristics of the system. The SHOW ACCOUNT command displays data about an account.

The following is an example of the SHOW USER command, along with an explanation of each numbered element in the display:

```
$ SHOW USER (RET)

①  RSTS V9.0  status at 09-Jun-85, 05:50 PM Up: 6:22:19

②     ③      ④      ⑤      ⑥      ⑦       ⑧        ⑨
Job    Who    Where   What    Size    State   Run-Time  RTS
 3     1,226  KB27    NET     17/64K  SL          3.4   ...RSX
 5     1,214  KB26    SYSTAT  20/64K  RN Lck      3.2   ...RSX
11     1,196  KB30*   DCL      4/64K  ^C        9:39.4  DCL
14     1,231  KB45    BACKUP  15/64K  SL        8:13.6  ...RSX
```

The display tells you the following:

① System status information —You are using RSTS/E Version 9.0 software. At the present day and time (09-Jun-85, 05:50 PM) the system has been operating for 6 hours, 22 minutes, and 19 seconds (Up: 6:22:19).

② Job — There are five attached jobs now in use on the system.

③ Who — This column displays the accounts to which the jobs belong.

④ Where — This column displays the keyboards controlled by the jobs. The asterisk (*) following KB30 indicates a dial-up line.

⑤ What — Various operations (SYSTAT, BACKUP) are being performed.

⑥ Size — These numbers indicate both the actual size of your job, as well as the size to which the job can expand. The size is shown in thousands of words — or K words — of computer storage space. A word is a unit of storage in the computer that holds 16 bits, or 2 ASCII characters, of information. In computing, K approximates thousand-word units of memory storage (actually 1K = 1024 words). The Size column is mainly to tell you the load each job places on the system.

⑦ State — The status of each job is determined by the operation being performed. For example, the SYSTAT job's status is RN Lck. This means that the job is running or waiting to run (RN), and that it is locked in memory (Lck) during the SYSTAT operation. The SYSTAT operation is completed when the display appears at your terminal and is followed by the DCL prompt, which shows that your job is no longer locked in memory.

⑧ Run-Time — The system keeps track of the CPU time each job uses. For example, the BACKUP job has used a total of 8 minutes and 13 seconds of CPU time.

⑨ RTS (Run-Time System) — RSTS/E provides a variety of run-time systems, including DCL and null RTS (shown on the display as "...RSX").

Table 4-2 shows the elements you may find in a system status display. You might want to refer to this table when you use the SHOW USER or SHOW JOB commands.

Table 4-2: SHOW USER and SHOW JOB Abbreviations

Abbreviation	Meaning	Description
Where Column		
DET	Detached	Job is detached from all terminals.
*	Dial-up	Job is running over a dial-up line.
Who Column		
,	–	Job is not logged into the system.
State Column (Job Status)		
RN	Run	Job is running or waiting to run.
RS	Residency	Job is waiting for residency. (The job has been swapped out of memory and is waiting to be swapped back in.)
BF	Buffers	Job is waiting for buffers (no space is available for I/O buffers).
SL	Sleep	Job is sleeping (SLEEP statement).
SR	Send/Receive	Job is sleeping and is a message receiver.
FP	File Processor	Job is waiting for file processing by the system (opening or closing a file, file search).
TT	Terminal	Job is waiting to perform output to a terminal.
HB	Hibernating	Job is detached and waiting to perform I/O to or from a terminal. (Someone must attach to the job before it can resume execution.)
KB	Keyboard	Job is waiting for input from a terminal.
^C	CTRL/C	Job is at command level, awaiting a command. (In other words, the keyboard monitor has displayed its prompt and is waiting for input.)
CR	Card Reader	Job is waiting for input from a card reader.
MT,MM, or MS	Magnetic Tape	Job is waiting for magnetic tape I/O.
LP	Line Printer	Job is waiting to perform line printer output.
DT	DECtape	Job is waiting for DECtape I/O.
DK,DM,DB, DP,DL,DR	Disk	Job is waiting to perform disk I/O.
DX	Flexible Diskette	Job is waiting to perform flexible diskette I/O.

(continued on next page)

Table 4-2: SHOW USER and SHOW JOB Abbreviations (Cont.)

Abbreviation	Meaning	Description
RJ	RJ2780	Job is waiting for RJ2780 I/O. (The RJ2780 is a device for communicating with an IBM system.)
??	–	Job's state cannot be determined by the SHOW command.
+ or –	Temporary Privileges	Job has temporary privilege. A plus sign (+) means that the temporary privilege is active. The minus sign (–) means that the program has temporary privilege, but that it is not active.
The following status descriptions may appear after one or more of the other job state abbreviations:		
Lck	Locked	Job is locked in memory for the current operation.
Nsw	No Swapping	A program has requested that the job not be swapped from memory.
Swi	Swapping In	Job is currently being swapped into memory.
Swo	Swapping Out	Job is currently being swapped out of memory.
Xnn	–	Job is swapped out and occupies slot nn in swap file X; file is denoted by A, B, C, or D to represent files 0 through 3 of the swapping structure. For example, A03 means slot 3 of swapping file 0.
RTS Column (Run-time System)		
Note Any run-time system available on your system may appear in this column. Some of the most commonly used run-time systems are listed here.		
BASIC	BASIC	The BASIC-PLUS run-time system.
BASIC2	BASIC-PLUS-2	The BASIC-PLUS-2 run-time system.
DCL	DCL	The DIGITAL Command Language (DCL) keyboard monitor.
RT11	RT11	The RT11 run-time system.
RSX	RSX	The RSX run-time system.
...RSX	–	"Null RTS," meaning the monitor is executing your job directly without using a run-time system.

Attached and Detached Jobs

Some of the command displays of system status involve attached and detached jobs. An attached job is one that is associated with a terminal, and thus allows user input. By contrast, a detached job on RSTS/E is used to perform various system functions, such as printing files on the line printer.

For example, suppose the system manager invokes a program at system start-up time to keep track of jobs that run through any queue on the system. RSTS/E considers that job to be detached, because it is not associated with a specific terminal.

You may not recognize the names of programs that are run by detached jobs. They need not be standard RSTS/E programs — some could be created at your site. In addition, some of these programs are used primarily by the system manager. Programs for system management are described in the *RSTS/E System Manager's Guide*.

The following is an example of a display produced by the SHOW USER/ALL or SHOW JOB/ALL commands. Note that both attached and detached jobs are displayed:

```
RSTS V9.0  status at 09-Jun-85, 05:50 PM Up: 6:22:34

Job     Who     Where     What     Size     State     Run-Time   Pri/RB   RTS
 1      1,2     Det       ERRCPY   11/64K   SR A40       10.2     0/6    ...RSX
 2      1,2     Det       OPSRUN   22/64K   SL         2:44.7    -8/6    ...RSX
 4      1,196   KB30      DCL       4/64K   ^C         9:39.4    -8/6    DCL
 7      1,2     Det       QUMRUN   24/64K   SL         1:04.5     0/6    ...RSX
12      1,217   P1J17     DCL      14/64K   RN         8:52.8   -16/6    DCL
15      1,214   KB26      SYSTAT   20/64K   RN Lck        5.2    -8/6    ...RSX
17      1,2     Det       PBS...   18/64K   SL         6:00.0    -8/6    ...RSX
```

SHOW USER

The SHOW USER command displays information about the status of jobs on the system.

Format

SHOW USER [[ppn]]

Command Qualifiers	Defaults
/ALL	
/ATTACHED	/ATTACHED
/DETACHED	
/OUTPUT = filespec	
/TERMINAL = KBn:	

Command Parameters

[ppn]
Specifies the account whose jobs are to be displayed. If you do not specify an account, the SHOW USER command displays all attached jobs on the system.

Command Qualifiers

/ALL
Specifies that all jobs, both attached and detached, be displayed.

/ATTACHED
Specifies that only attached jobs be displayed.

/DETACHED
Specifies that only detached jobs be displayed.

/OUTPUT = filespec
Specifies whether the display is directed to a file.

/TERMINAL = KBn:
Specifies that only the job logged in at a particular terminal be displayed.

SHOW JOB

The SHOW JOB command, like SHOW USER, also displays information about the status of jobs on the system. This command is more convenient, however, for quickly showing the status of only your own job.

Format

SHOW JOB [job-number]

Command Qualifiers

/ALL
/ATTACHED
/DETACHED
/OUTPUT = filespec
/TERMINAL = KBn:

Command Parameters

job-number
Specifies the the job number to be displayed. If you do not specify a job number, the SHOW JOB command displays only your own job. For example:

```
$ SHOW JOB (RET)

 5      1,214   KB26   SYSTAT   20/64K   RN LcK      4,2   -8/6   ...RSX
```

Command Qualifiers

/ALL
Specifies that all jobs, both attached and detached, be displayed.

/ATTACHED
Specifies that only attached jobs be displayed.

/DETACHED
Specifies that only detached jobs be displayed.

/OUTPUT = filespec
Specifies whether the display is directed to a file.

/TERMINAL = KBn:
Specifies that only the job logged in at a particular terminal be displayed.

CTRL/T

In addition to using the SHOW JOB command, you can get a quick one-line status report on your current job by pressing CTRL/T. When you press CTRL/T, RSTS/E displays a report similar to the following:

```
18   GARP::KB32   SYSTAT+BAS4F   ^C(0R) 11(16)K+ 16K 3.3(+.5)
```

The report shows:

- Your current job number (for example, 18)
- The node name of your system (for example, GARP)
- The keyboard number of your terminal (for example, KB32)
- The name of the program or operation you are currently running (for example, SYSTAT)
- The current run-time system name (for example, BAS4F)
- The current job state (for example, ^C(0R))
- The current program size in words (for example, 11)
- The maximum program size in words (for example, (16)k)
- The run-time system size in words (for example, +16K)
- The amount of CPU time in seconds your job has used (for example, 3.3)
- The amount of time the job has run since the last CTRL/T (for example, (+.5))

Note

CTRL/T is an optional feature, so it may not be available on your system.

SHOW SYSTEM

The SHOW SYSTEM command displays the default characteristics of the system:

Format

SHOW SYSTEM

```
$ SHOW SYSTEM (RET)

System name:                    RSTS V9.0 GARP::
Date format:                    Alphabetic
Time format:                    AM_PM
Magtape label default:          DOS
Magtape density default:        800 BPI
Power fail restart delay:       13 seconds

Job limit:                      40
Current jobs:                   25

Password Prompting:             Network and Dialup users
```

SHOW ACCOUNT

The SHOW ACCOUNT command displays data about an account.

Format

SHOW ACCOUNT [dev:[ppn]]

Command Qualifiers	Defaults
/ACCOUNTING_DATA	
/BRIEF	/BRIEF
/FULL	
/OUTPUT=filespec	

Command Parameters

dev:[ppn]
Specifies the disk and account to be displayed. If you do not specify the disk, the default is SY0:; if you do not specify account, the default is your own PPN.

Command Qualifiers

/ACCOUNTING_DATA
/BRIEF
/FULL
Specifies the amount or type of information to be displayed about the account.

If you specify /ACCOUNTING_DATA, RSTS/E displays accounting information only. For example:

```
$ SHOW ACCOUNT/ACCOUNTING_DATA RET

Account    Name          Allocation          KCT      CPU-Time  Connect   Device

[1,214]                       4692       1041124   01:40:27.3    91:21    02:48
```

If you specify /BRIEF (the default), RSTS/E displays brief general information about your account. For example:

```
$ SHOW ACCOUNT RET

Account    Name          Allocation   IDNCLP Last Login                 Expires

[1,214]                       4692   IDN  P KB2:   19-Jun-85 10:48 AM
```

If you specify /FULL, RSTS/E displays complete information about your account. For example:

```
$ SHOW ACCOUNT/FULL (RET)

_SY0:[1,214]  (no name)                          Created: 26-Apr-85

   Privileges:  SETPAS

   Attributes:  INTERACTIVE    DIALUP      NETWORK     NOCAPTIVE
                NOLOOKUP       PASSWORD_PROMPT         NOEXPIRE

   Quotas:  Disk usage    - Logged out:       10000   Logged in:     unlimited
            Job limits    -   Detached:   unlimited      Total:     unlimited
            Send/Receive -         RIB:   unlimited    Message:             0

   Accounting:       CPU Time:    01:13:25.0    Kilo-Core Ticks:       612586
                  Device Time:    00:16         UFD Clustersize:           16
                 Connect Time:   105:07        Blocks allocated:         4692

                 Last Password change on 01-Jun-85 at 03:44 PM
                 Last login on _KB2: on 19-Jun-85 at 10:48 AM
```

/OUTPUT=filespec

Specifies whether the account display is directed to a file.

SET

Use SET and an option to define or change certain default characteristics, such as the default protection codes the system assigns to your files.

Format

SET option

Table 4-3 lists the options that you can use with the SET command.

Table 4-3: SET Command Options

Option	Function
ENTRY	Modifies an entry on a print or batch queue. See Chapter 7 for more information.
FILE	Specifies the characteristics of a file. See Chapter 3 for more information.
HOST	Specifies the network node to connect your terminal to, so you can log in to that node. See Chapter 1 for more information.
LOG_FILE	Selectively enables or disables output to a log file that you previously opened with the OPEN/LOG_FILE command. See Chapter 5 for more information.
PASSWORD	Lets you change your password. See following sections in this chapter for more information.
PROTECTION	Determines the protection code of a file. See Chapter 3 for more information.
TERMINAL	Specifies the characteristics of a terminal. See Chapter 5 for more information.

Note

The following SET options are used in command procedures. See the *RSTS/E Guide to Writing Command Procedures* for more information on how to use these options

[NO]CONTROL=C	Controls CTRL/C checking within a command procedure.
[NO]DATA	Controls whether DCL reads data required by a program or a command from your terminal, rather than from a command file.
[NO]ECHO	Controls whether terminal output is enabled or disabled during command procedure execution.
[NO]ON	Controls error checking within a command procedure.
[NO]VERIFY	Controls whether DCL command lines are displayed during command procedure execution.

SET PASSWORD

The SET PASSWORD command lets you change your own password at any time to provide a greater level of protection against unauthorized users of the system. To maintain secrecy, you should change your password frequently.

Usually, all users are granted the privilege to do this; however, in some cases the system manager may restrict which users have the right to change their passwords.

Format

SET PASSWORD dev:[ppn]

Command Qualifiers	**Defaults**
/[NO]LOG	/LOG

Prompts

Old password: old password
New password: new password
New password again, for verification: new password

Command Parameters

dev:[ppn]
Specifies the disk and account for which the password is to be changed. Unless you have sufficient privileges, you can only change your own account's password. If you do not specify the disk, the default is SY0:; if you do not specify an account, the default is your own PPN.

Command Qualifiers

/LOG
/NOLOG
Specifies whether the following message, confirming that the password was changed, displays:

```
Password set for account dev:[ppn].
```

The default is /LOG.

Passwords can contain from 6 to 14 characters. The characters can be in uppercase, lowercase, or a combination. Valid characters are:

- A through Z
- a through z

- 0 through 9
- Any printable character (for example, $, *,#, +), including spaces, except the question mark (?) character.

DIGITAL recommends the following guidelines to minimize the chances that unauthorized users can discover your password either by trial-and-error guessing or a systematic search:

- Do not use names or words that could be readily associated with any user (for example, WRITER, GUEST, or any part of your name).
- Change your password at least once a month.

When you type your password, RSTS/E does not display (echo) your input; this helps ensure secrecy. To protect against typing errors that you cannot see when you enter your new password, you must type the new password twice. If you do make an error, an error message displays and the password remains unchanged.

Note

In some cases, the system manager may have modified your account with a special qualifier (/LOOKUP). If so, you will not be able to use a password over six characters long, and you can only use letters and digits. See your system manager if you have trouble changing your password.

REQUEST

The REQUEST command displays a message at the system operator's terminal.

Format	
REQUEST	["]message["]

When you use the REQUEST command to send a message to an operator, the message is displayed at the operator services console.

Command Parameters

message

Specifies the text of a message to be displayed at the operator services console. For example:

```
$ REQUEST Do we need more paper for the line printer? RET
```

The message text can have a maximum of 128 characters.

Terminal Status and Operations 5

This chapter describes various operations you perform on your terminal, including:

- Displaying and setting your terminal's characteristics
- Creating a terminal log file to save a copy of a terminal session

Use the SHOW TERMINAL command to display information about your terminal and the SET TERMINAL command to change that information. Use the OPEN/LOG_FILE and CLOSE/LOG_FILE commands to begin and end terminal logging to a file, and the SET LOG_FILE command to selectively enable and disable logging within the log file.

Displaying and Setting Terminal Characteristics

The SHOW TERMINAL command, described in this chapter, produces a display similar to the following:

```
$ SHOW TERMINAL (RET)

Terminal: KB1:          Device_Type: VT100

    Tab                 No Form             Lowercase           Scope
    TTSync              Hostsync            Resume=Control_C    Break
    Control=(R,T,C)     No Control=X        No Broadcast        CRFill=0
    Width=80            No Delimiter        Speed not settable  No Eight_Bit
    Interactive
```

Now, suppose you want to change the No Broadcast characteristic (which prevents messages from other users from appearing on your terminal) to Broadcast (which allows these messages). You use the following SET TERMINAL command to change this characteristic:

```
$ SET TERMINAL/BROADCAST (RET)
```

This change (replacing "No Broadcast" with "Broadcast") is now shown in the SHOW TERMINAL display:

```
$ SHOW TERMINAL (RET)

Terminal: KB1:          Device_Type: VT100

    Tab                 No Form             Lowercase           Scope
    TTSync              Hostsync            Resume=Control_C    Break
    Control=(R,T,C)     No Control=X        Broadcast           CRFill=0
    Width=80            No Delimiter        Speed not settable  No Eight_Bit
    Interactive
```

Many of these characteristics (such as CRFill=n, Parity, and TTYSYNC) need to be set only once for the terminal, so you may never need to reset them. Usually the system manager sets these characteristics when the terminal is first connected to the system. However, characteristics such as Broadcast and Width=n depend on your preference for using the terminal. See the SET TERMINAL command in this chapter for a description of characteristics you can set.

There are no true "default" characteristics. That is, if you do not specify a characteristic, it is either implied by a terminal type (such as /VT100), or it remains constant when you change other settings. A terminal characteristic remains the same unless you either change it or specify a different terminal type. The SET TERMINAL command description lists the terminal types you can specify.

Although RSTS/E assigns a set of predefined characteristics for each specific terminal type, you can reset any of these characteristics. For example, suppose you type:

```
$ SET TERMINAL/DEVICE_TYPE=VT240 (RET)
```

RSTS/E now assumes that your terminal has a set of characteristics that correspond to a VT240. You can check these characteristics with the SHOW TERMINAL command, and change them if you want.

SHOW TERMINAL

The SHOW TERMINAL command displays the current characteristics of your terminal. Each of these characteristics can be changed with a corresponding option of the SET TERMINAL command.

Note

The SHOW TERMINAL command has additional capabilities for users with greater levels of privilege. See the *RSTS/E System Manager's Guide* for more information.

Format

SHOW TERMINAL [KBn[:]]

Command Qualifiers	Defaults
/BRIEF	/BRIEF
/FULL	/BRIEF
/[NO]PERMANENT	/NOPERMANENT

Command Parameters

[KBn[:]]

Specifies the terminal whose characteristics are to be displayed. Unless you have additional privilege to specify other terminals, you can only display the characteristics of your own terminal.

Command Qualifiers

/BRIEF

Specifies a brief display; it is the default. For example:

```
$ SHOW TERMINAL (RET)
Terminal: KB1:          Device_Type: VT100

    Tab                 No Form             Lowercase           Scope
    TTSync              Hostsync            Resume=Control_C    Break
    Control=(R,T,C)     No Control=X        Broadcast           CRFill=0
    Width=80            No Delimiter        Speed not settable  No Eight_Bit
    Interactive
```

/FULL

Specifies a display showing additional characteristics. For example:

```
$ SHOW TERMINAL/FULL (RET)
Terminal: KB1:          Device_Type: VT100

    Tab                 No Form             Lowercase           Scope
    TTSync              Hostsync            Resume=Control_C    Break
    Control=(R,T,C)     No Control=X        Broadcast           CRFill=0
    Width=80            No Delimiter        Speed not settable  No Eight_Bit
    Interactive         No Parity           No Escape_Sequence  No Local_Echo
    Up_Arrow            No Alt_Mode         Restricted          No Dialup
    No Autobaud         No ANSI             No Advanced_Video   132_Columns
    No Printer_Port     No Regis            Sixel               No Katakana
    No Select_Erase     No Loadable_Characters
    No User_Defined_Keys
```

/PERMANENT

/NOPERMANENT

Specifies whether the display shows the permanent characteristics set for your terminal by the system manager. The default is /NOPERMANENT, showing the characteristics you have set for the current terminal session. When you log out, RSTS/E resets your terminal to the permanent characteristics.

SET TERMINAL

The SET TERMINAL command lets you specify the characteristics of your terminal.

Note

The SET TERMINAL command has additional capabilities for users with greater levels of privilege. See the *RSTS/E System Manager's Guide* for more information.

Format

SET TERMINAL [KBn[:]]

Command Qualifiers	Command Qualifiers (Cont.)
/[NO]132_COLUMNS	/LOWERCASE[=INPUT:OUTPUT]
/[NO]ADVANCED_VIDEO	/[NO]PARITY[=option]
/[NO]ALT_MODE	/[NO]PRINTER_PORT
/[NO]ANSI	/[NO]REGIS
/[NO]BREAK	/RESET
/[NO]BROADCAST	/RESUME=[ANY:CONTROL_C]
/[NO]CONTROL=[(option[,option,...])]	/SCOPE
/[NO]CRFILL[=n]	/[NO]SELECT_ERASE
/[NO]DELIMITER[=c]	/SETUP=file-name
/DEVICE_TYPE=terminal type	/[NO]SIXEL
/[NO]EIGHT_BIT	/SPEED=(input[,output)]
/[NO]ESCAPE_SEQUENCE	/[NO]TAB
/[NO]FORM_FEED	/[NO]TTSYNC
/HARDCOPY	/TYPE=n
/[NO]HOST_SYNC	/[NO]UP_ARROW
/INQUIRE	/UPPERCASE[=INPUT:OUTPUT]
/[NO]KATAKANA	/[NO]USER_DEFINED_KEYS
/[NO]LOADABLE_CHARACTERS	/WIDTH=n
/[NO]LOCAL_ECHO	

Command Parameters

[KBn[:]]

Specifies the terminal whose characteristics are to be set. Unless you have sufficient privilege to specify other terminals, you can only set the characteristics of your own terminal.

Command Qualifiers

/132_COLUMNS
/NO132_COLUMNS
Specifies whether the terminal can display single-width characters in an array of 24 lines by 132 columns.

/ADVANCED_VIDEO
/NOADVANCED_VIDEO
Specifies whether the terminal supports the Advanced Video Option (AVO), which includes additional character attributes, screen memory, and ROM sockets.

/ALT_MODE
/NOALT_MODE
Specifies whether the system treats ASCII 027, 125, and 126 as ESCAPE. If you specify /NOALT_MODE, the system treats only ASCII 027 code as ESCAPE.

/ANSI
/NOANSI
Specifies whether the terminal supports ANSI escape sequences.

/BREAK
/NOBREAK
Specifies whether the system translates the BREAK key as a CTRL/C character (Binary 3). If you specify /NOBREAK, the system translates the BREAK key as a NULL character (Binary 0).

/BROADCAST
/NOBROADCAST
Specifies whether the terminal receives system broadcast messages.

/CONTROL=[(option[,option,...])]
/NOCONTROL=[(option[,option,...])]
Specifies the following:

/CONTROL=R enables the CTRL/R retype facility.
/NOCONTROL=R disables the CTRL/R retype facility.

/CONTROL=T enables the CTRL/T job status display facility.
/NOCONTROL=T disables the CTRL/T job status display facility.

/CONTROL=C means that CTRL/Cs are trapped and handled.
/NOCONTROL=C means that CTRL/Cs are ignored by the system.

/CONTROL=X means that CTRL/X clears the type-ahead buffer.
/NOCONTROL=X means that CTRL/X is interpreted as data.

To set more than one of these options on a single command line, you can combine them in a list. For example:

```
SET TERMINAL/CONTROL=(R,T,C)
```

/CRFILL[=n]
/NOCRFILL

Specifies the carriage return fill factor, where n is between 0 and 6. The default is /NOCRFILL, meaning that no fill characters are generated.

/DELIMITER[=c]
/NODELIMITER

Specifies a private delimiter. If the argument is a character within quotation marks (for example, /DELIMITER="."<tab>"), that character becomes the private delimiter.

If the argument is a number (for example, /DELIMITER=9), the private delimiter becomes the character whose binary value matches that number. Thus, any ASCII character whose binary value is between 1 and 127 can be a private delimiter.

The /NODELIMITER qualifier cancels any private delimiter previously set with /DELIMITER.

/DEVICE_TYPE=terminal type

Sets the default characteristics for the specified terminal type. The following terminal types can be specified with this qualifier:

LA12	LA50	VK100	VT101	VT131
LA34	LA100	VT52	VT102	VT132
LA36	LA120	VT55	VT105	VT220
LA38	LA180S	VT100	VT125	VT240

Note

You can specify the following terminal types without explicitly using the /DEVICE_TYPE qualifier: LA34, LA36, LA38, LA120, VT52, or VT100.

/EIGHT_BIT
/NOEIGHT_BIT

Specifies whether the terminal supports eight bit characters.

/ESCAPE_SEQUENCE
/NOESCAPE_SEQUENCE
Specifies whether the system treats an ESC character (value 27) as an indication of an incoming escape sequence. The system does not echo either the ESC character or the characters in the sequence.

If you specify /NOESCAPE_SEQUENCE, the system treats the ESC character as a line terminator and echoes it as a dollar sign ($) character.

/FORM_FEED
/NOFORM_FEED
Specifies whether the hardware form feed and vertical tab control are enabled. The system transmits form feed and vertical tab characters without translation.

If you specify /NOFORM_FEED, the hardware form feed and vertical tab control are disabled. The system transmits four line feed characters in place of a form feed or vertical tab character.

/HARDCOPY
/SCOPE
Specifies whether the terminal is a hardcopy or a CRT display device. If you specify /HARDCOPY, then /NOTTSYNC is set by default.

If you specify /SCOPE, then /TTSYNC is set by default.

/HOST_SYNC
/NOHOST_SYNC
Specifies whether the terminal has special hardware that lets the computer interrupt character transmission by sending an XOFF character (Value 19) to the terminal. The computer instructs the terminal to resume character transmission by sending an XON character (Value 17) to the terminal.

/INQUIRE
Interrogates the terminal by sending an ANSI ESCAPE identifying sequence to determine the terminal type. The system then sets the appropriate terminal characteristics.

If the terminal does not respond within five seconds, the system searches the default file $TERDFL.SYS for the terminal's default characteristics. If an entry is found, the system sets the specified characteristics; if no entry is found, the system displays an error message.

This qualifier causes the terminal's type-ahead buffer to be cleared.

You should only use the /INQUIRE qualifier on DIGITAL terminals. (However, DIGITAL does not support it on LA36 terminals.)

/KATAKANA
/NOKATAKANA
Specifies whether the terminal supports the Katakana character set.

/LOADABLE_CHARACTERS
/NOLOADABLE_CHARACTERS
Specifies whether the terminal supports dynamically redefinable character sets, allowing the terminal to load fonts which can then be invoked as a character set.

/LOCAL_ECHO
/NOLOCAL_ECHO
The /LOCAL_ECHO qualifier specifies that the system does not echo characters received from the terminal. This qualifier is used only for terminals that have their own local echo.

If you specify /NOLOCAL_ECHO, characters are sent to the system, which echoes each character received and translates certain characters as to their action. For example, the CR character echoes as a carriage return and line feed sequence.

/LOWERCASE[=INPUT:OUTPUT]
/UPPERCASE[=INPUT:OUTPUT]
Specifies the following:

/LOWERCASE enables lowercase for input and output.
/UPPERCASE disables lowercase for input and output.

/LOWERCASE=INPUT enables lowercase for input.
/LOWERCASE=OUTPUT enables lowercase for output.

/UPPERCASE=INPUT disables lowercase for input.
/UPPERCASE=OUTPUT disables lowercase for output.

If you specify /LOWERCASE or /LOWERCASE=INPUT, then /NOALT_MODE is set by default. If you specify /UPPERCASE or /UPPERCASE=INPUT, then /ALT_MODE is set by default.

/PARITY[=option]
/NOPARITY
Specifies whether EVEN or ODD parity option is set, in which case the system sends characters to the terminal with the parity bit set for either even or odd parity, but ignores the parity bit on characters received.

If you specify /NOPARITY, the system ignores the parity bit on characters it receives and treats the parity bit on characters it transmits to the terminal as a data bit.

/PRINTER_PORT
/NOPRINTER_PORT
Specifies whether the terminal has a printer port.

/REGIS
/NOREGIS
Specifies whether the terminal supports the Remote Graphic Instruction Set (ReGIS).

/RESET
Resets the terminal's characteristics, if you have changed them, back to the permanent set of characteristics that the system manager originally assigned for that terminal.

/RESUME=[ANY:CONTROL_C]
Defines XON/XOFF processing. The argument ANY enables typeout and echo when you type any character after XOFF. The argument CONTROL_C enables typeout and echo only when XON or CTRL/C are entered after XOFF.

/SELECT_ERASE
/NOSELECT_ERASE
Specifies whether the terminal supports the selectively eraseable character attribute.

/SETUP=file-name
Sends the specified file's data, in binary mode, to the terminal. If you specify the file name alone, the system assumes the file is in the user's account on the system disk with a file type of .ESC.

Use this qualifier to initialize a terminal for which special software settings must be made.

/SIXEL
/NOSIXEL
Specifies whether the terminal supports Sixel Graphics, which transfers binary graphic images between the system and the terminal or the terminal and a printer.

/SPEED=(input[,output)]
Specifies the speed at which the terminal sends and/or receives data. If you specify only one speed, RSTS/E uses that speed for both input and output.

/TAB
/NOTAB
Specifies whether hardware tab control is enabled. The system transmits tab characters without translation.

If you specify /NOTAB, hardware tab control is disabled. To move to the next tab stop, the system transmits the correct number of space characters instead of the tab character.

/TTSYNC
/NOTTSYNC
Specifies whether the system responds to the CTRL/S combination (XOFF) by interrupting character transmission and to the CTRL/Q combination (XON) by resuming character transmission.

If you specify /NOTTSYNC, the CTRL/S and CTRL/Q combinations have no special meaning.

/TYPE=n
Sets the value of the terminal's type code, which identifies the terminal type in the SHOW TERMINAL display. Values 0-127 are reserved for DIGITAL terminals. Values 128-255 can be used to identify other terminals.

/UP_ARROW
/NOUP_ARROW
Specifies whether the system echoes a control and graphic character combination as the circumflex (^) character (Value 94), followed by the proper graphic. For example, CTRL/E prints out as ^E.

If you specify /NOUP_ARROW, the system echoes control and graphic character combinations as is.

/USER_DEFINED_KEYS
/NOUSER_DEFINED_KEYS
Specifies whether the terminal supports User Defined Keys (UDKs), which let you save a full command string and invoke it with a single key.

/WIDTH=n
Sets the width of the terminal's print line to n, which can be between 1 and 254.

Creating a Log File of a Terminal Session

This section tells you how to create and use a terminal log file to save a copy of the output that appears during a session at your terminal. After you open the log file, all characters that are written to the terminal are saved in the file. Note that you can use a terminal log file either at the interactive level or within a command procedure. Whenever a log file is open and terminal logging is in effect, DCL changes its dollar sign prompt ($) to a dollar sign followed by a period ($.), as a reminder that logging is in effect.

The characters written to a log file are the same characters output to the terminal, with the following exceptions:

- Messages broadcast to the terminal do not appear in the log file.
- If you enter CTRL/T at the terminal, neither the ^T nor the one line status display appear in the log file.
- If you enter CTRL/R at the terminal, neither the ^ R nor the redisplayed line appear in the log file.

Note that any condition or command that disables terminal output, such as entering CTRL/O at the terminal, also disables output to the log file. In addition, if you enable a log file during execution of a command procedure, the SET NOVERIFY command disables output of DCL command lines both to the terminal and to the log file. The SET NOECHO command disables output of data and command lines to the terminal only; output continues to be written to the log file.

The following sections describe the commands that you use to:

- Open a terminal log file
- Close a terminal log file
- Selectively disable and enable output to the log file

OPEN/LOG_FILE

The OPEN/LOG_FILE command opens a disk file for terminal logging. You can issue the OPEN/LOG_FILE command either interactively or within a command procedure.

You can only open one log file at any given time. If a log file is already open when you attempt to open a second log file, RSTS/E issues an error message and does not open the second file; instead, the current log file remains open.

Format

OPEN/LOG_FILE file-spec

Command Qualifiers	Defaults
/APPEND	See discussion
/DISABLE	/ENABLE
/ENABLE	/ENABLE
/[NO]REPLACE	See discussion
/[NO]TIME_STAMP	/NOTIME_STAMP

Prompts

File: file-spec

Command Parameters

file-spec

Specifies the disk file to open for logging terminal output. RSTS/E displays an error message if you specify a nondisk file, or if the file-spec is invalid. RSTS/E also displays an error message if a log file is already open. If you do not specify a file type, RSTS/E assumes a file type of .LOG. Unless you include the /APPEND qualifier, RSTS/E opens the file for output, and deletes any existing file of the same name.

Command Qualifiers

/APPEND

Tells RSTS/E to add data to the end of the file you specify. This qualifier lets you append terminal output to the end of an existing log file. If the file you specify does not exist, then RSTS/E ignores this qualifier and opens a new file. Note that only RSTS/E stream ASCII files can be appended: if you specify a file that has RMS attributes, an error message displays.

/DISABLE

Indicates that terminal output should not be logged to the file until you issue the SET LOG_FILE/ENABLE command to enable terminal logging.

/ENABLE
Indicates that output should be logged to the file. This qualifier is the default when you issue the OPEN/LOG_FILE command.

/REPLACE
/NOREPLACE
Specifies whether to replace an existing file with a new file. If the log file that you specify already exists:

- The /REPLACE qualifier tells RSTS/E to delete the existing file and create a new file.
- The /NOREPLACE qualifier tells RSTS/E not to replace the file if it exists; in this case, an error message displays.

If you specify neither qualifier, then RSTS/E issues a warning message and asks if you want to replace the file.

Note that this qualifier conflicts with the /APPEND qualifier.

/TIME_STAMP
/NOTIME_STAMP
Indicates whether to prefix each line in the log file with a date/time stamp. The default is /NOTIME_STAMP. However, if you specify /TIME_STAMP, RSTS/E prefixes each line in the log file with a date/time stamp specifying the date and time that the line was copied to the log file. Note that the date and time format is based on the system defaults that your system manager establishes. The date/time fields occupy the first 22 characters of each line.

Here is an example using the OPEN/LOG_FILE command:

```
$ OPEN/LOG_FILE/TIME_STAMP TERM.LOG (RET)
$.
          .
          .
          .
```

This command opens the log file TERM.LOG. The TIME_STAMP qualifier specifies that the date and time is to precede each line written to the log file. Note that after you open the log file, the prompt changes to $. to remind you that a log file is active.

CLOSE/LOG_FILE

The CLOSE/LOG_FILE command closes a log file that you opened with the OPEN/LOG_FILE command.

Format

CLOSE/LOG_FILE

Use the CLOSE/LOG_FILE command to close an open log file either during a session at your terminal or within a command procedure. The RSTS/E monitor makes sure that any open log file is closed whenever you log out of the system or whenever your job is killed.

If logging is enabled when you issue the CLOSE/LOG_FILE command, then RSTS/E writes the command itself to the log file, before closing it. If no log file is open when you issue this command, then RSTS/E displays an error message. For example:

```
$ OPEN/LOG_FILE DIR.LOG

$.DIRECTORY

 Name .Typ      Size     Prot      Name .Typ     Size     Prot     SY:[52,20]
FINISH.COB       130    < 60>     LETTER.MEM       4    < 60>
START .BAS        89    < 60>     ROAD  .MAP         3    < 60>
TEMP16.TMP       258    < 60>

Total of 484 blocks in 5 files in SY:[52,20]


$.CLOSE/LOG_FILE
```

This example:

1. Opens the log file DIR.LOG with the OPEN/LOG_FILE command.
2. Lists the files in your directory, which are also written to the log file.
3. Closes the log file with the CLOSE/LOG_FILE command.

SET LOG_FILE

After you open a log file, you can use the SET LOG_FILE command to selectively enable and disable logging to the file. You can also use this command to enable or disable the time stamp feature in the log.

Format

SET LOG_FILE

Command Qualifiers

/DISABLE
/ENABLE
/[NO]TIME_STAMP

Command Qualifiers

/DISABLE
Indicates that terminal or command file output should not be logged to the current log file. If logging is currently disabled, then RSTS/E ignores this qualifier. This qualifier conflicts with /ENABLE.

/ENABLE
Indicates that terminal or command file output to the log file should be enabled. If logging is currently enabled, then RSTS/E ignores this qualifier. This qualifier conflicts with /DISABLE.

/[NO]TIME_STAMP
Enables or disables the time stamp feature, which prefixes each line in the log file with a date/time stamp in a format that your system manager has defined.

Use the SET LOG_FILE/DISABLE command to temporarily suppress output to the current log file. After you disable logging, RSTS/E displays the $ prompt to indicate that logging is disabled.

If you issue the SET LOG_FILE/DISABLE command when no log file is open, RSTS/E displays an error message. If you issue the command when logging is already disabled, then RSTS/E ignores the command. For example:

```
$ OPEN/LOG_FILE TEMP.LOG
          .
          .
          .
$ INQUIRE FILE "Enter the name of the file to edit"
$ SET LOG_FILE/DISABLE
$ EDIT 'FILE'
$ SET LOG_FILE/ENABLE
          .
          .
          .
$ CLOSE/LOG_FILE
$ EXIT
```

In this command procedure you:

1. Open the log file TEMP.LOG and write output to the file.
2. Use the INQUIRE command to prompt for the name of a file to edit.
3. Issue the SET LOG_FILE/DISABLE command to temporarily disable logging because you do not want to log the editing session.
4. Use the SET LOG_FILE/ENABLE command to reenable logging after the editing session ends, and continue writing output to the file.
5. Issue the CLOSE/LOG_FILE command to close TEMP.LOG before the procedure exits.

Use the SET LOG_FILE/ENABLE command to reenable output to an open log file. After you reenable logging, RSTS/E displays a $ prompt followed by a period ($.), to indicate that logging is in effect. If you issue the SET LOG_FILE/ENABLE command when no log file is open, RSTS/E displays an error message. If you issue the command when logging is already enabled, then RSTS/E ignores the command.

Note that you can also use this command to enable or disable the time stamp feature, as well as enabling or disabling output to the current log file. For example:

```
                .
                .
                .
$. !Disable logging temporarily
$.SET LOG_FILE/DISABLE
                .
                .
                .
$  !Now reenable logging
$ SET LOG_FILE/ENABLE
                .
                .
                .
$. !Now disable time stamps
$.SET LOG_FILE/NOTIME_STAMP
                .
                .
                .
$. !Now reenable time stamps
$.SET LOG_FILE/TIME_STAMP
$.
                .
                .
                .
```

In this example, you temporarily disable and reenable logging, and then temporarily disable and reenable time stamps in the log file.

Working with Devices 6

All RSTS/E users share the system's peripheral devices, such as line printers and magnetic tape drives. You can refer to these devices by either physical or logical device names.

Table 6-1 summarizes the commands you use to work with devices.

Table 6-1: Commands for Devices and Logical Names

Command	Meaning
ALLOCATE	Reserves a device so that only you can use it
DEALLOCATE	Releases a device from your exclusive use, so that other users may access the device
MOUNT	Logically loads a magnetic tape or disk onto a device
INITIALIZE	Clears ("zeros") a tape for use
DISMOUNT	Logically unloads a magnetic tape or disk from a device
SHOW DEVICE	Displays devices that are in use on your system
SHOW DEVICE/ALLOCATED	Displays status of allocated and open devices
SHOW DISKS	Displays status of all mounted disks
ASSIGN	Assigns a logical name to a directory or a physical device
DEASSIGN	Cancels logical name assignments you made with the ASSIGN command

Working with Physical Devices

Each of the devices on a RSTS/E system has its own device name. A physical device name is also called a device designator or device specification (because you use it to specify a device).

The system assigns physical device names to peripheral devices, such as:

- Line printers (LP:)
- Terminals (KB:, for "keyboard")
- Magnetic tape drives (MM:, MT:, or MS:, depending on the kind of tape drive)
- Disk drives (such as DB:, DM:, or DR:, depending on the kind of disk)

Note that a physical device name consists of two letters, optionally followed by a unit number, and always ended with a colon (:). The alphabetic characters are generally an abbreviation of the device's generic name. The unit number uniquely identifies the device itself or the drive on which it is mounted.

You can use physical device names in all operations that let you specify a device. For example, you can display the characteristics of a specific terminal by including a physical device name, such as KB2:

```
$ SHOW TERMINAL KB2: RET
```

Similarly, you can determine which disk or magnetic tape you want to use in file transfer operations. For instance, assume you have a tape mounted on a tape drive named MT2:. To copy file RACE.ME in your directory to the tape on drive MT2:, type:

```
$ COPY RACE.ME MT2: RET
```

Another example of a physical device name is SY:, which refers to the public disk structure. You can therefore use SY: to refer to the public disk structure, regardless of the drive on which a disk is located. Such an assignment is useful if, for example, the drive on which the disk is usually mounted is put offline for maintenance, and the system disk is temporarily mounted on another drive. Then, in order to reference accounts on the system disk, you do not need to adapt your commands or programs to fit hardware changes.

If you do not specify a physical device name, the system assumes the public structure (SY:). For non-file-structured devices (for example, a line printer) you only need to specify the physical device name; the system ignores any specified file name, type, or directory.

Physical device names are also often used in logical name assignments (discussed later in this chapter). The following example assigns the logical name MARCH to DB2:, which is a disk drive:

```
$ ASSIGN DB2: MARCH RET
```

Table 6-2 lists the RSTS/E physical device names.

Table 6-2: RSTS/E Physical Device Names

Designator	Device
DK:,DL:,DM:, DP:,DR:,DB:, DU:, or SY:	RSTS/E public disk structure
SY0:	System disk
DK0: to DK7:	RK05 disk units 0 through 7
DL0: to DL3:	RL01/RL02 disk units 0 through 3
DM0: to DM7:	RK06/RK07 disk units 0 through 7
DP0: to DP7:	RP02/RP03 disk units 0 through 7
DR0: to DR7:	RM02/RM03/RM05/RM80 disk units 0 through 7
DB0: to DB7:	RP04/RP05/RP06 disk units 0 through 7
DU0: to DU7:	RA80/RA81/RA60/RC25/RD51/RX50 units 0 through 7
PP:	High speed paper tape punch
PR:	High speed paper tape reader
CR:	CR11 punched or CM11 mark sense card reader
CD:	CD11 punched card reader
MT0: to MT7:	TE10/TU10/TS03 magnetic tape units 0 through 7
MM0: to MM7:	TE16/TU16/TU45/TU77 magnetic tape units 0 through 7
MS0: to MS3:	TS11/TU80/TSV05 magnetic tape units 0 through 3
LP0: to LP7:	Line printer units 0 through 7
DT0: to DT7:	TU56 DECtape units 0 through 7
KB:	Your terminal
TT: or TI:	Your terminal (synonyms for KB:, the terminal that initiated the job)
KBn:	Terminal n on the system
TTn:	Terminal n on the system (synonym for KBn:)
NL:	The null device
PKn:	Pseudo keyboard n
DX0: to DX7:	RX01/RX02 flexible diskette (floppy disk) units 0 through 7

Note

You can reference LPn:, DTn:, DXn:, KBn:, MMn:, MTn:, and MSn: where n is between 0 and the maximum number of such units on the system. LP:, DT:, DX:, MM:, MT:, and MS: are each the same as specifying unit 0 of the related device.

If your system has only TS11, TU80, or TSV05 tape units, the designators MS: and MT: are the same. If your system has only TE16, TU16, TU45, or TU77 tape units, the designators MM: and MT: are the same.

If you do not specify a unit number, the defaults are:

Disk — Public structure
Keyboard — Yours
All other devices — Unit 0

Use the SHOW DEVICE command, described later in this chapter, for a list of devices in use on your system. If you specify a device or type of device that is not part of your system's configuration, the system displays the following error message:

```
?Not a valid device
```

An underscore (_) before a name suppresses any interpretation of the name as a logical name. For example, if you assign the logical name MT0: to a disk, you specify _MT0: when you want to refer to magnetic tape unit 0.

Allocating Devices

Sometimes you may need to allocate (reserve), a device for your exclusive use. Only one job at a time can use an allocated device.

You can only allocate devices that are not already in use. For example, if another user is already using a tape drive, you cannot immediately allocate it to yourself. If you attempt to do so, the system displays the following error message:

```
?Device not available
```

Assigning and Allocating Devices

The ALLOCATE command allocates a device to you. ALLOCATE can also assign a logical name for the device you are using.

Use the ALLOCATE command to assign logical names and allocate devices for your use. Use the ASSIGN command to assign logical names only. In either case, you get an error message if you try to exceed the number of logical names you can assign.

The DEALLOCATE command releases a device from your exclusive use, but only the DEASSIGN command cancels the logical name assignment.

Device Independence

Device independence is a central idea in RSTS/E. This means that you can use a file on one device much as you can use a file on any other device. This idea applies both to the system user and to the programmer.

As much as possible, the system tries to hide the differences between devices from both the system user and the programmer. For example, you can read data that is on a tape just as you read data that is on a disk.

However, different devices offer different capabilities. RSTS/E cannot entirely hide these differences. The following sections discuss some of the differences among devices.

Working with Disks

A disk is either public or private, as determined by your system manager. Any user who has an account on the system can create files on a public disk. If a disk is private, you cannot place files there unless your system manager (or a user with sufficient privileges) creates an account for you on that disk.

Every RSTS/E installation has at least one public disk (and sometimes more), called the system disk. For most purposes, the public disks on a given system are treated as a unit. This unit is called the public disk structure, and has the device name SY: (which stands for system).

When you create a file and do not specify what device to create it on, or when you specify SY:, the system creates the file on one of the public disks. (The system also decides which public disk to create it on if there is more than one public disk.) When you specify an existing file and do not specify a device (or when you specify SY:), the system searches each of the public disks for the file.

You cannot have two files of the same name and type on two different public disks, just as you cannot have two files of the same name and type on a single disk. If you try to give a file the same specification as an existing file, one of the following two things can happen (depending on the command):

- You get the following error message:

```
?Name or account now exists
```

- The existing file is replaced

The Public Disk Structure

Users share computing time on the RSTS/E system. Each user is allocated a "slice" of the processor's time. Users may share another system resource as well: the array of devices. One set of devices (disks) is shared by a number of users. This shared set of devices is called the public disk structure, because it is always accessible to all users and because the system treats it as a unit. On some systems, it may include many separate disk packs. One of these, the system disk, contains the system code,

language processors, and, possibly, the library of system programs. The other disk packs (the public disks) contain information created by the users. (The system disk also may contain user programs and data, which are stored in files.)

When you are logged in to the system and working with a disk file, you are usually not concerned about which disk in the public structure happens to contain your file. The particular disk is chosen by the system according to current time-sharing needs. Each of the disks contains a master list of users' accounts, and, for each account, a list of all files stored under that account. By using these lists, the system is able to locate your file when you request it from your terminal.

Private Disks

A good deal of system file activity, such as creation, access, editing, and deletion, takes place on the public disk structure. Because it is the largest constantly available medium of file storage, it is generally the busiest. However, not all the disks on the system need be in the public structure. Some of them may be private disks, which belong to a single user account or perhaps to a few user accounts. (These accounts alone are on the disk.)

An important difference between a private disk and one on the public structure is that you can always create a file on the public structure, whereas you can do so on a private disk only if you have a directory there. On both types of disks, file protection codes govern your read and write access to existing files. Another difference is that a private disk can be mounted or dismounted at any time, whereas a public disk must remain mounted during timesharing. This difference is of special concern to the owner(s) of the private disk and to your system manager, who determines which disks on the system are public and which are private.

Working with Magnetic Tapes

Magnetic tape is a compact, relatively inexpensive medium that can provide large amounts of offline storage. One reel of magnetic tape can store many files.

Unlike disks, magnetic tape allows only sequential access to files.

To copy files from a tape, you should:

1. Allocate the drive. (This step is optional, but it provides security.)
2. Physically mount the tape.
3. Issue the MOUNT command. (This step is optional, if the tape has default density and format.) The MOUNT command also allocates the drive, if you did not allocate it already.
4. Copy files from the tape using the COPY command. (With the COPY command, you can copy a file that requires multiple tapes. User programs can access tape files if they are stored on only one tape, but system programs can access files stored on more than one tape.)

5. Issue the DISMOUNT command. (This step is optional, but it deallocates the drive and rewinds the tape.)
6. Physically dismount the tape.

Protecting Files on Tapes

Protection of files on tapes works differently from protection of files on disks. While you are using a tape, the device is allocated to your job. While the device is allocated to you, you have complete control over the tape. You can read any file on it. No other user can access the tape while it is allocated to your job. The MOUNT command and the ALLOCATE command both allocate the tape device to your job. The DISMOUNT, DEALLOCATE, and LOGOUT commands free the tape device so that other jobs can allocate it.

Tape Density

When you initialize a tape on a device, you decide what the tape's density will be. The two considerations are:

- The amount of information to be stored on the tape
- The degree of portability (interchangeability) between systems

For example, assume you have a choice of specifying a density of 800 or 1600 bits per inch (bpi). You can get roughly twice as much information on a tape with a density of 1600 bpi; however, not all tape drives support 1600 bpi.

ANSI and DOS Format

RSTS/E has two file structures for tapes:

- ANSI format, because it complies with the standards of the American National Standards Institute.
- DOS format, because RSTS/E adopted it from the Disk Operating System format.

The ANSI format is an industry standard. The advantages to using ANSI format tapes are:

- Portability — If you expect to move the contents of a tape from one system to another, it is very possible that the other systems will recognize ANSI format.
- Multiple tapes — The files you copy can be stored on more than one reel of tape.

Note

Where ANSI is used in RSTS/E documentation, it refers to the RSTS/E implementation of American National Standard X3.27-1978, magnetic tape labels and file structure for information exchange. RSTS/E implements a subset of this standard.

In addition, RSTS/E uses U (undefined) record format, which is not defined in ANSI standard X3.27-1978.

The advantages of using DOS tape are:

- Files are associated with PPNs.
- Files are better for storing binary files that have no attributes, such as .OBJ (object) or .BAC (compiled BASIC-PLUS) files.

When you initialize a tape with the INITIALIZE command, you can select either ANSI or DOS format. Later, when you mount the tape, you must again tell the system whether the tape is in ANSI or DOS format. If you do not specify a format, the system uses the default determined by your system manager.

There are two ways to tell if a tape containing data is written in DOS or ANSI format:

- Look for a label on the tape. The person who put data on the tape may have placed a label on the tape and written DOS or ANSI on it.
- Try mounting the tape as either DOS or ANSI. When you mount a tape on a drive, RSTS/E checks the format. If you entered the right format, you can continue with tape operations. If you entered the incorrect format, RSTS/E displays the error message ?Bad directory for device, in which case you can try again using the other format.

If the tape is in ANSI format, then files on the tape do not have PPNs. They have only names and types.

If the tape is in DOS format, then each file on the tape has a PPN. The PPN of a tape is like the directory on a disk file. For example, a DOS format tape on drive MT0: might contain a file whose specification is:

```
MT0:[1,2]SLOW.RUN
```

Another file on the tape might have the specification:

```
MT0:[3,49]GOOD.RUN
```

Physical Device Commands

This section contains descriptions of some commands you use when working with physical devices. See the *RSTS/E System Manager's Guide* for commands and functions that require additional privileges to use.

ALLOCATE

The ALLOCATE command reserves a physical device for your use during the current session and optionally establishes a logical name for the device. Once a device has been allocated, other users cannot access the device until you specifically deallocate it or log out. You can allocate a device only when it is not allocated by another job.

Format

ALLOCATE device-name[:] [logical-name[:]]

Prompts

Device: device-name

Command Parameters

device-name[:]
A logical or physical name.

logical-name[:]
Specifies a one- to six-character logical name to be associated with the device. RSTS/E automatically translates any references you make to the logical name to a physical device associated with the name. The logical name parameter is optional.

DEALLOCATE

The DEALLOCATE command releases a device that you reserved for private use, so that other users may have access to it. (However, note that DEALLOCATE does not deassign any logical name you may have assigned to the device; for that you need the DEASSIGN command.)

Format

DEALLOCATE [device-name[:]]

Command Qualifiers

/ALL

Prompts

Device: device-name[:]

Command Parameters

device-name[:]

Specifies the name of the device to be deallocated. The device name can be a physical device name or a logical name. The device name parameter is required unless you specify the /ALL qualifier.

Command Qualifiers

/ALL

Requests that all devices you have currently allocated be deallocated.

If you specify /ALL, you cannot specify a device name.

MOUNT

The MOUNT command prepares a tape or disk for processing by system commands or user programs. Issue the MOUNT command after you physically mount the tape or disk and put the drive on line.

Note

The MOUNT command has additional capabilities for users with greater levels of privilege. See the *RSTS/E System Manager's Guide* for more information.

If the MOUNT command fails with the error message ?Disk pack needs REBUILDing, but you do not have sufficient privileges, have your system manager mount the disk.

Format

For disks:

MOUNT device-name[:] pack-id [logical-name]]

For magnetic tapes:

MOUNT device-name[:] {[label]}

Command Qualifiers	**Defaults**
/[NO]WRITE	See description
Qualifiers for Disks	**Defaults**
/PRIVATE	/PRIVATE
/[NO]SHARE	/SHARE

Qualifiers for Tapes

/DENSITY=n
/FORMAT=ANSI
/FORMAT=DOS
/FORMAT=FOREIGN

Prompts

Device:	device-name[:]
Label:	label (for ANSI tapes only)
Pack-id:	pack-id (for disks only)

Depending on whether you are mounting a tape or disk, the MOUNT command:

- Specifies a tape's density and format
- Checks that a tape or disk's device specification is correct
- Checks that a tape or disk has been initialized
- Verifies that a tape or disk drive had not been allocated to another user
- Allocates a tape drive (reserves it for your use)
- Verifies that a disk or tape is physically loaded on the device you specified
- Verifies that the pack-id on a disk or the label on a tape (except for DOS tape) matches the label you specified
- Allows you to mount a private pack for your own use

Command Parameters

device-name[:]
Specifies the physical or logical name of the drive on which a device is to be logically mounted.

pack-id
logical-name
label
Identifies the tape or disk to be mounted. The pack-id, logical-name, or label can have from one to six alphanumeric characters.

The label is required when you mount a disk or a tape in ANSI format. It is not required (and is ignored) when you mount a tape in DOS or foreign format.

Command Qualifiers

/DENSITY=n
Specifies the density in bits per inch (bpi) at which the tape will be read or written. This qualifier is valid only with tapes.

You can specify 800 or 1600 bpi, depending on the density supported by the tape drive. The default density depends on your installation.

/FORMAT=ANSI
/FORMAT=DOS
/FORMAT=FOREIGN
Indicates whether the tape is in a standard format used by the RSTS/E operating system. This qualifier is valid only with tapes.

A tape is FOREIGN if it is not in ANSI or DOS format. If you mount a tape with /FORMAT=FOREIGN, the program you use to read the tape must be able to process any labels on the tape.

/PRIVATE

Allows you to mount a private disk, accessible only to those users who have accounts on the disk. (Your system manager designates whether disks are public or private.)

/SHARE
/NOSHARE

Determines whether the private disk you use is shared. The /NOSHARE qualifier allows you to mount a private disk that is accessible only to the job that mounts it. The /SHARE qualifier allows all users who have accounts on the disk to access it. The /PRIVATE and /SHARE qualifiers have the same meaning. If you specify /NOSHARE, you cannot specify /PRIVATE.

The default is /SHARE.

/WRITE
/NOWRITE

Controls whether or not you can write data to a tape or disk. The /WRITE qualifier allows you to modify the data contained on a tape or disk. If you specify /NOWRITE, you cannot modify the data.

You can specify /NOWRITE to provide read-only access to protect files. For a disk, this is equivalent to write-protecting it. For a tape, the drive itself must be write-protected; /NOWRITE simply generates an error message if the device is not write-protected.

When the device is not write-protected, /WRITE is the default for tapes. When the disk drive is not write-protected, /WRITE is also the default for disks (initialized as read/write). However, /NOWRITE is the default if the drive is write-protected. Therefore, if you receive the warning message %Device write protected, you have logically mounted your disk or tape with read-only access. The default is also /NOWRITE for disks initialized as read-only.

INITIALIZE

The INITIALIZE command deletes any data on a tape and writes a new label. INITIALIZE allocates the tape drive if it is not already allocated.

Use INITIALIZE to prepare a new tape or a tape that contains no useful files. Use MOUNT to prepare an already initialized tape that contains files you want to keep.

Note

See the *RSTS/E System Manager's Guide* for information on using INITIALIZE with disks.

Format

INITIALIZE device-name[:] [label]

Command Qualifiers

/FORMAT=ANSI
/FORMAT=DOS
/DENSITY=n

Prompts

Device:	device-name[:]
Label:	label
Proceed (Y or N):	

Note

The label prompt appears only for a tape in ANSI format or if /FORMAT=ANSI is specified. The proceed prompt asks you to confirm whether you want to initialize the tape. Type yes or no (Y or N) in response to the prompt.

Command Parameters

device-name[:]
Specifies the name of the drive on which the tape is physically mounted.

label
Specifies the identification to be encoded on the tape. You can specify a maximum of six alphanumeric characters. For an ANSI-format tape, the label is required. For a DOS-format tape, the label is ignored, because DOS tapes do not have labels.

Command Qualifiers

/DENSITY=n

Specifies the density in bpi at which to write the tape. You can specify a density of either 800 or 1600 bpi if the tape drive supports it.

If you do not specify a density, the system uses the default specified by your system manager.

/FORMAT=ANSI
/FORMAT=DOS

Specifies the tape format. If you do not specify a format, DCL uses the default determined by your system manager.

DISMOUNT

The DISMOUNT command releases a disk or tape previously accessed with a MOUNT command. Use the DISMOUNT command before you take the drive off line, or before you physically dismount the tape or disk.

Note

The DISMOUNT command has additional capabilities for users with greater levels of privilege. See the *RSTS/E System Manager's Guide* for more information.

The DISMOUNT command deallocates the device if it was allocated to you. You cannot DISMOUNT a device if there are open files on it. If you try, RSTS/E displays the error message:

```
?Account or device in use
```

Format

DISMOUNT device-name[:] [label]

Command Qualifiers (for tapes only)	**Defaults**
/[NO]UNLOAD	/UNLOAD

Prompts

Device: device-name[:]
Pack-id: pack-id label

Command Parameters

device-name[:]
Specifies the name of the device to be dismounted. If you omit a unit number, the default depends on the type of device.

label
Specifies the identification label of the device. Unless you have additional privilege, you must specify a disk's pack-id label to dismount it. For tapes, the label is optional (and is ignored).

Command Qualifiers

/UNLOAD
/NOUNLOAD
Rewinds and unloads the tape. This protects the tape against unauthorized access by other users. The default is /UNLOAD.

SHOW DEVICE, SHOW DEVICE/ALLOCATED, SHOW DISKS

There are three SHOW commands that display information about physical devices on the system. They are:

- SHOW DEVICE [dev:] — Displays all devices on the system, or a specified device
- SHOW DEVICE/ALLOCATED — Displays the status of all allocated and open devices
- SHOW DISKS — Displays the status of all mounted disks

Examples of each command, and their displays, follow:

```
$ SHOW DEVICE KB32: (RET)
Device _KB32:  (KBG3:)    Control DZ0:3    CSR 760140   Status: Restricted


$ SHOW DEVICE/ALLOCATED (RET)

Busy Devices:
Device       Job     Why
PK0          10                                  Open
DMR-0        TRN                                 AS+Open
DMC-1        TRN                                 AS+Open
DMP-0.0      TRN                                 AS+Open


$ SHOW DISKS (RET)

Disk Structure:
Dsk  Open     Size       Free     Clu    Err Name       Level  Comments
DB3     0   340664    15288   4%   8      0 DBL8         1,1   Pri, R-O, DLW, DP
DB4     2   340664    21920   6%   8      2 S            1,2   Pri, DLW
DU1    47   400168    41864  10%   8      0 GROKSY       1,2   Pub, DLW
```

See the *RSTS/E System Manager's Guide* for more information on each column type in the previous three examples.

Table 6-3 contains the abbreviations included in the SHOW DEVICE/ALLOCATED and SHOW DISKS displays.

Table 6-3: Abbreviations in SHOW DEVICE/ALLOCATED and SHOW DISKS Displays

Abbreviation	Meaning	Description
SHOW DEVICE/ALLOCATED Command		
AS	Assigned	Device is explicitly allocated to a job.
OPEN	Open	Device is open on a channel.
DOS	DOS	Magnetic tape is assigned with DOS (Disk Operating System) format.
ANSI	ANSI	Magnetic tapeis assigned with ANSI (American National Standards Institute) format.
TRN	Network Services	The monitor is using the device for DECnet communications.
SHOW DISKS Command		
Pub	Public	Disk is public.
Pri	Private	Disk is private.
NFS	Non-file-structured	Disk is open as a non-file-structured device.
R-O	Read-only	Disk unit is read-only (write-protected).
DLW	Date of Last Write	Date of last write (modify), rather than date of last access, is stored in file accounting entries.
DP	Dual-Ported	The disk can be accessed by two different RSTS/E systems at the same time. The disk must be read-only.
Lck	Locked	Disk is in a locked state.
NFF	New Files First	New files on the disk are put at the beginning of the directory.
Job n	Job n	Private disk is allocated to job n (reported with PRI and NFS).
Dirty	Dirty	Disk needs rebuilding (reported with PRI, LCK, and R-O).

Logical Names

Logical names let you keep programs and command files independent of the physical locations of those files. They also provide a covenient shorthand way to specify devices and directories that you refer to frequently.

You can assign a logical name to a physical device name or to a directory. The logical names that you assign are valid for your job only. For example, only your job can reference the logical name that you assign to a device. No other job can reference that logical name.

Logical Names and Devices

You can use logical device names to describe the nature of information on a disk, magnetic tape, or other medium. The logical name may be easier to remember than the physical name with its device number. You can choose a logical name without regard to what device types may be available at some future time.

The logical names that you assign for devices do not depend on the physical device specifications. Unlike a physical name, a logical name is independent of the drive on which the medium is mounted. Logical device names make it easier to adapt a program for use on different drives. Thus, if you write a program referencing physical devices, you can give these devices logical names of your own choosing by issuing the ASSIGN command. This action associates your chosen names with the devices and makes the program independent of the devices' physical locations on the system.

System-Wide and User Logicals

When you assign logical names, you are the only one who can use them. Your logical name assignments are deleted when you log out. The logical names that you assign are called user logical names, to distinguish them from system-wide logicals.

System-wide logicals are logical names that everyone on the system can use. Certain system-wide logicals are standard for most RSTS/E systems; others are assigned for your site by the system manager. See the *RSTS/E System Manager's Guide* for more information on system-wide logicals.

Numbers of Logical Names

A job can have a maximum of four logical name assignments at a time if you specify only device names. However, if one or more of the logical names is associated with a directory, the job is limited to three logical name assignments.

You can assign more than one logical name to a device. For example, you could assign to the tape on drive MM0: the logical names RACE1, RACE2, and RACE3, and use any of these three names as your application requires.

How to Override Name Precedence

Sometimes you may have a logical name that conflicts with a physical name.

The system always recognizes and accepts a device's physical name, whether or not a logical name has been assigned to that device. Thus, you can specify a magnetic tape running on drive MT2 and assigned the logical name STAR as either MT2: or as STAR:.

If you put an underscore before a physical name, the system will not try to interpret it as a logical name (for example, _MT2:). A physical device name preceded by an underscore causes the system to access that device, regardless of any prior assignment.

RSTS/E evaluates the parts of a file specification in the following sequence. When a file specification contains a name followed by a colon, RSTS/E first determines whether or not the name is a user logical. If it is, then the system replaces the name with its expansion. (An expansion is the device name or PPN to which the logical name was assigned.) Otherwise, it checks to see if the name is a system-wide logical.

If the name is not a system-wide logical, RSTS/E then determines whether the name designates a valid physical device on your RSTS/E system. (During system generation, your system manager selects which devices are valid on your system.)

Finally, if the device name is not a physical device name, the system displays the error message:

```
?Not a valid device
```

ASSIGN

The ASSIGN command assigns a logical name to a directory or a physical device. The names you assign stay in effect until you log out, or until you deassign the name with the DEASSIGN command.

Note

The ASSIGN command has additional capabilities for users with greater levels of privilege. See the *RSTS/E System Manager's Guide* for more information.

Format

ASSIGN device-name:[[ppn]] logical-name[:]

Prompts

Device: devnam:[[ppn]]
Logical name: logical-name

When you use the ASSIGN command, one of the following can occur:

- The command works as expected, and no message is displayed.
- The assignment statement is incorrect, and one of the following error messages displays:

```
?Not a valid device
```

```
?No file name or type permitted
```

- You exceed the maximum number of logical name assignments you can make, and the following error message displays:

```
?Too many logical names assigned
```

- You reassigned a logical name and the following informational message displays:

```
Previous logical name assignment replaced
```

Command Parameters

devnam:[[ppn]]
Specifies the name of the device or PPN to be assigned a logical name. Brackets around the PPN are required, as in the following example:

```
$ ASSIGN MT1:[2,214] WINDY:
```

logical-name[:]
Specifies a one- to six-character logical name to associate with the device.

The ASSIGN command equates a logical name to a part of a physical file specification. The following example equates the logical name EXCER: to the directory DB2:[3,24]:

```
$ ASSIGN DB2:[3,24] EXCER:
```

The string "DB2:[3,24]" is called the "expansion" of the logical name EXCER.

Later, you can refer to the directory by its logical name when you issue a RSTS/E command. For example:

```
$ DIRECTORY EXCER:
```

When the system executes this DIRECTORY command, it replaces the logical name EXCER: with its expansion, DB2:[3,24], and lists all of the files in that directory on your terminal.

Another example is:

```
$ TYPE EXCER:RUNNER.DAT
```

When the system executes this command, it replaces the logical name EXCER: with its expansion and displays the contents of the file DB2:[3,24]RUNNER.DAT on your terminal.

You can equate a logical name to a physical device name, to a directory, or to both. For example:

```
$ ASSIGN DB2: EXCER           !(DB2: is a physical device)
$ ASSIGN [3,24] EXCER:        !( [3,24] is a directory)
$ ASSIGN DB2:[3,24] EXCER:    !( [3,24] is a directory on DB2:)
```

When you assign a logical name, the partial file specification that you assign to the logical name may itself include a previously assigned logical name. When the system executes the ASSIGN command, it replaces the previously assigned logical name with its expansion and displays an informational message. Therefore, if you later change the assignment of the first logical name, the expansion of the second logical name is not affected. For example, consider the following sequence of commands:

```
$ ASSIGN DB2: WORK:
$ ASSIGN WORK:[3,24] EXCER:
$ DEASSIGN WORK:
$ TYPE EXCER:RUNNER.DAT
```

In this example, the logical name EXCER: corresponds to DB2:[3,24]. The TYPE command displays DB2:[3,24]RUNNER.DAT even though the logical name WORK: has been deassigned.

DEASSIGN

The DEASSIGN command cancels logical name assignments you made with the ASSIGN command.

Note

The DEASSIGN command has additional capabilities for users with greater levels of privilege. See the *RSTS/E System Manager's Guide* for more information.

Format

DEASSIGN [logical-name[:]]

Command Qualifiers

/ALL

Prompts

Logical Name: logical-name[:]

Command Parameters

logical-name[:]

Specifies a one- to six-character logical name to deassign.

A logical name is required unless you specify /ALL. If you do not specify either /ALL or a logical name, the system prompts you for a logical name. For example:

```
$ DEASSIGN (RET)
Logical Name: EXCER (RET)
```

Command Qualifiers

/ALL

Deletes all logical names that you assigned. For example:

```
$ DEASSIGN/ALL (RET)
```

Print/Batch Services 7

Whenever you use the PRINT command to print files or use the SUBMIT command to execute command procedures for batch processing, you are automatically using the RSTS/E Print/Batch Services (PBS) facility. This chapter describes the commands you use to:

- Place entries on a print or batch queue for printing or batch processing
- Check the status of your print or batch job on a queue
- Modify your print or batch entry before it is printed or processed
- Delete a print or batch entry from a queue

Your system manager usually sets up default print and batch queues for your system, so that you only need to use the simple commands this chapter describes to print files or submit command procedures for batch processing. If you are interested in, or need to know, more detailed information about how PBS performs its print and batch processing functions, see the complete description of PBS organization and operations in the *RSTS/E System Manager's Guide.*

Table 7-1 summarizes the PBS commands that this chapter describes.

Table 7-1: Print/Batch Services Commands

Command	Meaning
PRINT	Produces a listing, or "hard copy," of a file
SUBMIT	Submits a command file for batch processing
SHOW ENTRY	Displays the names of files that are currently being printed or processed, as well as files pending on the print or batch queue
SET ENTRY	Modifies one or more attributes of a file that is entered on a print or batch queue
DELETE/ENTRY	Cancels an entry on the print or batch queue
SHOW QUEUE	Displays the status of a print or batch queue.

Specifying Print and Batch Entries

The PRINT and SUBMIT commands instruct PBS to enter files on a print or batch queue, respectively. A PBS queue controls how print and batch system resources are allocated.

Once a file is queued for printing or batch processing, it becomes an "entry." You often may need to access your print or batch entry. For example, you may want to modify the entry or delete it, or you may want to display the entry's status in the queue so you can see whether it has started or finished printing or batch processing. PBS provides two methods of identifying and accessing entries in a queue:

- Entry number
- Entry specification

Entry Number

PBS assigns a unique entry-number to each queued print or batch entry. This number shows the order in which the entry was placed in the print or batch queue. Since it is unique, you can use the entry-number to identify and access your entry if you want to modify, delete, or display it.

For example, here is an entry on a print queue, as displayed by the SHOW ENTRY command:

```
1st  Print entry 5   PRINT:[   1,214]MAIL   Status STARTED  Pri 128
```

This display tells you that the entry-number is 5, which you can then use to identify the entry while it is in the queue.

Entry Specification

Besides the entry-number, you can also access an entry in a queue by specifying the following attributes:

- The name of the queue on which the entry is located
- The entry's owner (PPN)
- The entry name

PBS permanently establishes these attributes when it creates the entry; you cannot modify them once the entry is placed in the queue. The format of an entry-spec is:

queue-name:[PPN]entry-name

Thus, you would type the entry-spec for the entry shown in the previous example as follows:

```
PRINT:[1,214]MAIL
```

Unlike entry-numbers, entry-specs do not uniquely identify an entry. For instance, in the above example, you (as user [1,214]) could have several entries named MAIL on the print queue named PRINT:. You should therefore be very careful when using entry-specs to access an entry to modify or delete it; you might inadvertently modify or delete other entries having the same entry-spec.

Printing Files: PRINT

The PRINT command queues one or more files for printing, either on the default print queue or on a queue you specify.

Format

PRINT file-spec[,...] [entry-spec]

Command Qualifiers	Defaults
/AFTER=date-time	
/FORMS=form-name	See discussion
/HOLD	
/JOB_COUNT=n	/JOB_COUNT=1
/NAME=entry-name	See discussion
/OWNER=ppn	See discussion
/PAGE_LIMIT=n	See discussion
/PRIORITY=n	See discussion
/QUEUE=queue-name[:]	See discussion

File Qualifiers	Defaults
/[NO]CONVERT	/NOCONVERT
/COPIES=n	/COPIES=1
/[NO]DELETE	/NODELETE
/[NO]FEED	/FEED
/[NO]FLAG_PAGES	/FLAG_PAGES
/[NO]TRUNCATE	/NOTRUNCATE

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more files to be printed. If you specify more than one file, separate the file specifications with commas (,) or plus signs (+). In either case, the PRINT command prints all the files as a single print job. You can use wildcard characters for the file name, file type, and directory.

If you do not specify a file type, the PRINT command uses the default file type .LST. If you do not specify a device, the PRINT command uses SY:.

You can also specify a node name as part of the file specification. If you specify a file on a remote node, PBS temporarily copies the file into your account, prints it, and then deletes it. Note that you cannot use any qualifiers if you specify a node name.

entry-spec
queue-name:[ppn]entry-name

Specifies the queue and the name of the entry. The entry-name can contain from one to nine characters. If you do not specify a queue, PBS submits the entry on the default print queue. If you do not specify an entry-name, PBS assigns the first non-numeric file name in the list of specified file-specs, or PRINT if none is found. Your own PPN is the default.

Command Qualifiers

/AFTER=date-time
Requests that the file not be printed until a specific time of day or date. See Chapter 2 for a description of the standard syntax rules for specifying date and time values.

For example, to have a file named TIBET.MEM start printing after 5:30 PM on September 11 (after normal working hours when there are fewer requests for listings), specify the date and time as:

```
$ PRINT/AFTER=11-SEP:05:30PM TIBET.MEM
```

If the specified date or time has already passed, the file is queued for printing as soon as possible.

/FORMS=form-name
Specifies the name of the form required for the specified file(s). You specify the forms type with an alphanumeric name defined by the system manager. Check with the system manager to learn which form names you can specify.

/HOLD
Specifies that the entry be placed in a HOLD state on the queue. Entries in a HOLD state are not processed until you release them with the SET ENTRY/RELEASE command.

/JOB_COUNT=n
Requests that the entire job be printed n times, from 1 to 255. By default, the job is printed once.

The following example prints four copies of a file named AFGHAN.TXT:

```
$ PRINT/JOB_COUNT=4 AFGHAN.TXT
```

/JOB_COUNT differs from the /COPIES qualifier in the order in which multiple files are printed. For example, if you request three copies of files HOGAN.TOM and MARCH.ANN, the /JOB_COUNT qualifier prints one copy of HOGAN.TOM, followed by one copy of MARCH.ANN, and repeats the grouping until three copies of each file are printed:

```
$ PRINT/JOB_COUNT=3 HOGAN.TOM,MARCH.ANN
```

/NAME=entry-name
Specifies the 1 through 9 character name assigned to the entry. You can display the entry-name with the SHOW ENTRY command. In addition, PBS prints the entry-name on the first page (also known as the flag, or banner, page) of the hard copy.

/OWNER=[ppn]
Indicates the owner of the print job. PBS displays the project-programmer number on the job header page. You can only specify jobs with your own PPN (the default), unless you have sufficient privilege to specify others.

/PAGE_LIMIT=n
/PAGE_LIMIT=UNLIMITED
Specifies the maximum number of pages to be printed in the job. You can specify only up to the maximum page limit assigned to the queue by the system manager.

You can specify /PAGE_LIMIT=UNLIMITED only if the queue's maximum page limit is also set to UNLIMITED.

/PRIORITY=n
Specifies the priority of the print job, in the range 1 through 255. The PRINT command processes entries in priority order: higher-priority entries before lower-priority entries.

You cannot specify a priority higher than the maximum priority assigned to the queue by the system manager.

/QUEUE=queue-name[:]
Specifies the name of the queue on which the entry is placed. The queue you specify must be a print queue.

If you do not specify a queue-name in the entry-spec or if you do not specify /QUEUE, PBS places the entry on the default print queue.

Use the SHOW ENTRY command to verify an entry's position in the queue.

File Qualifiers

/CONVERT
/NOCONVERT
Specifies whether all zeros should be converted to the letter O before printing. The default is /NOCONVERT.

/COPIES=n
Specifies the number of copies to print. By default, the PRINT command prints one copy of a file; you can use /COPIES to request up to 255 copies. For example, the following command line requests three copies of file INDIA.TXT:

```
$ PRINT/COPIES=3 INDIA.TXT
```

If you specify /COPIES after the PRINT command name, each file is printed the number of times you specify. For example, to print three copies each of BOMBAY.TXT and LAHORE.MEM, type:

```
$ PRINT/COPIES=3 BOMBAY.TXT,LAHORE.MEM
```

If you specify /COPIES after a file specification, PBS prints only that file the specified number of times. The following command line requests two copies of NEPAL.DAT, one copy of SIKKIM.DAT (by default), and five copies of BHUTAN.DAT:

```
$ PRINT NEPAL.DAT/COPIES=2,SIKKIM.DAT,BHUTAN.DAT/COPIES=5
```

The /COPIES qualifier differs from /JOB_COUNT in the order in which multiple files are printed. For example, if you request three copies of files TEDFOR.BOB and MOODY.JON, PBS prints three copies of TEDFOR.BOB, followed by three copies of MOODY.JON.

```
$ PRINT/COPIES=3 TEDFOR.BOB,MOODY.JON
```

/DELETE
/NODELETE

Controls whether files are deleted after printing. /NODELETE is the default.

If you specify /DELETE after the PRINT command name, all files specified are deleted after printing. For example:

```
$ PRINT/DELETE OUTPUT.DAT,RESULT.TXT
```

Both OUTPUT.DAT and RESULT.TXT are deleted after printing.

If you specify /DELETE after a file-spec, only that file is deleted after printing. For example:

```
$ PRINT CONCRD.MAS,LOWELL.MAS/DELETE,BOXBRO.MAS
```

Only the file LOWELL.MAS is deleted after printing.

You must have write access to the file to delete it after printing,

/FEED
/NOFEED

Specifies whether the printer should leave six blank lines at the end of each page. The default is /FEED.

/FLAG_PAGES
/NOFLAG_PAGES

Specifies whether flag (file header) pages should be printed at the beginning of each file listing. If you specify /FLAG_PAGES, the number of flag pages printed is determined by the system manager, and may vary according to the form you specify with the /FORMS qualifier.

/TRUNCATE
/NOTRUNCATE

Indicates whether lines that exceed the width of the page should be truncated. The system manager determines the page width; it may vary according to the form you specify with the /FORMS qualifier.

The default is /NOTRUNCATE, which means that lines exceeding the form width continue, or "wrap," onto the next line.

Submitting Entries for Batch Processing: SUBMIT

The SUBMIT command enters a request for batch processing, either on the default batch queue or on a queue you specify.

When you use the SUBMIT command, you specify one or more command procedures to be executed by the batch server. See the *RSTS/E Guide to Writing Command Procedures* for detailed information about writing and executing command procedures.

Format

SUBMIT file-spec[,...] [entry-spec]

Command Qualifiers	**Defaults**
/AFTER=date:time	
/CPU_LIMIT=n	See discussion
/HOLD	
/[NO]LOG_DELETE	/NOLOG_DELETE
/[NO]LOG_FILE=file-spec	See discussion
/[NO]LOG_QUEUE=queue-name	/NOLOG_QUEUE
/NAME=entry-name	See discussion
/OWNER=ppn	See discussion
/PARAMETERS=(parameter,...)	
/PRIORITY=n	See discussion
/QUEUE=queue-name	See discussion
/TIME_LIMIT=n	See discussion

File Qualifiers	**Defaults**
/[NO]DELETE	/NODELETE

Prompts

File: file-spec[,...]

Command Parameters

file-spec

Specifies one or more command files (procedures) to be submitted for batch job execution. You must specify a file name. If you do not specify a file type, PBS assumes a default file type of .COM, and if you do not specify a device, PBS assumes the default device _SY:. Wildcards are allowed.

You can only submit files to which you have read access. For each file-spec in the SUBMIT command, PBS attempts to find at least one file to which you have read access; if it finds none, PBS displays an error message and refuses the batch request. If you specify the /DELETE qualifier with a file-spec, you must have both read and write access.

entry-spec
queue-name:[ppn]entry-name

Specifies the queue and the name of the entry. The entry-name can contain from one to nine characters. If you do not specify a queue, PBS submits the entry on the default batch queue. If you do not specify an entry-name, PBS assigns the first non-numeric file name in the list of file-specs, or BATCH if none is found. Your own PPN is the default.

Command Qualifiers

/AFTER = date:time
Specifies a date:time value after which the entry is eligible for batch processing. If you do not specify this qualifier, PBS assumes the entry is immediately eligible.

If you specify only a time value, PBS assumes today's date. If you specify only a date value, PBS assumes the end of the day (11:59 p.m.).

/CPU_LIMIT = n
/CPU_LIMIT = UNLIMITED
Specifies the maximum CPU time, in minutes, allowed for your batch job. This limit applies to the entire batch job, not to each command procedure within the job. You can only specify the maximum CPU limit of the batch request's queue.

If you specify /CPU_LIMIT = UNLIMITED, the batch request's queue must also have its maximum CPU limit set to UNLIMITED. A batch request with CPU_LIMIT = UNLIMITED is allowed an infinite amount of CPU usage during processing.

If you do not specify this qualifier, PBS assigns the batch queue's default CPU limit.

/HOLD
Specifies that the entry be placed in a HOLD state on the queue. Entries in a HOLD state are not processed until you release them with the SET ENTRY/RELEASE command.

/LOG_DELETE
/NOLOG_DELETE
Specifies whether the log file should be deleted after printing. The default is /NOLOG_DELETE.

/LOG_FILE [= file-spec]
/NOLOG_FILE
Specifies whether a log file is created for the batch job and if so, the file-spec of the log. By default, PBS creates a log file at the start of a batch job.

If you do not specify a file-spec, PBS creates the log file in your own account on the default device SY:, using the first six characters of the entry name and a file type of .LOG. (Any file with the same name will be replaced.)

If you specify a file-spec, PBS creates that file at the start of the batch job, replacing any file having the same name.

/LOG_QUEUE [=queue-name]
/NOLOG_QUEUE
Specifies whether a log file should be queued for printing when the batch job completes and if so, the queue on which it should be printed. The default is /NOLOG_QUEUE.

If you do not specify a queue name, PBS queues the log file on the default print queue.

/NAME=entry-name
Specifies the one- to nine-character name assigned to the entry.

/OWNER=ppn
Indicates the owner of the batch job. You can only specify your own PPN, unless you have sufficient privilege to specify others.

/PARAMETERS=(parameter[,...])
Specifies from one to eight optional parameters to be passed to the batch job. PBS assigns the parameters to the local symbols P1 through P8 when it executes the first command procedure in the job.

/PRIORITY=n
Specifies the priority of the batch request, in the range 1 to 255. PBS processes entries with higher priorities before those with lower priorities. You cannot specify a priority higher than the queue's maximum.

/QUEUE=queue-name[:]
Specifies the name of the queue on which the entry is placed. The queue specified must be a batch queue.

Use the SHOW ENTRY command to verify an entry's position in the queue.

/TIME_LIMIT=n
/TIME_LIMIT=UNLIMITED
Specifies the maximum elapsed time, in minutes, allowed for the requested batch job. This limit applies to the entire batch job, not to each command procedure within the job. You can only specify the maximum time limit of the batch request's queue.

If you specify /TIME_LIMIT=UNLIMITED, the batch request's queue must also have its maximum time limit set to UNLIMITED. A batch request with TIME_LIMIT=UNLIMITED is not limited to any amount of elapsed time during processing.

If you do not specify this qualifier, PBS assigns the batch queue's default time limit.

File Qualifiers

/DELETE
/NODELETE

Specifies whether the command file should be deleted after batch processing completes. You must have write access to the file for which you specify /DELETE. The default is /NODELETE.

You may type the /DELETE qualifier after any file name to delete only that file, or you may type the /DELETE qualifier immediately after the SUBMIT command to delete all files included in the command.

Displaying Print and Batch Queue Entries: SHOW ENTRY

The SHOW ENTRY command displays information about entries in the print or batch queue. This display shows each entry's current status as well as its characteristics, such as queue-name, entry-name, entry-number, owner, and priority.

Format

SHOW ENTRY [entry-spec or entry-number]

Command Qualifiers	Defaults
/ALL	
/BATCH	See discussion
/BRIEF	/BRIEF
/FILES	/BRIEF
/FULL	/BRIEF
/PRINT	See discussion

Command Parameters

entry-spec
entry-number

Specifies the entry-spec or entry-number of the entry you want to display. Use the entry-number when you want to display a single entry. Use the full entry-spec (queue:[ppn]name) to display all entries meeting the specification.

You can specify wildcard characters anywhere in the entry-spec. If you do not specify any entry parameter, the SHOW ENTRY command displays all print and batch entries owned by you.

Note

By default, PBS does not display entries by queue. To see a list of the entries in a particular queue, specify the queue-name in the entry-spec.

Command Qualifiers

/ALL

Specifies that all entries are to be displayed. Using the /ALL qualifier is the same as specifying the entry-spec *:[*,*]*, although you may include other qualifiers to restrict the entries selected.

If you do not specify /ALL, the entry-spec defaults to entries with your PPN.

/BATCH
Specifies that only batch queue entries be displayed. This qualifier conflicts with /PRINT.

The default is that both print and batch queue entries are displayed.

/BRIEF
Requests a brief listing of information about entries in the queue. It is the default. For example:

```
$ SHOW ENTRY/BRIEF (RET)

1st  Print entry 5   PRINT:[   1,214]MAIL   Status STARTED  Pri 128
```

The /BRIEF qualifier displays the following information about each entry:

- Position — the entry's start position relative to other entries
- Entry type — Print or Batch
- Entry number
- Queue name
- Owner's PPN
- Entry name
- Entry status — one of the following:
  - READY — The entry is ready for processing.
  - AFTER — The entry is waiting for an /AFTER date and time to be met before it can start processing.
  - HOLD — The entry is waiting for a SET ENTRY/RELEASE command in order to start processing.
  - STARTED — The entry has started processing. (Use the /FULL qualifier to display the date and time that the entry started.)
  - ABORTING — The entry is in the process of being terminated.
- Priority in the queue

/FILES
Displays all file-specs, and their qualifiers, included in the entry. Unless you have additional privilege, you can only see the file-specs contained in your own entries.

/FULL

Displays the full status of each entry. In addition to all the information shown by the /BRIEF and /FILES qualifiers, the /FULL qualifier also displays the following:

- Priority
- JOB_COUNT (number of copies to be printed)
- Date and time the entry was submitted
- Date and time the entry was last modified (if ever)
- Date and time the entry started processing
- AFTER date and time (specified by the /AFTER qualifier)
- Server name (if the job is STARTED)

For example:

```
2nd  Print entry 16   PRINT:[   1,214]BETH   Status READY   Pri 128
  Job Count 1, Form NORMAL
  Entered on 08-Jun-85 at 03:10 PM
     1>_SY:[1,214]BETH.DOC
```

Modifying a Print or Batch Queue Entry: SET ENTRY

The SET ENTRY command modifies one or more attributes of an entry. You also use SET ENTRY to hold or release an entry for processing.

Format

SET ENTRY entry-spec or entry-number

Command Qualifiers	Defaults
/[NO]AFTER=date:time	
/ALL	See discussion
/BATCH	See discussion
/CPU_LIMIT=n	
/FORMS=form-name	
/HOLD	
/JOB_COUNT=n	
/PAGE_LIMIT=n	
/PRINT	See discussion
/PRIORITY=n	
/RELEASE	
/TIME_LIMIT=n	

Prompts

Entry: entry-spec or entry-number

Command Parameters

entry-spec
entry-number

Specifies either the entry-spec or entry-number of the print or batch entry to be modified. If you specify the entry-spec, PBS modifies all entries matching the specification. If you specify the entry-number, PBS modifies only the entry with the matching number. Wildcards are allowed.

Command Qualifiers

/AFTER=date:time
/NOAFTER

Specifies a date:time value after which the entry is eligible for printing or batch processing.

If you specify only a time value, PBS assumes today's date. If you specify only a date value, PBS assumes the end of the day (11:59 p.m.).

For example, you type the following command line to have a job named BOCAT printed after March 2 at 1:00 in the afternoon:

```
$ SET ENTRY BOCAT/AFTER=02-JUN-85:01:00PM
```

The /NOAFTER qualifier releases an entry (previously set with the /AFTER qualifier) for immediate processing.

/ALL

Specifies that all of your entries are to be modified. Using the /ALL qualifier is the same as specifying the entry-spec *:[*,*]*, although you may include other qualifiers to restrict the entries selected.

/BATCH

Specifies that only batch queue entries be modified.

By default, both print and batch queue entries are selected to be changed.

/CPU_LIMIT=n
/CPU_LIMIT=UNLIMITED

(Batch entries only.) Specifies the maximum CPU time, in minutes, allowed for the batch job. This limit applies to the entire batch job, not to each command procedure within the job.

If you specify /CPU_LIMIT=UNLIMITED, the batch request's queue must also have its maximum CPU limit set to UNLIMITED. A batch request with CPU_LIMIT=UNLIMITED is allowed an infinite amount of CPU usage during processing.

/FORMS=form-name

(Print entries only.) Specifies the name of the form required by the entry. See your system manager to find out which form names you can specify.

/HOLD

Specifies that the entry be placed in a HOLD state on the print or batch queue. PBS does not process entries in a HOLD state until you release them with the SET ENTRY/RELEASE command.

/JOB_COUNT=n

(Print entries only.) Specifies that the entire job be printed n times, from 1 to 255.

/PAGE_LIMIT=n
/PAGE_LIMIT=UNLIMITED

(Print entries only.) Specifies the maximum number of pages to be printed in the job. You can specify only up to the maximum page limit assigned to the queue by the system manager, unless you have sufficient privilege to specify a higher limit.

You can specify /PAGE_LIMIT=UNLIMITED only if the queue's maximum page limit is also set to UNLIMITED.

/PRINT
Specifies that only print queue entries be modified.

The default is that both print and batch queue entries are selected to be changed.

/PRIORITY=n
Specifies the priority of the print or batch request, in the range 1 to 255. PBS processes entries with higher priorities before those with lower priorities. You cannot specify a priority higher than the queue's maximum.

/RELEASE
Specifies that an entry previously placed in a HOLD state be released.

/TIME_LIMIT=n
/TIME_LIMIT=UNLIMITED
(Batch entries only.) Specifies the maximum elapsed time, in minutes, allowed for the requested batch job. This limit applies to the entire batch job, not to each command procedure within the job.

If you specify /TIME_LIMIT=UNLIMITED, the batch request's queue must also have its maximum time limit set to UNLIMITED. A batch request with TIME_LIMIT=UNLIMITED is not limited to any amount of elapsed time during processing.

Deleting a Print or Batch Entry from a Queue: DELETE/ENTRY

The DELETE/ENTRY command uses the name or number of an entry to cancel a request to the print or batch queue.

Format

DELETE/ENTRY entry-spec or entry-number

Command Qualifiers

/ALL
/BATCH
/PRINT

Prompts

Entry: entry-spec or entry-number

Command Parameters

entry-spec
entry-number

Specifies either the entry-spec or entry-number of the print or batch entry to be deleted. If you specify the entry-spec, PBS deletes all entries matching the specification. If you specify the entry-number, PBS deletes only the entry with the matching number. Wildcards are allowed.

Note

Be careful when you delete jobs using the entry-spec parameter. More than one entry with the same entry-spec may exist.

Command Qualifiers

/ALL

Specifies that all of your entries are to be deleted. Using the /ALL qualifier is the same as specifying the entry-spec *:[*,*]*, although you may include other qualifiers to restrict the entries selected.

/BATCH

Specifies that only batch queue entries be deleted.

The default is that both print and batch queue entries are selected.

/PRINT

Specifies that only print queue entries be deleted.

The default is that both print and batch queue entries are selected.

Displaying a Queue's Characteristics: SHOW QUEUE

The SHOW QUEUE command displays the status of one or more print or batch queues. The display includes information about the queue's characteristics, such as default form name, size limits, and the servers assigned to the queue.

Format

SHOW QUEUE queue-name[:]

Command Qualifiers	**Default**
/ALL	
/BATCH	See discussion
/BRIEF	/BRIEF
/FULL	/BRIEF
/PRINT	See discussion

Prompts

Queue: queue-name[:]

Command Parameters

queue-name
Specifies the name of the queue to be displayed. Wildcards are allowed.

Command Qualifiers

/ALL
Specifies that all queues be displayed.

/BATCH
Specifies that only batch queues be displayed.

The default is to display all queues matching the queue-name parameter, regardless of type.

/BRIEF
Specifies a brief display (the default), which includes the following information about the specified queue:

- Queue type - Print or Batch
- Queue name

- Queue status - one or more of the following:
  - DEFAULT — Queue is the default queue.
  - OPEN — Queue can accept requests.
  - CLOSED — Queue will not accept requests.
  - STARTED — Entries on the queue can be started.
  - STOPPED — Entries on the queue cannot be started.
  - MARKED FOR DELETION — Queue will be deleted as soon as it becomes empty.

/FULL

Specifies a full display, which includes all the information in the brief (default) display, along with the following additional information:

- Default form name (Print queues only) — The name assigned to the default paper form in the printer.
- Default/maximum priority — The default priority assigned to entries in the queue and the maximum priority that can be specified for entries.
- Default/maximum page limit (Print queues only) — The default page limit assigned to entries in the queue and the maximum page limit that can be specified for entries.
- Default/maximum CPU limit (Batch queues only) — The default CPU limit assigned to entries in the queue and the maximum CPU limit that can be specified for entries.
- Default/maximum time limit (Batch queues only) — The default time limit assigned to entries in the queue and the maximum time limit that can be specified for entries.
- Assigned servers — The list of print and batch servers currently assigned to the queue.

/PRINT

Specifies that only print queues be displayed.

The default is to display all queues matching the queue-name parameter, regardless of type.

Program Development 8

This chapter describes DCL commands for developing, linking, and running programs on RSTS/E.

Note

This manual does not explain programming concepts. To use this chapter, you need to know how to write programs in the language you select. See the appropriate programming manuals for information about using each language.

Table 8-1 describes the commands you use to develop programs on RSTS/E.

Table 8-1: Program Development Commands

Command	Function
	Programming Languages and Environment
BASIC	Invokes the BASIC-PLUS or BASIC-PLUS-2 programming environment, depending on the qualifiers you use or the default set by your system manager.
COBOL	Compiles COBOL-81 programs.
DIBOL	Compiles a DIBOL-11 program.
FORTRAN	Compiles a FORTRAN program. Two compilers are available: FORTRAN-IV and FORTRAN-77; depending on the qualifiers you use or the default set by your system manager.
MACRO	Assembles a program written in MACRO-11.
	Programming Operations
LINK	Links together object files to produce an executable file.
RUN	Executes a system or user program.

Developing Programs on RSTS/E

You can develop programs on RSTS/E using a variety of techniques and languages. BASIC-PLUS (supplied with all RSTS/E systems) is highly interactive, and provides a self-contained programming environment.

BASIC-PLUS is an interpreted language. As you enter lines of source code into the system, BASIC-PLUS immediately translates them into code that the system can execute. Thus, you can run a program right after you finish entering it.

Other high-level languages (such as BASIC-PLUS-2, COBOL, DIBOL, and FORTRAN) are compiled languages, generating PDP-11 object code. To develop programs in these languages, you must create a source program, then compile it and link it before it can be run.

In addition, the MACRO-11 assembler (supplied with all RSTS/E systems) lets you write programs that correspond line-for-line with PDP-11 machine instructions. You follow the same general steps to develop MACRO programs as you do for compiled languages.

BASIC-PLUS and BASIC-PLUS-2 are command environments on RSTS/E. This means that you can switch from DCL command level to a BASIC environment, which has its own prompt for accepting your input. (The BASIC-PLUS prompt is "Ready." The BASIC-PLUS-2 prompt is usually "BASIC2," depending on what your system manager has specified.) By contrast, you develop programs in some other languages by using DCL commands.

The number and type of programming tools available on any given RSTS/E system (as well as the language you choose for any given application) depend on many factors. These factors include: the size of the system; its cost; and the application environment.

The following is an overview of the program development cycle for compiled languages — that is, those RSTS/E languages other than BASIC-PLUS. For information about programming in a specific language (including sample programs and procedures), see the appropriate language manual.

Overview

RSTS/E offers a number of different languages for program development. Although each language is unique, you use a similar process when developing any program that you compile.

Briefly, the major steps are:

1. Design — Planning and initial coding.
2. Edit — Entering code into a source file.
3. Compile — Compiling or assembling source code into an object file.

4. Link — Combining object files into an executable file.
5. Test — Making trial runs of an executable file and finding errors.

The next four sections introduce the tools you use for the edit, compile, link, and test steps. The commands you use in each step depend on the programming language you select.

Editing

A source program is a file containing text; this is the form of your program that you actually write. The text in a source program consists of statements in BASIC, COBOL, DIBOL, FORTRAN, MACRO, or some other high-level language.

You generally use an editor such as EDT (which you invoke with the EDIT command) to create a file containing the text of your source statements. An editor allows you to type lines of the program and save it as a file. When you have entered the program lines into the file the way you want them, you are ready for the compile step.

Compiling

There is a command for compiling each high-level language. The compiler:

1. Reads the source-code file you have created with your editor
2. Compiles it into PDP-11 machine code
3. Writes the compiled machine code into an object file

On PDP-11 systems, the object output is called a "module"; it contains your program in the binary language that, when linked, is executable by a PDP-11 computer.

A single line of source code, especially in a high-level language, may be compiled into many PDP-11 instructions. The efficiency (and improved clarity) of not having to write machine instructions a line at a time is one of the important reasons for using a high-level language.

MACRO is called an "assembler." For this discussion, its function is the same as that of any compiler. MACRO turns source code into object code, and it fits into the program development cycle the same way as do the compilers.

In addition, the MACRO assembler can produce a listing that shows both your source code and the generated object code. This listing, like other listings prepared by compilers, is helpful during the debugging process.

Another listing that is useful in debugging is the "cross-reference" listing. This listing shows where each symbol is referenced in your source program. For the MACRO assembler, you can request this cross-reference as part of the assembler's program listing. For some of the other languages, there are separate programs that operate independently of the compilers to produce cross-reference listings. For example, BASIC-PLUS has BPCREF; BASIC-PLUS-2 has B2CREF.

Linking

A program that is ready for execution by the system is called an executable file. (Other terms for executable file are save image, task, or load module.)

To prepare an executable file from your object module, you must link it. Linking a program into an executable file performs three major functions:

- Combination of modules — You can link several separately compiled object files together into a single executable file.
- Relocation of addresses — Because the assignment of actual memory locations for any object module must depend on the other object modules which go into the load module with it, the link step is the time when references to absolute memory locations are fixed.
- Other object modules — You can incorporate modules that the system supplies into your executable file. Each high-level language requires certain object modules to be linked into executable files for programs written in that language.

Combining Modules

There are two main reasons why you want to be able to link an executable file from separately compiled object modules:

- If your program is complex, it is helpful to be able to compile it in several smaller pieces as opposed to one large program. During debugging, you can then recompile only the smaller piece that you have changed before linking a new executable file for further testing.
- You can link subroutines into a main program. You do so by maintaining a library of subroutines from which the linker can extract object modules. Therefore, you can reduce repetitious source coding in your main program.

The use of subroutines also allows you to write the main portion of your program in one language, and other portions in another. For example, when writing a main program in FORTRAN, you might also want to code a special or performance-sensitive subroutine in MACRO.

Relocating Addresses

The final location of an object module in a linked executable file cannot be determined until the object modules are combined during linking. Therefore, object modules are relocatable. In addition to binary machine instructions, an object module contains tables of information that direct link-time operations, such as address relocation.

Note that the linker can optionally produce a listing of resolved addresses, called a map.

Testing

You can rarely create a program that does not contain at least one error, either in its logic or in its coding. You may discover errors while you are editing the program. Furthermore, the compiler may find errors during the compilation process. The compiler informs you of any errors it finds by using error flags in the listings. The linking process may also catch certain errors and issue appropriate messages.

However, often you do not discover that the program does not work properly until you run your program. Some programming errors are difficult to find. For this reason, special debugging tools are supplied with most high-level languages. See the appropriate programming language manual for information about available debugging tools.

The RT11 and RSX Tools

Most of the high-level languages available on RSTS/E are very similar to corresponding languages on RT-11 or RSX-11. Therefore, the program development commands you use on RSTS/E are similar to the RT11 and RSX program development commands.

For example, when you develop a FORTRAN-IV program on RSTS/E, you use commands similar to those on RT11. When you develop a program in other high-level languages (including FORTRAN-77, but excluding BASIC-PLUS), you use commands similar to those on RSX. You can develop MACRO-11 programs using either set of program development commands.

The description of the program development cycle stated earlier applies to the commands for either RT11 or RSX. However, the appropriate command for each programming environment is unique: there is one set of commands for RT11, and another set for RSX.

Table 8-2 shows the two sets of commands.

Table 8-2: Program Development Commands on RSTS/E

Generic Name	RT11 Command	RSX Command
COBOL	–	COBOL
DIBOL	–	DIBOL
FORTRAN	FORTRAN/FOR	FORTRAN/F77
MACRO	MACRO/RT11	MACRO/RSX11
Linker	LINK/FOR LINK/RT11	LINK/F77 LINK/C81 LINK/DIBOL LINK/RSX11 LINK/BP2

The file types assigned to executable files are .SAV for the RT11-based linker and .TSK for the RSC-based linker. The file type of object files is .OBJ in either environment.

RT11-Based Programming

You can use RSTS/E's RT11-based commands to develop programs in either FORTRAN-IV or MACRO. Depending on your application, the programs you develop can be either:

- Run directly on your RSTS/E system
- Debugged on RSTS/E, compiled, and run on an RT-11 system

Using MACRO, you can write programs that use any feature of RSTS/E, or programs that use only RT11 monitor directives.

RSX-Based Programming

You can use RSTS/E's RSX-based commands to develop programs in MACRO or any of the high-level languages. Depending on your application, your developed programs can be either:

- Run directly on your RSTS/E system
- Debugged on RSTS/E, compiled, and run on an RSX-11 system

Using MACRO, you can write programs that use any feature of RSTS/E, or programs that use only RSX monitor directives. Certain features of RSTS/E are also available directly through some high-level languages (for example, the BASIC-PLUS SYS calls, which are also available in BASIC-PLUS-2).

If you want to use the RMS data management package in your application, you will be using RSTS/E's RSX-based languages. (See RMS documentation for more information.)

BASIC

The BASIC command places you in the BASIC-PLUS or BASIC-PLUS-2 programming environment, depending on the qualifiers you use and the system's default.

Format	
BASIC	
Command Qualifiers	**Defaults**
/BP2 /BPLUS	/BP2

These commands place you in the BASIC-PLUS or BASIC-PLUS-2 programming environment. In BASIC-PLUS, you can use BASIC-PLUS keyboard monitor commands (for program development), or CCL commands. In BASIC-PLUS-2, you can use program development commands.

To return to DCL from the BASIC-PLUS or BASIC-PLUS-2 environment, type:

```
$
```

or

```
$SWITCH
```

See Appendix B, or the appropriate language manual, for details on the commands you can use in these environments.

BASIC/BP2 invokes the BASIC-PLUS-2 environment to begin a programming session:

```
$ BASIC/BP2 (RET)
```

The system displays the following prompt for the BASIC-PLUS-2 environment:

```
BASIC2
```

BASIC/BPLUS invokes the BASIC-PLUS environment:

```
$ BASIC/BPLUS (RET)
```

The system displays the following prompt for the BASIC-PLUS environment:

```
Ready
```

Command Qualifiers

/BP2
Invokes the BASIC-PLUS-2 programming environment.

/BPLUS
Invokes the BASIC-PLUS programming environment.

COBOL

The COBOL command compiles a COBOL-81 program. (You can compile only one source file at a time with COBOL-81.)

Format

COBOL [source_file]

Command Qualifiers	Defaults
/[NO]ANSI_FORMAT	/NOANSI_FORMAT
/C81	/C81
/[NO]CHECK	/CHECK
/CHECK=[NO]BOUNDS	
/CHECK=[NO]PERFORM	
/CODE=[NO]CIS	See description
/[NO]CROSS_REFERENCE	/NOCROSS_REFERENCE
/[NO]DEBUG	/NODEBUG
/DIAGNOSTICS[=diagfile]	/NODIAGNOSTICS
/NODIAGNOSTICS	
/LIST[=listfile]	/NOLIST
/NOLIST	
/[NO]MAP	/NOMAP
/NAMES=aa	/NAMES=SC
/OBJECT[=objfile]	/OBJECT[=objfile]
/NOOBJECT	
/[NO]SHOW	/NOSHOW
/SHOW=[NO]MAP	
/[NO]SUBPROGRAM	/NOSUBPROGRAM
/TEMPORARY=device	/TEMP=SY:
/[NO]TRUNCATE	/NOTRUNCATE
/[NO]WARNINGS	/WARNINGS
/WARNINGS=[NO]INFORMATIONAL	

Prompts

File: source_file

Command Parameters

source_file

Specifies a COBOL program to be compiled. You must specify a source file name. If you do not include a file type, the compiler uses the default type of .CBL.

No wildcards are allowed for the source file.

Command Qualifiers

/ANSI_FORMAT
/NOANSI_FORMAT
Indicates whether the source program is in ANSI COBOL format or in DIGITAL's terminal format.

An ANSI COBOL source file has 80-character records with Area A beginning in column position 8.

The recommended default is for the COBOL command to assume that the input records are in terminal format (Area A begins in column position 1 and the lines do not have line numbers). However, your system manager can change the default at system installation.

/C81
Specifies the COBOL-81 compiler. This is the default.

/CHECK
/NOCHECK
/CHECK=[NO]BOUNDS
/CHECK=[NO]PERFORM
Controls whether the compiler produces extra code to check for program correctness at run time. The /CHECK qualifier verifies subscripts, indexes, and perform statement ranges. The /CHECK qualifier is the recommended default and causes COBOL-81 to check each subscript and table at run time against the ranges defined by its data-name's OCCURS clause. (Your system manager can change this default at system installation.) The /CHECK qualifier also causes COBOL-81 to determine if your program's PERFORM ranges are nested properly (if nested at all). If COBOL-81 detects improper nesting during execution, it issues an error message to that effect.

The /NOCHECK qualifier suppresses all checking, which can save execution time and decrease task image size. If you use /NOCHECK to suppress error checking, an out-of-range subscript or index does not generate an error message. The /NOCHECK qualifier also stops the compiler from generating code needed for checking PERFORM statement ranges. If you use /NOCHECK and the program's PERFORM statements are nested incorrectly, the program does not produce valid results.

You can have the compiler generate code to check only subscripts and indexes by specifying /CHECK=(BOUNDS, NOPERFORM). Likewise, you can check only PERFORM statement ranges by specifying /CHECK=(PERFORM, NOBOUNDS).

/CODE=CIS
/CODE=NOCIS

Controls whether the compiler uses Commercial Instruction Set (CIS) in the object code it produces. CIS speeds program execution; however, not all processors support CIS.

If your processor has CIS, the recommended default is /CODE=CIS; however, your system manager can change the default at system installation. If your processor does not have CIS, the default is /CODE=NOCIS.

/CROSS_REFERENCE
/NOCROSS_REFERENCE

Controls whether the compiler listing includes a cross-reference listing. By default, the compiler does not create a cross-reference listing. In addition, you receive the error message ?Additional qualifier needed, unless you specify the /LIST qualifier.

/DEBUG
/NODEBUG

Causes the compiler to use the COBOL-81 Symbolic Debugger. (The debugger lets you control and monitor your program as it runs by referring to the source version rather than the object code produced by the compiler.) The compiler generates symbol information in the object module for all data-names and procedure-names. This increases the size of the object file, but when you finish debugging and no longer need the symbols, you can recompile without using this qualifier.

The recommended default is /NODEBUG; however, the system manager can change the default at system installation.

If you use the Symbolic Debugger in your program, you must also use the LINK/DEBUG command to include the Debugger in your task image. See the LINK command for more information.

/DIAGNOSTICS[=diagfile]
/NODIAGNOSTICS

Controls whether the compiler produces an output diagnostic file.

The diagnostic file is a subset of the list file. It contains only the diagnostics issued, along with the line of source code to which each diagnostic applies.

By default, the COBOL-81 compiler does not create a diagnostics file. If you specify /DIAGNOSTICS without a file specification, the compiler creates a diagnostic file with the same name as the object file, and in the same directory on the same device as the object file, but with a .DIA file type. If you also specify /NOOBJECT, then the file has the same name as the source file, and is in your directory on SY:.

/LIST[=listfile]
/NOLIST
Controls whether the compiler produces an output listing.

The COBOL-81 compiler, by default, does not create a listing file. If you specify /LIST without a file specification, the compiler creates a listing with the same name as the object file, and in the same directory on the same device as the object file, but with an .LST file type. If you also specify /NOOBJECT, then the file has the same name as the source file, and is in your directory on SY:.

If you include a file specification, the listing is written to that file or device. The default file type is .LST.

/MAP
/NOMAP
The /MAP qualifier is the same as /SHOW=MAP. The /NOMAP qualifier is the same as /SHOW=NOMAP. This qualifier exists for compatibility with previous releases of RSTS/E.

/NAMES=aa
Requests the compiler to generate program section names ending with the two-character alphanumeric suffix, aa. If you link several object files into one executable file then each object file must have a different suffix for its program section names.

If you do not specify the /NAMES qualifier, RSTS/E uses the default suffix SC.

/OBJECT[=objfile]
/NOOBJECT
Controls whether the compiler produces an object file and a skeleton file. These files are used as input to the LINK command, which in turn produces an executable program.

By default, (or if you specify /OBJECT without a file specification), the compiler produces an object file with the same file name as the input file in your directory on the public disk structure, with a file type of .OBJ. The compiler also uses the default file type of .OBJ when you include a file name but no file type with the /OBJECT qualifier. The skeleton file is in the same directory and has the same name as the object file, but has an .SKL file type.

You cannot use wildcard characters in the file specification.

/SHOW
/NOSHOW
/SHOW=MAP
/SHOW=NOMAP
Determines whether the compiler produces a Data Division map showing the memory addresses for Data Division entries. The /SHOW and /SHOW=MAP qualifiers perform the same operations. The /NOSHOW and /SHOW=NOMAP

qualifiers also produce the same results and are the recommended defaults. However, your system manager can change the default at system installation.

You can use this qualifier only in conjunction with /LIST.

/SUBPROGRAM
/NOSUBPROGRAM

Indicates the compiler is compiling a subprogram. Use this qualifier only if the subprogram has not passed parameters from the calling program — that is, if the subprogram does not contain the PROCEDURE DIVISION USING header. The default is /NOSUBPROGRAM.

/TEMPORARY = device

Changes the storage area for temporary work files from SY: (the default) to the value you specify for "dev" (device). This qualifier is useful if you are running out of space on the system disk, or if you have a faster disk available. Because of the way the compiler names its temporary work files, two simultaneous compilations of source programs with the same file name, in the same account, (regardless of the types used) produces unpredictable results. This problem will not occur, however, if you use the /TEMPORARY = dev qualifier to specify a nondefault device for those files.

/TRUNCATE
/NOTRUNCATE

Specifies that the compiler perform decimal truncation on the values of COMP data items.

The recommended default is for COBOL-81 to perform binary truncation rather than decimal truncation. (Your system manager can change the default at system installation.) With binary truncation, the maximum value a COMP item can contain depends on its storage allocation. If you specify /TRUNCATE, the maximum value depends on the item's PICTURE character-string. See the *COBOL-81 RSTS/E User's Guide* for more information.

/[NO]WARNINGS
/WARNINGS = INFORMATIONAL
/WARNINGS = NOINFORMATIONAL

The /NOWARNINGS and /WARNINGS = NOINFORMATIONAL qualifiers stop the compiler from issuing informational diagnostics during the compilation. If you specify either of these qualifiers, only warning and fatal diagnostics will appear in the list file, diagnostics file, and diagnostic summary.

The /WARNINGS and /WARNINGS = INFORMATIONAL qualifiers, the recommended defaults, specify that the compiler issue informational diagnostics during the compilation.

DIBOL

The DIBOL command compiles a DIBOL-11 program. You can include up to six source file specifications to be compiled into a single object file with the DIBOL compiler. The default file type is .DBL.

Format

DIBOL filespec[,...]

File Qualifiers	Defaults
/[NO]DEBUG	/NODEBUG
/LIST[=listfile]	/NOLIST
/NOLIST	
/OBJECT[=objfile]	/OBJECT
/NOOBJECT	
/WARNINGS	/WARNINGS
/NOWARNINGS	

File Qualifiers

/DEBUG
/NODEBUG

Controls whether the compiler uses the DIBOL Debugging Technique (DDT).

If you specify /DEBUG, you must also use the /DEBUG qualifier when you link the program. If your program consists of a main program and one or more separately compiled subroutines, and you want to use the debugging tool on the entire program, you must compile each subroutine with the /DEBUG qualifier. You can mix sections of your program that have been compiled with /DEBUG with sections that have not been compiled with /DEBUG. However, you can use debugging tools only in the sections compiled with /DEBUG.

/LIST[=listfile]
/NOLIST

Controls whether or not the DIBOL compiler produces an output listing.

By default, the compiler does not create a listing file. If you specify /LIST without a file specification, then the compiler gives a listing file the same file name as the object file (or, if you specify /NOOBJECT, as the first input source file). The file is placed in the same directory on the same device as the object file, but with a .LST file type.

You cannot use wildcard characters in the file specification.

/OBJECT[=objfile]
/NOOBJECT

Controls whether the compiler produces an output object file.

By default, or if you specify /OBJECT without a file specification, the compiler produces an object module that has the same file name as the first input source file, and places it in your directory on the public disk structure with a file type of .OBJ.

You cannot use wildcard characters in the file specification.

/WARNINGS
/NOWARNINGS

Controls whether the compiler produces diagnostic messages for warning conditions.

Use /NOWARNINGS to override the compiler default. (The default is to issue warning diagnostic messages.)

FORTRAN

The FORTRAN command compiles up to six FORTRAN source files into a single object file.

There are two FORTRAN compilers available on RSTS/E:

Command	Invokes
FORTRAN/F77	FORTRAN-77
FORTRAN/FOR	FORTRAN-IV

FORTRAN/F77

The FORTRAN/F77 command invokes the FORTRAN-77 compiler.

Format

FORTRAN/F77 file-spec[,...]

Command Qualifiers	Defaults
/[NO]CHECK	/CHECK
/CONTINUATIONS=n	/CONTINUATIONS=19
/[NO]D_LINES	/NOD_LINES
/[NO]I4	/NOI4
/[NO]IDENTIFICATION	/NOIDENTIFICATION
/LIST[=listfile]	/NOLIST
/NOLIST	
/[NO]MACHINE_CODE	/NOMACHINE_CODE
/OBJECT[=objfile]	/OBJECT
/NOOBJECT	
/[NO]WARNINGS	/WARNINGS
/WORK_FILES=n	/WORK_FILES=2

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more FORTRAN source programs to be compiled. If you do not specify a file type, the compiler uses the default file type of .FTN.

You can specify more than one input file. The files are compiled as a single input file. The result is one object module and, if /LIST is specified, one listing.

You cannot use wildcard characters in the file specification.

Command Qualifiers

/CHECK
/NOCHECK
Controls whether the compiler produces extra code that checks for program correctness at run time. The /CHECK qualifier (the default) causes the production of code to check that all array references are to addresses within the address boundaries specified in the array declaration.

/CONTINUATIONS=n
Specifies the maximum number of continuation lines to be permitted.

The number of lines, n, is a decimal number from 0 through 99. By default, the compiler accepts a maximum of 19 continuation lines.

/D_LINES
/NOD_LINES
Indicates whether the compiler reads and compiles debugging lines, which have a D in column 1 of the source program. (Debugging lines check that the program is working; they are not needed after the program is debugged.) If you specify /D_LINES, lines that have a D in column 1 are compiled.

The default is /NO_DLINES, which means the compiler assumes that lines beginning with a D are comments and does not compile them.

/I4
/NOI4
Controls how the compiler interprets INTEGER and LOGICAL declarations that do not specify a length.

If you specify /I4, the compiler allocates a default length of two words (four bytes) for integer and logical variables.

If you specify /NOI4, the compiler interprets the declarations as INTEGER*2 and LOGICAL*2, respectively. /NOI4 is the default.

/IDENTIFICATION
/NOIDENTIFICATION
Controls whether the FORTRAN-77 compiler identification and version numbers are typed on your terminal. The default is /NOIDENTIFICATION.

/LIST[=listfile]
/NOLIST
Controls whether a listing file is produced.

By default, the compiler does not create a listing file. If you specify /LIST without a file specification, then the compiler gives a listing file the same file name as the object file, (or, if you specify /NOOBJECT, as the first input source file) and in the same directory on the same device as the object file, but with an .LST file type.

If you give a file specification, the listing is written to that file. The default file type is .LST.

You cannot use wildcard characters in the file specification.

/MACHINE_CODE
/NOMACHINE_CODE
Controls whether the listing produced by the compiler includes the machine language code that the compiler generated.

The default is /NOMACHINE_CODE, which omits machine language code in the listing. The /MACHINE_CODE qualifier is invalid unless you also specify the /LIST qualifier.

/OBJECT[=objfile]
/NOOBJECT
Controls whether the compiler produces an output object file.

By default (or if you specify /OBJECT without a file specification), the compiler produces an object module that has the same file name as the first input source file, in your directory on the public disk structure. The default file type .OBJ.

You cannot use wildcard characters in the file specification.

/WARNINGS
/NOWARNINGS
Controls whether the compiler produces diagnostic messages for warning conditions.

Use /NOWARNINGS to override the compiler default, which is to issue warning diagnostic messages.

/WORK_FILES=n
Determines the number of temporary disk work files that should be used during compilation. You can use from one to three files; the default value for n is 2.

If you increase the number of files, the size of the largest program to be compiled can be increased, but you may decrease compilation speed.

FORTRAN/FOR

The FORTRAN/FOR command invokes the FORTRAN-IV compiler.

Format

FORTRAN/FOR filespec[,...]

Qualifiers	Defaults
/CODE=EAE EIS FIS THR	
/[NO]D_LINES	/NOD_LINES
/[NO]I4	/NOI4
/[NO]LINENUMBERS	/LINENUMBERS
/LIST[=listfile]	/NOLIST
/NOLIST	/NOLIST
/[NO]MACHINE_CODE	/NOMACHINE_CODE
/OBJECT[=objfile]	/OBJECT
/NOOBJECT	
/[NO]OPTIMIZE	/OPTIMIZE
/[NO]WARNINGS	/WARNINGS

Command Parameters

filespec [,...]

Specifies one or more source files to be compiled into a single object file. The default file type is .FOR.

Qualifiers

/CODE=EAE
EIS
FIS
THR

The /CODE qualifiers select the type of object code to be generated.

The compiler produces distinctly different types of object programs by generating either threaded code or inline code. The default is determined by the system manager at system installation time.

The valid values are:

EAE — Selects EAE hardware
EIS — Selects EIS hardware
FIS — Selects EIS and FIS hardware
THR — Selects threaded code

When the compiler produces inline code (/CODE= followed by EAE, EIS, or FIS) the object program executes at greater speed and generally uses less physical memory. Inline code achieves this optimization, in part, by omitting instructions to detect or report certain error conditions. However, you can generate code for error checking by including the /CODE=THR option in the compiler command line.

When the compiler generates threaded code (through the /CODE=THR option), the object program produced uses a symbolic library routine to perform each operation required for program execution. The executable program consists of a "threaded" list of the addresses of library routines and appropriate operand addresses. This type of code generation produces an object module that operates independently of hardware arithmetic configuration. You can combine it with any of the FORTRAN-IV OTS libraries to produce a valid executable program for each type of arithmetic hardware without any need for recompilation.

/D_LINES
/NOD_LINES

Indicates whether the compiler reads and compiles debugging lines. (Debugging lines have a D in column 1 of the source program.) If you specify /D_LINES, lines that have a D in column 1 are compiled.

The default is /NOD_LINES, which means the compiler assumes that lines beginning with a D are comments and does not compile them.

/I4
/NOI4

Controls how the compiler interprets INTEGER and LOGICAL declarations that do not specify a length.

If you specify /I4, the compiler allocates a default length of two words (four bytes) for integer and logical variables.

By default (or if you specify /NOI4), the compiler interprets these as INTEGER*2 and LOGICAL*2, respectively.

/LINENUMBERS
/NOLINENUMBERS

The /LINENUMBERS qualifier indicates that internal sequence numbers are to be included in the executable program for routine diagnostics.

The /NOLINENUMBERS qualifier suppresses generation of internal sequence numbers.

/LIST=[listfile]
/NOLIST

Controls whether a listing file is produced.

The compiler does not create a listing file by default. If you specify /LIST without a file specification, then the compiler gives a listing file the same file name as the

object file. (If you specify /NOOBJECT, then the compiler gives the name of the first input source file.) The file also is in the same directory on the same device as the object file, but with an .LST file type.

If you give a file specification, the default file type is .LST. You cannot use wildcard characters in the file specification.

/MACHINE_CODE
/NOMACHINE_CODE
Controls whether the listing produced by the compiler includes the machine language code that the compiler generated.

The default is /NOMACHINE_CODE, which omits machine language code in the listing. The /MACHINE_CODE qualifier is invalid if /LIST is not specified.

/OBJECT[=objfile]
/NOOBJECT
Controls whether the compiler produces an output object file.

By default, or if you specify /OBJECT without a file specification, the compiler produces an object module that has the same file name as the first input source file, in your directory on the public disk structure, with a default file type of .OBJ.

If you give a file specification, the default file type is .OBJ. You cannot use wildcard characters in the file specification.

/OPTIMIZE
/NOOPTIMIZE
Tells the compiler whether to produce optimized code. The default is /OPTIMIZE.

Use the /NOOPTIMIZE qualifier during a debugging session to make sure that the debugger has sufficient information to help you locate errors in your source program.

/WARNINGS
/NOWARNINGS
Controls whether the compiler produces diagnostic messages for warning conditions.

Use /NOWARNINGS to override the compiler default. (The default is to issue warning diagnostic messages.)

MACRO

The MACRO command invokes a MACRO-11 assembler. You can include up to six file specifications with the MACRO command.

On RSTS/E you can use either MACRO/RT11 or MACRO/RSX11. The default is MACRO/RSX11, unless your system manager has changed it. The default file type is .MAC with one exception: .MLB is the default file type if you use the RSX-based assembler with the /LIBRARY qualifier.

Format

MACRO/RT11 filespec[,...]

OR

MACRO/RSX11 filespec[,...]

Command Qualifiers	**Defaults**
/CROSS_REFERENCE	
/LIST[=listfile]	/LIST[=listfile]
/NOLIST	
/OBJECT[=objfile]	/OBJECT[=objfile]
/NOOBJECT	

File Qualifiers

/LIBRARY

Command Parameters

filespec [,...]

Specifies one or more source files to be compiled into a single object file. The default file type is .MAC with one exception: .MLB is the default file type if you use the RSX-based assembler with the /LIBRARY qualifier.

Command Qualifiers

/CROSS_REFERENCE

Requests a cross-reference listing of symbols in the listing file.

/LIST[=listfile]
/NOLIST

Controls whether a listing file is produced. Unlike the other languages on RSTS/E, MACRO produces a listing file by default. (In other words, the /LIST qualifier is assumed.) The assembler gives a listing file the same file name as the object file and puts it in the same directory on the same device as the object file. If you specify /NOOBJECT, then it assumes the name of the first input source file, and is placed in your directory on the public structure. The default file type is .LST.

You cannot use wildcard characters in the file specification.

/OBJECT[=objfile]
/NOOBJECT

Controls whether the assembler produces an output object module.

By default (or if you specify /OBJECT without a file specification), the assembler produces an object module that has the same file name as the first input source file, on your directory on the public structure. The default file type is .OBJ.

You cannot use wildcard characters in the file specification.

File Qualifiers

/LIBRARY

Indicates that the file is a macro library.

When a macro call is found in the source program, MACRO-11 searches the user macro library for the named macro definitions. If necessary, MACRO-11 then continues the search with the system macro library.

LINK

The LINK command combines one or more of your compiled or assembled object files, including routines from appropriate libraries, into a single executable file.

Note

This section describes the DCL LINK command and its qualifiers. See the *RSTS/E Task Builder Reference Manual* or the *RSTS/E RT11 Utilities Manual* for a complete description of linking concepts, overlays, and libraries.

Format for Simple (Non-Overlaid) Link

LINK file-spec1[,file-spec2,...]

Format for Overlaid Link

LINK /STRUCTURE

.
.
.

(LINK prompts for overlay structure)

Command Qualifiers	Defaults
Language Qualifier	
/BASIC or /BP2 /COBOL or /C81 /DIBOL /F77 /FOR /RSX11 /RT11	/BP2 (unless your system manager has selected another)
Debugging Qualifier	
/[NO]DEBUG	/NODEBUG
Description Qualifier	
/DESCRIPTION	
Forms Management System Qualifier	
/[NO]FMS /NOFMS	

(continued on next page)

Library Qualifiers

/FMS = [NO]RESIDENT
/OTS = [NO]RESIDENT
/[NO]RMS
/RMS = [NO]RESIDENT

Output File Qualifiers

/[NO]EXECUTABLE[= filespec]	/EXECUTABLE
/[NO]MAP[= filespec]	/NOMAP

Overlay Qualifier

/STRUCTURE	(no overlay structure)

Prompt (No Overlays)

Files: file-spec1[,file-spec2,...]

Prompt (Overlays requested with /STRUCTURE qualifier)

ROOT files:	file-spec1[,file-spec2,...]
Root COMMON areas:	[common-area][,...]
Overlay:	[file-spec[,...][+]]

.
.
.

(more prompts for overlays)

Overview

Two linker programs are available on RSTS/E:

- The RSX-based linker (the Task Builder)
- The RT11-based linker (LINK.SAV)

The LINK command invokes one or the other, depending on your source language. (You indicate the source language by including a language qualifier.) BASIC-PLUS-2 is the default.

Table 8-3 shows the language qualifiers and whether the RSX-11-based or RT11-based linker is run in each case.

Table 8-3: Language Qualifier, Source Language, andLinker Relationship

Qualifier	Source Language	Linker
/BP2 /BASIC	BASIC-PLUS-2	RSX-11-based
/COBOL /C81	COBOL-81	RSX-11-based
/DIBOL	DIBOL	RSX-11-based
/F77	FORTRAN-77	RSX-11-based
/FOR	FORTRAN-IV	RT11-based
/RSX11	MACRO Assembler	RSX-11-based
/RT11	MACRO Assembler	RT11-based

Input File List

The LINK command creates an executable file from one or more of your compiled or assembled object files. You specify the input files, that is, the object files, in one of two places in the LINK command: either beginning on the same line as the LINK command or in response to the prompt, depending on whether you use the /STRUCTURE qualifier to request overlays.

In any case, the standard RSTS/E defaults for file specifications apply. The default device is the public structure (SY:); the device must be disk. The default file type for input files (your object files to be linked) is .OBJ. Note that for COBOL programs, the LINK command expects to find a skeleton file (type .SKL) on the same device and account, with the same file name, as each .OBJ file listed. (A COBOL compile automatically produces .SKL and .OBJ files.)

Simple (Nonoverlaid) Linking

For a simple (nonoverlaid) link, specify the object files beginning on the same line as the LINK command. If you have many files, you can continue a line by typing a hyphen (-) at the end of the line.

The LINK command translates the command you specify into a command line to another program. The translated command cannot exceed 127 characters (80 characters for a COBOL program). If the continued command line is too long, you get the error message ?Command too long. If you need to link more files, you must run the Task Builder or the LINK linker directly (see the *RSTS/E Task Builder Reference Manual* or the *RSTS/E RT11 Utilities Manual*).

The files you name are linked, together with routines from appropriate libraries, in the order you specify. That is, the first file you name occupies the lowest range of addresses in the program, the second file you name is linked to occupy a range of addresses immediately following the first, and so forth.

Overlaid Linking

For overlays, do not specify the object files on the same line as the LINK command. Instead, simply specify the /STRUCTURE qualifier (and other qualifiers, if you want). The /STRUCTURE qualifier causes LINK to prompt you for an overlay structure, described later in this chapter. Specify the object files in response to these prompts.

Language Qualifiers

The linking process includes routines in the executable file that are obtained from appropriate libraries. You can choose the libraries from which these routines are called. The choice depends on the qualifiers you use with the LINK command, and whether you choose to use the resident libraries installed on your system.

/BASIC
/BP2
: Indicates that the object files listed in the command are BASIC-PLUS-2 object files. Unless your system manager selects another default for your site, /BP2 is the default if you do not specify any language qualifier. (/BASIC and /BP2 are equivalent qualifiers.)

  This qualifier provides an easy way to link BASIC-PLUS-2 programs. However, DIGITAL recommends that you use the /DESCRIPTION qualifier instead of /BP2 to link nonoverlaid BASIC-PLUS-2 programs. /DESCRIPTION allows you to use the defaults you specify at installation and with the BASIC-PLUS-2 BUILD command. You cannot use these defaults when you use the /BASIC and /BP2 qualifiers with the LINK command. Instead, you must specify what you would like to use each time you type the LINK command.

  A BASIC-PLUS-2 program requires the resident library RMS (Record Management Services) if any OPEN statements include the ORGANIZATION clause or if the FMS (Forms Management System) qualifier is specified. See the *PDP-11 BASIC-PLUS-2 Language Reference Manual* for more information.

/COBOL
/C81
: Indicates that the skeleton and object files listed in the command are COBOL-81 files. (/COBOL and /C81 are equivalent qualifiers.)

  The LINK command includes RMS if the COBOL program requires it.

/DIBOL
: Indicates that the object files listed in the command are DIBOL object files.

/FOR
Indicates that the object files listed in the command are FORTRAN-IV object files.

/F77
Indicates that the object files listed in the command are FORTRAN-77 object files.

With FORTRAN-77, you can use I/O channels 1-14 in your program. The maximum record size for I/O is 512 bytes. See the *PDP-11 FORTRAN User's Guide* if you need a larger record size.

If your program stores a format specification in an array, the size of the specification is limited by the size of an internal buffer, which is 64 bytes. See the *PDP-11 FORTRAN-77 User's Guide* if your program requires more space.

The LINK command always includes RMS with this qualifier. You cannot specify /OTS=RESIDENT with this qualifier. FORTRAN-77 does not have an Object Time System (OTS) resident library.

/RSX11
Indicates that the RSX-11-based assembler (using the MACRO/RSX11 command) on RSTS/E systems (MAC.TSK) produced the object files listed in the command.

The LINK command includes RMS only if you specify the /RMS qualifier. The default for MACRO programs is /NORMS.

/RT11
Indicates that the RT11-based assembler (using the MACRO/RT11 command) on RSTS/E systems (MACRO.SAV) produced the object files listed in the command.

The LINK command cannot use RMS with this qualifier.

Forms Management System Qualifier

/FMS
/NOFMS
Indicates that you used the capabilities of the DIGITAL Forms Management System (FMS-11) in your program. You can use this qualifier in conjunction with any of the RSX-based language qualifiers. You cannot use it with the /FOR or /RT11 qualifiers.

/FMS=RESIDENT
Specifies the FMS resident libraries FDVRDB and FDVRES. FDVRDB is the FMS resident library that uses debug mode. FDVRES is the resident library without debug mode.

Your system manager can decide to let these libraries belong to a cluster. Cluster libraries are useful if you need to use more than one resident library. When they are clustered, resident libraries share the same virtual address space in your

program. See the *RSTS/E Task Builder Reference Manual* for more information on cluster libraries.

Without cluster libraries, FMS uses 4K words of virtual address space in your program.

With cluster libraries, the LINK command clusters the FMS resident libraries with all resident libraries being used. However, the LINK command will not use cluster libraries with the /DIBOL qualifier.

/FMS=NORESIDENT
Specifies FMS without a resident library.

Debugging Qualifier

/DEBUG
/NODEBUG
Invokes the appropriate debugging tool for a program that is written in COBOL-81, DIBOL, or RSX-11-based MACRO, and/or uses FMS. The debuggers invoked with each qualifier follow. The default for each language is /NODEBUG.

Note that when you use the /DEBUG qualifier with either /C81 or /DIBOL, you must also use the /DEBUG qualifier when you compile the program. If your program consists of a main program and one or more separately compiled subroutines, and you want to use the debugging tool on the entire program, you must compile each subroutine with the /DEBUG qualifier. You can mix sections of your program that have been compiled using /DEBUG with sections that have not been compiled using /DEBUG. However, the debugging tools will be used only in the sections compiled with /DEBUG.

For a COBOL-81 program, /DEBUG invokes the COBOL-81 Symbolic Debugger. See the *COBOL-81 RSTS/E User's Guide* for information on the debugging techniques used.

For a DIBOL program, /DEBUG invokes the DIBOL Debugging Technique (DDT). See the *CTS-500 DIBOL User's Guide* for information on the debugging techniques used.

If you are using FMS, /DEBUG invokes the debug mode of FMS in addition to the other debugging tool. The debug mode of FMS is not available with COBOL-81. See the *FMS-11/RSTS Reference Software Manual* for information on the debugging techniques used.

For an RSX-11-based MACRO program, /DEBUG invokes the Octal Debugging Tool (ODT). See the *IAS/RSX-11 ODT Reference Manual* for information on the debugging techniques used.

For a FORTRAN or RT11-based MACRO program, /DEBUG has no meaning.

If you use /DEBUG and it cannot be applied, you receive the error message:

Description Qualifier

/DESCRIPTION

Indicates that the input file describes the process for linking the program, including object files to link, the executable and/or map files to create, and various other options. A description file is sometimes called a "task builder command file."

If you program in BASIC-PLUS-2, you can specify /DESCRIPTION if you want to use CMD and ODL Task Builder command files created by the BASIC-PLUS-2 BUILD command. Although /DESCRIPTION and /BP2 can perform the same functions, unlike /BP2, you can tailor /DESCRIPTION to your system, generally resulting in a smaller program.

You cannot use this qualifier with any of the other LINK qualifiers. The default file type is .CMD. You can specify only one input file with /DESCRIPTION.

Address Space and Library Qualifiers

Two general types of libraries may be available on your system:

- Disk libraries (nonresident)
- Resident libraries

A library is a collection of software modules in a single file. For disk libraries, the Task Builder takes a copy of each routine that you reference in your program and builds it into your program.

Your system manager defines libraries as resident (residing in computer memory) so that more than one user can share them. Instead of building routines into your program (as is done with disk libraries), you share a copy of the library. The copy is resident in memory as long as you or someone else requires it.

Because they are shared, resident libraries occupy less space on the system. Your program uses less memory and processor time: therefore; it runs faster. However, because it is necessary to dedicate permanent physical memory for resident libraries, your system manager might choose not to install them. By default, the LINK command uses resident libraries if they are present on your system. Otherwise, LINK uses the appropriate disk libraries.

Because both resident and disk libraries take memory from your job, their use affects the amount of memory that is available for your program to work in. There is a 32K-word limit on job space for programs on RSTS/E. Your system manager may also have set a parameter called the SWAP MAX that can limit program size further. SWAP MAX has an upper limit of 31K; however, your system manager can reassign this upper limit to a lower value.

The BASIC-PLUS-2, COBOL-81, and RMS resident libraries can be clustered, which means they share virtual address space in your program. In general, 24K words of address space are available to your program when you use resident libraries that are clustered.

To learn which libraries are available on your system, type SYSTAT/L. Note that the defaults are to use the resident libraries on your system. Therefore, the following descriptions of the resident libraries are of interest only if you want to change the defaults.

/OTS=RESIDENT
Specifies the resident library for a language Object Time System (OTS). You can use /OTS=RESIDENT only with BASIC-PLUS-2, COBOL-81, and DIBOL.

/OTS=NORESIDENT
Specifies the disk library for the language you are using.

/RMS
/NORMS
The /RMS qualifier specifies that RMS will be used and is the default for all RSX-11-based langugages except MACRO. The default for MACRO programs is /NORMS. For more information, see the previous section on Language Qualifiers.

/RMS=RESIDENT
Specifies the RMS resident library, RMSRES.

/RMS=NORESIDENT
Specifies the RMS disk library.

Output File Qualifiers

Two qualifiers, /[NO]EXECUTABLE and /[NO]MAP, control the output files produced by the LINK command.

/EXECUTABLE[=file-spec]
/NOEXECUTABLE
Indicates that you want the LINK command to produce an executable file. The default is /EXECUTABLE.

If you use the /EXECUTABLE qualifier, you can specify a name for the executable file. If you omit the file specification, LINK produces an executable file on the public disk structure (SY:) in your account with a file name the same as the first file name in the input list. The default file type is or .SAV (for RT11-based languages).

If you use the /NOEXECUTABLE qualifier, the LINK command does not produce an executable file. You must use the /MAP qualifier if you use /NOEXECUTABLE.

/MAP[= file-spec]
/NOMAP
Use the /MAP qualifier to indicate that you want the LINK command to produce a memory map file. Use the /NOMAP qualifier, or (since /NOMAP is the default) simply omit the qualifier if you do not want a memory map file. You must specify /MAP if you use the /NOEXECUTABLE qualifier.

If you use /MAP, you can also assign a file specification for the memory map file. If you omit the file specification, LINK produces map file on the same device, with the same account and name as the executable file. The default file type is .MAP.

For RSX-based languages, specifying the /MAP qualifier produces the Task Builder's memory map file. For RT11-based languages, the /MAP qualifier produces the LINK.SAV memory map.

Overlay Qualifier

/STRUCTURE
Indicates that you want to be prompted for an overlay structure (see the following sections).

When You Can Use /STRUCTURE

If you wrote your program in BASIC-PLUS-2, DIBOL, or FORTRAN-77, you can use the /STRUCTURE qualifier with the LINK command to specify an overlay structure. The /STRUCTURE qualifier presents a dialogue that makes it easy to work with overlays.

Note

Instead of using LINK and /STRUCTURE, you can specify overlays for these languages using the Overlay Description Language (ODL) recognized by the Task Builder. The *RSTS/E Task Builder Reference Manual* describes ODL. Using the ODL language is more complex than using the /STRUCTURE qualifier with LINK; however, it is also more powerful.

COBOL programmers work with overlays within the language itself, by using the segmentation facility of the COBOL compiler. The *RSTS/E COBOL User's Guide* describes this facility. You do not need to use the /STRUCTURE qualifier if you are doing overlays in COBOL; the LINK command processes the skeleton files output from the COBOL-81 compiler.

If you program in FORTRAN-IV or in MACRO under the RT11 run-time system, you must run the LINK.SAV program directly to specify an overlay structure. The *RSTS/E RT11 Utilities Manual* describes this program in detail.

When you use the /STRUCTURE qualifier, you are prompted for an overlay structure.

When Must You Use Overlays?

If your program is too large to fit in the space available, you must specify an overlay structure for it. The discussion of the language qualifiers tells how much space is available for programs written in the various languages when you use the LINK command.

The easiest way to find out if your program is too large is to try to link it using a language qualifier. If you get the error message ?Task has invalid memory limits, your program is too large.

What Are Overlays?

The best way to explain overlays is by example. Suppose your program consists of a main program (called MAIN) and two separately compiled subroutines (called SUB1 and SUB2). Suppose further that MAIN calls both SUB1 and SUB2, and that neither SUB1 nor SUB2 contain any calls to separately compiled subroutines or to MAIN.

Figure 8-1 shows the call structure.

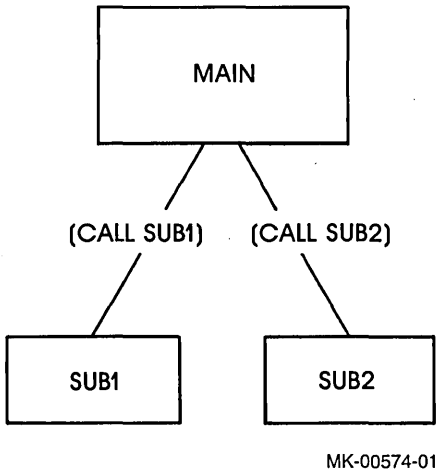

Figure 8-1: Outlining the Call Structure

You can specify an overlay structure such that MAIN is loaded when the program is first run. When MAIN calls SUB1, the loading code built into MAIN during the LINK process loads SUB1 for execution. Then, when control passes back to MAIN and it calls SUB2, SUB2 is brought in to memory overlaying SUB1.

Figure 8-2 shows a simple overlay in memory. Note that SUB1 and SUB2 do not call or use data from each other. This logical independence is necessary for program pieces that overlay each other. In this example, calls or references to data that are not currently in memory must be made from the root, which is the MAIN program.

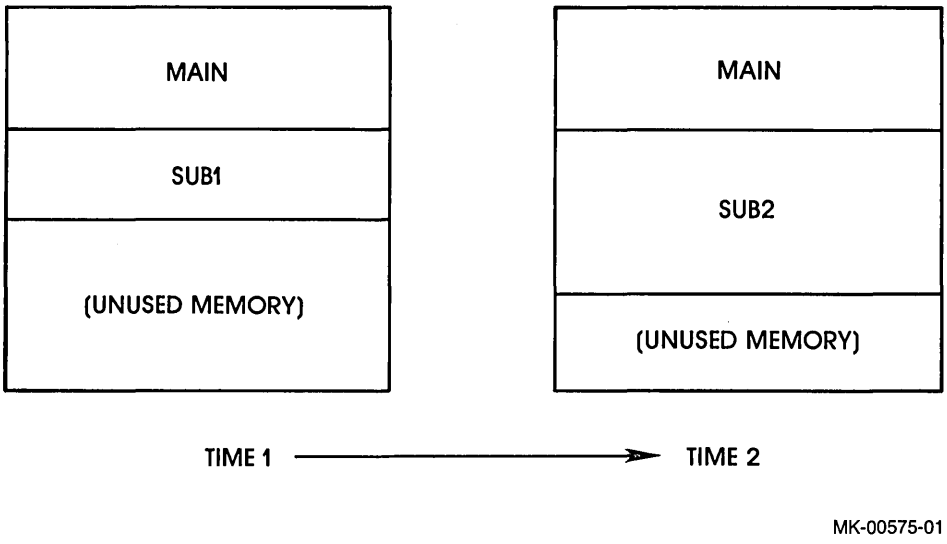

Figure 8-2: A Simple Overlay in Memory

Rules for Constructing Overlays

In general, you must structure an overlay tree so that calls or references to data take place between overlay pieces that are on the same path. A path is any route from the root of the structure that follows a series of branches to an outermost piece of the tree. Calls or references to data cannot take place between pieces that are along different paths.

Consider the overlay structure shown in Figure 8-3. The structure shows pieces that overlay each other as separate branches of a tree. C and D would start at the same virtual address, as would E, F, and G. The paths in this structure are A-B-C, A-B-D-F, A-B-D-E, and A-B-D-G. Calls may be made between pieces on any of these paths. However, F could not call G, E, or C; C could not call D, E, F, or G, and so forth.

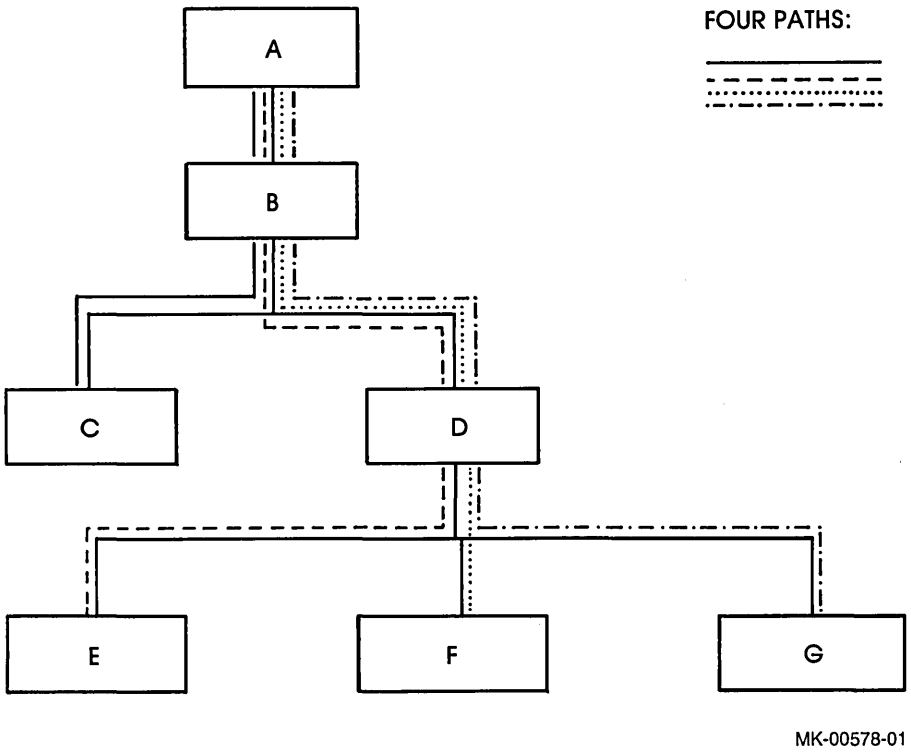

Figure 8-3: Separate Paths in An Overlay Structure

When you use the /STRUCTURE dialogue to specify an overlay structure, you must be careful to specify a structure that does not contain calls or references to data that cross paths.

You must also be careful when you work with common areas in an overlay structure. If a FORTRAN program and subprogram both define a common area A, for example, where is the space for A to be allocated?

Two examples are shown in Figure 8-4. The common block COMA is defined in the pieces A2 and A21. The space for COMA is allocated in A2 because that piece is closer to the root. The common block COMB, however, is defined in A0 and B0. These pieces are not on the same path, so the space for COMB is allocated in both A0 and B0. Note that A0 and B0 cannot communicate through COMB. When the overlay containing B0 is loaded, for example, any data stored in COMB by A0 is lost.

The /STRUCTURE dialogue that follows allows you to place COMB in the root of the overlay structure, where it can be accessed by A0 and B0 properly.

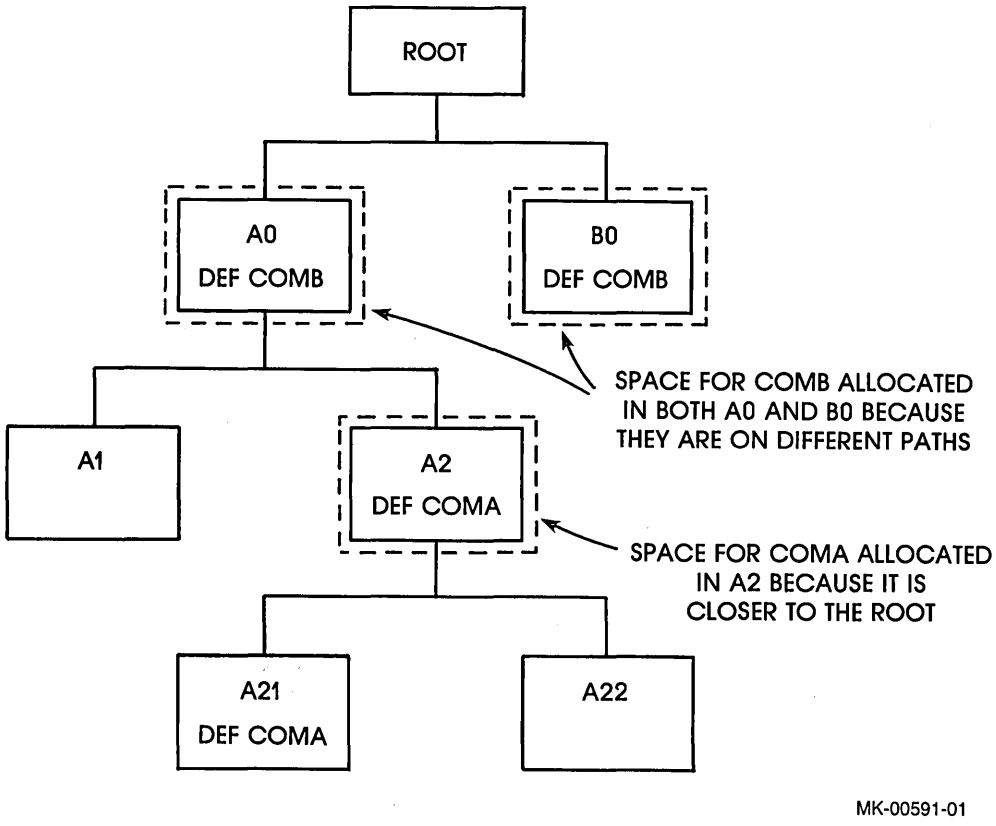

Figure 8-4: Allocating Space for Common Areas

The /STRUCTURE Dialogue

You can use the /STRUCTURE qualifier with /F77, /BASIC, or /BP2 and the /DIBOL qualifier. You cannot use /STRUCTURE with the /FOR, /RT11, /COBOL, or /C81 qualifiers.

When you use /STRUCTURE, the LINK command prompts you for an overlay structure, as shown in the following example. This example shows how you would link the overlay structure shown in Figure 8-4:

```
$ LINK/F77/STRUCTURE (RET)
ROOT files: ROOT (RET)
Root PSECTS: COMB (RET)
Overlay: A0+ (RET)
  Overlay: A1 (RET)
  Overlay: A2+ (RET)
    Overlay: A21 (RET)
    Overlay: A22 (RET)
    Overlay: (RET)
 Overlay: (RET)
Overlay: B0
Overlay: (RET)
```

ROOT files: Prompt

The first prompt presented is "ROOT files:". You respond to this prompt by typing the object files that are to form the root of the overlay tree. In this case, there is only one such file, named ROOT. If you have more than one such file, type them all on one line and separate them with commas. If you need to continue, type a hyphen (-) at the end of a line. The LINK command then displays the Continue: prompt.

There is a limit on the number of characters you can type for root files; the translated command line must be less than or equal to 127 characters.

Root PSECTS: Prompt

The next prompt presented is "Root COMMON areas:". If you are a high-level language programmer, you will probably be interested in this question only if you define a common area in two pieces of your program that overlay each other. When you need to place such a common area in the root of the overlay structure, where it can be accessed properly as a common area, you specify the name of the common area in response to this question.

If you need to place more than one such common area in the root, you simply type the common area names, separated by commas. To continue a line, type a hyphen or a comma at the end of the line. If you type a comma, the LINK command assumes you have finished one PSECT name and displays the Continue: prompt for you to type more. Typing a hyphen lets you continue typing a PSECT name you began on the previous line.

There is no limit to the number of program sections you can place in the root of your program.

Overlay: Prompts

With further /STRUCTURE prompts, you define the overlay structure beyond the root section. You respond to the "Overlay:" prompts with file specifications for object

files to be linked. You use two symbols, the plus sign (+) and the comma (,) to show how the files are to be overlaid. You can also use the exclamation point (!) to include comments in the overlay dialogue.

The Plus Sign (+) Symbol

Consider again the LINK command to overlay the structure shown in Figure 8-4:

```
$ LINK/F77/STRUCTURE RET
ROOT files: ROOT RET
Root COMMON areas: COMB RET
Overlay: A0+ RET
  Overlay: A1 RET
  Overlay: A2+ RET
    Overlay: A21 RET
    Overlay: A22 RET
    Overlay: RET
 Overlay: RET
Overlay: B0 RET
Overlay: RET
```

As shown, you indicate a branch in the overlay structure by ending a line with a plus sign following the pieces from which the branch occurs. The next Overlay: prompt is indented two places, to show that you are at the next overlay level. For example, A1 and A2 overlay each other; their addresses begin immediately after the address taken by A0. Likewise, A21 and A22 overlay each other after A2.

Repeat this process to obtain the desired overlay structure. When you want to return to the previous level in the overlay structure, press the RETURN key in response to an Overlay: prompt.

Note that you can nest overlays to seven levels.

The Comma (,) Symbol

You use the comma (,) operator in an Overlay: prompt between files that are to be concatenated in the overlay structure.

For example, consider the structure in Figure 8-5.

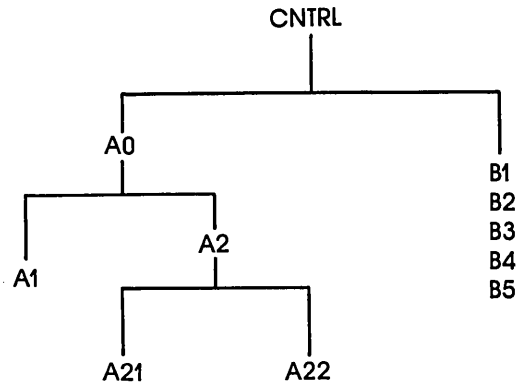

Figure 8-5: Overlay Structure Using Concatenated Files

This structure is similar to the one shown in Figure 8-4, except that the files B1, B2, B3, B4, and B5 are concatenated to form one overlay. Use commas to separate files that are to be concatenated. The following example shows how to use LINK with this structure:

```
$ LINK/F77/STRUCTURE RET
ROOT files: CNTRL RET
Root COMMON areas: RET
Overlay: A0+ RET
  Overlay: A1 RET
  Overlay: A2+ RET
    Overlay: A21 RET
    Overlay: A22 RET
    Overlay: RET
  Overlay: RET
Overlay: B1,B2,B3,B4,B5 RET
Overlay: RET
```

If you end a line with a comma, the LINK command assumes that you want to continue concatenating files on the same branch. A Continue: prompt at the same level displays for you to continue. For example, the following two lines are the same as the Overlay: B1,B2,B3,B4,B5 line shown previously:

```
Overlay: B1,B2,
Continue: B3,B4,B5
```

The Memory Map File

As noted in the section on output file qualifiers, the /MAP qualifier causes the LINK command to produce a memory map file. This file is a detailed listing of the virtual addresses that your program occupies. You can use the /MAP qualifier with any language qualifier.

There are two maps available, depending on the language qualifier you use. The RSX-based languages produce a memory map that is described in the *RSTS/E Task Builder Reference Manual.* The RT11-based languages produce a memory map that is described in the *RSTS/E RT11 Utilities Manual.*

You will probably be interested in the memory map only if you are working with overlays and want to examine in some detail what the LINK command has done for you.

Figure 8-6 shows the first page of a memory map the following LINK command sequence produces.

```
$ LINK/STRUCTURE/MAP=TRY2/EXEC=TRY2 (RET)
ROOT Files: USER (RET)
Root COMMON areas: (RET)
Overlay: INTRO (RET)
Overlay: CRUNCH (RET)
Overlay: CHATR (RET)
Overlay: (RET)

TRY2.TSK    Memory allocation map    TKB 08_.006        PAGE 1
                          22-JUN-85    13:42

PARTITION NAME : GEN
IDENTIFICATION : 16RE
TASK  UIC      : [1,196]
STACK    LIMITS: 001000 001777 001000 00512.
PRG XFR ADDRESS: 021154
TOTAL ADDRESS WINDOWS: 4.
TASK   EXTENSION    : 512. WORDS
TASK  IMAGE  SIZE   : 6048. WORDS
TOTAL  TASK  SIZE   : 6560. WORDS
TASK ADDRESS LIMITS: 000000 027447
R-W DISK BLK LIMITS: 000002 000036 000035 00029.

TRY2.TSK OVERLAY DESCRIPTION:

BASE     TOP         LENGTH
----     ---         ------
000000   022127   022130   09304.        USER
022130   023707   001560   00880.             INTRO
022130   022577   000450   00296.             CRUNCH
022130   024317   002170   01144.             CHATR
024320   025613   001274   00700.        R3PUT
025614   027447   001634   00924.        R3UPDA
```

Figure 8-6: Sample From A Memory Map File

Figure 8-6 shows that the memory map file is obscure in terms of what you specify in the LINK command. However, you can determine some basic facts:

- The size of the program is listed in the line headed "TOTAL TASK SIZE". In this case, the program is 6560 words long.
- The section headed TRY2.TSK OVERLAY DESCRIPTION shows the overlay structure constructed by LINK. The first two columns of information give the starting address (BASE) and ending address (TOP) of each major overlay piece in the structure. The next two columns (LENGTH) give the length of each piece in octal and decimal number of bytes, respectively.
- The original LINK command dealt with four files: USER, INTRO, CRUNCH, and CHATR. You can see the overlay structure specified by the indentation in the memory map: INTRO, CRUNCH, and CHATR overlay each other following USER.
- R3PUT and R3UPDA are overlay pieces from the disk library RMSLIB.OLB, which the LINK command has placed in two special overlay structures called co-trees. The *RSTS/E Task Builder Reference Manual* describes co-trees.

The Temporary Files Produced by LINK

When you use the LINK command with RSX-based languages, the LINK command processor creates two files for the Task Builder: a command file and an ODL file. These files tell the Task Builder what to do in creating the executable file and map file.

These files are left in your account. The .TMP file type indicates that they are temporary files; that is, they are deleted from your account when you log out. The command file is the same type of command file you would use with the /DESCRIPTION qualifier. (These temporary files are not created if you use the /DESCRIPTION qualifier.)

RUN

Runs an executable file.

Format

RUN file-spec

Prompts

Program: file-spec

Command Parameters

file-spec

Specifies an executable file.

If you do not specify a file type, the RUN command uses any of the following file types, which correspond to executable files:

- .BAC — A BASIC-PLUS compiled program. (See the BASIC command description for more information.)
- .COM — A DCL command file. You may use the [#] command on any .COM file, but to use the RUN command, the executable bit (64) must be set.
- .SAV — A "save image" file, which runs under the RT11 run-time system. (For example, a FORTRAN-IV executable file is a save image file.) Save image files are produced by LINK/FOR or by LINK/RT11.
- .TSK — A "task" file, which runs under the RSX run-time system (or a derivative of RSX). See the LINK command description and the *RSTS/E Task Builder Reference Manual* for more information.

Your installation may have a different set of executable file types, depending on the run-time systems your system manager has installed.

If you omit a file type, the RUN command searches the specified directory for a file having each runnable file type. The order of the search depends on your installation. For example, at some installations the RUN command might search for a .TSK file first. If that fails, run then searches for a .BAC file, then a .SAV file. However, another installation might search for file types in a different order.

You cannot use wildcard characters in the file specification, and you must specify a file name.

Examples of using the RUN command are:

```
$ RUN $SYSTAT
```

This example runs a system program named SYSTAT. The dollar sign ($ in $SYSTAT) indicates the directory [1,2], which is the system library.

```
$ RUN BANKER
```

This example runs a user program named BANKER in your account on the public structure.

DCL Error Messages A

This appendix lists some DCL error messages that you may encounter when using RSTS/E. Table A-1 also indicates the probable cause of the error in your use of a command or system program.

If you perform DECnet/E operations, you may receive messages that are described in DECnet/E documentation but not in this appendix. In addition, the utility programs that support DCL can produce messages not all of which are listed here. Those messages are listed in the *RSTS/E Utilities Reference Manual.*

Special Characters Used in Error Messages

The question mark (?) and percent sign (%) characters preceding a message indicate its severity. A single question mark precedes nonfatal error messages, which means that the command failed to do what you asked but you can continue. For example:

```
?Too many parameters
```

In this case, the command failed because you included too many parameters.

A double question mark precedes severe error messages, which means that the command failed to do what you asked and you cannot continue. For example:

```
??Fatal system I/O failure
```

By contrast, the percent sign identifies a warning message, which means that the command may not have worked as you intended. For example:

```
%Logical name has not been assigned
```

In this case, the MOUNT command worked but no logical name was assigned to the device.

Informational messages do not have a preceding character; they provide extra details about the effects of a command. For example:

```
Queue file being reorganized - please wait...
```

This message informs you that, as a result of your command, the file is being reorganized.

Angle brackets (< >) surrounding text indicate a place holder for what the system inserts when an error message occurs.

Table A-1 lists the DCL error messages that you can get when using RSTS/E and the probable cause.

Table A-1: DCL Error Messages

Message and Meaning
?Account or device in use The device cannot be mounted or dismounted because it is open or has one or more open files. This error can also occur when a user attempts to access a non-shared disk from a job other than the one that mounted the disk.
?Additional argument required You did not specify enough arguments in the DCL function.
?Additional qualifier required You did not include a qualifier required in a command.
?Ambiguous keyword You abbreviated a keyword too short; DCL cannot distinguish it from other keywords.
?Argument not allowed You used an argument with a qualifier that does not accept one.
?Argument required An argument was not supplied with a qualifier which required one. For example, /SINCE=, without an argument, would generate this error. The user did not supply an argument to a DCL function that required one.
??Bad directory for device The directory of the device referenced is in an unreadable format. For example, the tape format differs from the format you specified with the MOUNT command.
?Can't access terminal <error text> An error occurred while attempting to access the terminal to send the setup file to.
?Can't find file or account Either the account or file does not exist, or you typed a file specification incorrectly.
?Can't mount a private disk as public You tried to mount as public (using the MOUNT command's /PUBLIC qualifier) a disk that was initialized as private.
?Can't rebuild disk because device is write protected A privileged user attempted to rebuild a disk (using the MOUNT command's /REBUILD qualifier), and the drive is write protected. Therefore, the mount and rebuild operations fail.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?Channel already open The channel specified in the OPEN command already has a file open on it. You must either specify a different channel number or use the CLOSE command to close the channel first.
?Channel not open for input The channel specified by a READ command is either not open or open for writing only.
?Channel not open for output The channel specified by a WRITE command is either not open or open for reading only.
?Command not installed You typed a DCL command that has not been installed on your system. The DCL keyboard monitor could not find the utility needed to carry out your command. This message can also be displayed if you type a remote file specification in a command that allows remote file specifications, but your system does not have DECnet/E.
?Command too long The length of the command, including continuation lines, exceeds the maximum allowed for the command. Or, the length of the translated command string exceeds the maximum allowed for the command. For example, this message occurs with the LINK command in either of two cases: • The text you typed in response to the $ prompt and the Files: or Root files: prompt added up to more than 127 characters after translation. • With LINK/COBOL, the text was longer than 80 characters after translation.
%Command will proceed as requested The user's symbol tables could not be written out to the DCL work file, but the command will be processed. The user may be over quota. Corrective action should be taken before the user continues processing.
?Comma required A DCL function required two or more arguments and they were not separated by a comma.
?Conflicting arguments Two arguments are considered to conflict within a qualifier argument list. For example, you cannot specify both /BRIEF and /RESET with the SHOW ACCOUNT command.
?Conflicting elements You specified qualifiers or parameters in a command that do not agree in type (for example, a print qualifier with a batch server).
?CPU limit exceeds queue's maximum You specified a /CPU_LIMIT value in a SUBMIT command larger than the maximum allowed for the queue.
?Data error on device One or more characters may have been transmitted incorrectly due to a parity error, bad punch combination on a card, or similar error.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?Data error or incorrect density This message occurs for one of the following reasons: • You specified the incorrect density. • You did not specify a density, but your system's default density is incorrect for the tape. • The data on the tape is damaged. • The tape drive is faulty. To correct the problem, specify the correct density. If that does not solve the problem, try mounting the tape on another drive.
?Default macro for KB <n>: does not exist The SET TERMINAL/INQUIRE command could not recognize the terminal and there was no default macro specified for this keyboard in the default file, TERDFL.SYS. You should supply a default macro for the keyboard. Until that is done, the keyboard's characteristics have to be set manually.
?Device does not exist The device name you specified does not exist on your system.
%Device hung or write locked %Dismount will proceed as requested You attempted to dismount a disk that has been physically dismounted. However, the logical dismount will succeed.
?Device hung or write locked Check the hardware condition of the device requested. Possible causes of this error include a line printer out of paper or a disk drive being off line.
%Device in use You specified /NOSHAREABLE in an INITIALIZE/SERVER command; however, the print server's device is not available or no psuedo-keyboards are currently available for the batch server. PBS allocates the device as soon as it becomes available.
?Device is not a terminal A device other than a terminal was specified. Only specify keyboards.
?Device must be disk You specified a file on a nondisk device, where a disk device is required.
?Device not available The specified device exists on the system, but an attempt to allocate or use it is prohibited for one of the following reasons: • The device is currently reserved by another job. • The device requires privilege for ownership and you do not have the necessary privilege. • The device or its controller has been disabled by the system manager. • The device is a keyboard line for pseudo keyboard use only.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?Device not file-structured An attempt was made to access a device, other than a disk drive or magnetic tape drive, as a file-structured device.
?Device not write protected You specified the /NOWRITE qualifier when you tried to mount a tape, but the device is not write protected. Write protect the device by removing the plastic ring from the tape hub.
?Device offline You tried to use a tape or disk, but the device is off line.
%Device write protected The tape or disk is protected against write access. Therefore, you can only read files on the device, but cannot write (perform output) to the device. If you want to write to the device, first write enable the device, next dismount the device, and then mount the device again.
?Device write protected You requested write access to a device that is write protected. Write enable the device and then retype the command.
?Directory does not exist A file specification indicates a directory that does not exist on the particular disk. Or, a wildcard directory specification failed to produce a match on the disk specified. This error can occur only with disk files.
??Disk error during swap A hardware error occurs when your job is swapped into or out of memory. The contents of the job area are lost, but the job remains logged in to the system and returns to DCL. Report such occurrences to the system manager.
Disk is being rebuilt - wait... This informational message is displayed when a user attempts to rebuild a disk. The disk is logically mounted when the rebuilding operation is complete.
%Disk is restricted and mounted non-shared, read-only A user specified /NOREBUILD (to suppress rebuilding) in the MOUNT command and the disk was dirty. Therefore, the disk is mounted nonshared and restricted, and read-only access is granted. ("Restricted" means that access is allowed only to users with DEVICE privilege.)
?Disk is mounted non-shared This message can be displayed for one of two reasons: • A user with MOUNT privilege attempted to dismount a private or public disk that was mounted as nonshared by specifying /PUBLIC in the DISMOUNT command. • A user without MOUNT privilege attempted to dismount a private disk that was mounted as nonshared by specifying /PUBLIC in the DISMOUNT command. Therefore, the dismount fails. In both cases, the requests to mount and dismount are from different jobs.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?Disk is mounted private or non-shared This message can be displayed for one of two reasons: • A user with MOUNT privilege attempted to dismount a private or public disk that was mounted as nonshared (by the same job) or private by specifying /PUBLIC in the DISMOUNT command. • A user without MOUNT privilege attempted to dismount a private disk that was mounted as nonshared (by the same job) or private by specifying /PUBLIC in the DISMOUNT command. Therefore, in both cases the dismount fails.
?Disk is mounted public A user with MOUNT privilege attempted to dismount a public disk that was mounted as public, without specifying /PUBLIC in the DISMOUNT command, so the dismount fails.
%Disk is mounted read-only This warning occurs if you do not specify either read or write access (/NOWRITE or /WRITE) when mounting a disk initialized as read-only. Therefore, you are granted read access only.
?Disk needs rebuilding but device is write protected A privileged user attempted to mount a dirty disk (and did not specify /REBUILD or /NOREBUILD), but the device is write protected and the automatic rebuilding cannot be performed. Therefore, the mount fails.
?Disk needs rebuilding but you do not have the necessary privilege This message occurs when a user without MOUNT privilege tries to mount a private disk that was not logically dismounted. The system manager can correct the situation by rebuilding the disk.
?Disk pack is locked out This message occurs when a user without DEVICE privilege attempts to open files on a disk restricted to users with DEVICE privilege.
?Disk pack is not mounted The DISMOUNT command was attempted, but the disk pack was not mounted on the specified disk drive.
??Disk pack mount error Fatal disk mounting error. The disk is corrupt and cannot be successfully mounted with the MOUNT command.
?Division by zero The expression specified by the user attempted to divide by zero.
?Do not specify file name or type This message can occur with the first parameter of the ASSIGN command. Remember to use a space between the first parameter (the string you assign) and the second parameter (the name you assign it). For example, "ASSIGN DR3: X", not "ASSIGN DR3:X".

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?End of file on device A READ command was attempted but no data was available. Either the file is empty or you attempted to read past the last record in the file.
?Equal sign required You used a qualifier that requires an argument, but you did not give an argument.
?Error in forms definition <forms-name> <error message> An error, as the error message in the second line describes, was discovered in the forms definition displayed. You used a qualifier that requires an argument, but you did not give an argument.
?Error number <nn>(message text is not online) An I/O error occurred while the system was attempting to retrieve an error message. Possible causes could be that the device containing the system error file (ERR.SYS) is off line, or that the system error file contains a bad block. This is a serious error, and should be reported to the system manager.
?Error opening PBS file <file-spec> <error message> An error, as the error message in the second line describes, occurred opening the PBS file indicated.
<error text> opening setup file <filename> An error occurred while trying to gain access to the setup file.
<error text> opening speed file An error occurred while trying to gain access to the speed file. The speed is not set and the rest of the command is executed.
<error text> opening/reading default file An error occurred while processing the default file.
<error text> reading setup file <filename> An error occurred while processing the setup file.
<error text> reading speed file An error occurred while processing the speed file. The speed is not set and the rest of the command is executed.
??Fatal system I/O failure An I/O error has occurred on the system level. The results of the last command are unpredictable. This error is caused by a hardware condition. Report such occurrences to your system manager.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?File does not exist This message occurs for one of the following reasons: • An input file that must be present is not present. • A wildcard file specification does not match any files. • A file is not accessible because of its assigned protection code.
?File name required A file specification you typed does not include a file name, but one is needed.
?File specification required This error occurs if you specify node:: without a file specification (except with DIRECTORY). It also occurs if you have two commas with no characters between them in a file specification list.
?Files cannot be on different nodes This message occurs with network operations. Each input file specification you include in a network command must be on the same node.
?File must be disk The file specified on the OPEN command was not a disk file.
?Form <form-name> does not exist The form name you specified was not defined by the system manager. See your system manager to find out what form names are available.
?Forms Definition File does not exist The forms definition file, PBS$:FORMS.SYS does not exist.
?Forms not defined [for server <server-name>] The forms you specified could not be found in the forms definition file.
%ID label ignored You mounted a tape in DOS format and specified an ID label. Identification labels are not encoded on DOS tapes; therefore, the label you specified is not recognized by the MOUNT command.
%ID label should be specified when you mount an ANSI tape You mounted a tape in ANSI format, but did not specify the tape identification label. DIGITAL recommends that you specify the identification label. Thus, the MOUNT command can verify that the tape you selected has been mounted.
?ID labels don't match The identification label you specified does not match the identification label encoded on the tape. To correct the problem, specify the correct identification label. If you do not know what identification label is encoded on the tape, you can omit the ID label from the MOUNT command.
?Illegal byte count for I/O The range of memory starting at the load address given is not available. Refer to the memory status report of a display program (SYSTAT or DISPLY) to select an available range of memory.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?Illegal filename The file name given in the command contains characters other than alphabetic or numeric characters.
?Illegal switch usage A CCL command contains an error in an otherwise valid CCL switch (qualifier). For example, you cannot use the /SI:n switch without a value for n or a colon; or you cannot specify more than one of the same type of CCL switch.
?Illegal value <n> The swap file number specified is not 0, 1, or 3. (The swap file 2 already exists on the system disk and need not be added.)
?Impossible density for this device You tried to mount or initialize a tape, but specified a density not available on the tape drive you used. To correct the problem, use another drive or specify a different density.
?Incorrect density You specified a density different from the tape's density. This message appears only if you are using a tape drive that determines the tape's correct density. Therefore, you do not need to specify the tape density with the MOUNT command.
?Incorrect density or uninitialized tape This message can occur for the following reasons: • You tried to mount a tape that had not been initialized. • You specified an incorrect density. • You did not specify a density, and your system default density was incorrect for the tape. If the tape has not been initialized, use the INITIALIZE command. Otherwise, specify the correct density.
?Invalid account The account specified was not valid either containing invalid characters (for example, [^^^]) or an invalid format (for example, [1,2345]).
?Invalid argument This error can occur when you use a qualifier in the form /qualifier=argument. The qualifier you used is spelled properly, and the equal sign (=) or colon (:) is present; however, the argument is either missing or syntactically invalid. This error message is displayed when there is no more specific message to describe the syntax error.
?Invalid at interactive level You specified a command that is invalid at the interactive level. (For example, GOTO LABEL)
?Invalid BATCH command Although the command is valid in interactive mode, the command is not a valid BATCH command because it implies or requires interaction with the user. (For example, INQUIRE, SET NODATA)

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?Invalid CCL command You used the CCL prefix followed by a command that is not installed as a CCL command on your system.
?Invalid channel number The I/O channel number specified was invalid. Channel numbers can range from 1 to 13 on OPEN, READ and CLOSE commands, and from 0 to 13 on WRITE commands.
?Invalid character You typed an invalid punctuation character.
?Invalid command The command name you gave is not a DCL command, and is not defined on your system as a CCL command. Or, the line begins with a punctuation character rather than with a keyword.
?Invalid date A date either has improper syntax, represents a nonexistent date (for example, 30-Feb), or represents a date before 1970 or after 1999.
?Invalid density <n> You tried to mount or initialize a tape, but specified a density other than 800 or 1600 bpi. The n is the density you specified.
?Invalid device The device specified could not be found in the list of standard devices or in the user-defined macros. Specify a valid device.
?Invalid entry name The entry name you specified is invalid.
?Invalid expression This covers a range of expression problems. The most common of these is an expression ending in an operator without supplying a second operand (for example, A = B *).
?Invalid file specification A local file specification has improper syntax.
?Invalid fill factor A fill factor was specified that was less than 0 or greater than 6.
?Invalid form definition The definition of the specified form contains an invalid keyword or keyword argument.
?Invalid form name The form name you specified is either longer than six characters or consists of one or more nonalphanumeric characters.
?Invalid function name The function name specified after F$ is invalid. The minimum abbreviation point is not met for the function (for example, F$LE is not valid because of F$LEFT and F$LEN).

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
%Invalid keyboard numbers <n , n> in default file The numbers you specified in the default file were less than 0, greater than 127, or the first number of the range is greater than the second number of the range.
%Invalid keyboard numbers <n , n> in speed file The numbers you specified in the speed file were less than 0, greater than 127, or the first number of the range is greater than the second number of the range.
?Invalid keyword The keyword is not recognized. This error occurs with keywords that are not qualifiers and are not command names. (For example, it can occur with the options of SET and SHOW, and with qualifier values that are keywords.)
?Invalid label The label specified in the command procedure contains invalid characters.
?Invalid node name The node name field of the file specification contained invalid characters.
?Invalid operator The operator you specified in the expression is not a defined logical, arithmetic, string, arithmetic comparison or string comparison operator.
?Invalid PPN The project-programmer number (PPN) you specified does not have valid syntax.
?Invalid print device The device you specified is not a valid print device because it is not a line printer or terminal.
?Invalid private delimiter The delimiter you specified was not in one of the correct formats.
?Invalid qualifier The qualifier keyword is not valid in the command you typed. (This message may indicate an error in spelling or typing.)
?Invalid qualifier for disk This message occurs when you specify an invalid qualifier for disks. For example, /FORMAT=ANSI (applies to tapes).
?Invalid qualifier for tape This message occurs when you specify an invalid qualifier for tapes. For example, /PRIVATE (applies to disks).
?Invalid queue name The queue name you specified is invalid.
?Invalid server name The server name you specified is invalid.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?Invalid speed The speed specified was not a valid speed for the interface of the terminal. Only specify speeds that the interface supports.
?Invalid symbol name You specified a symbol name that contains invalid characters.
?Invalid time A time either has improper syntax or represents a nonexistent time (like 25:00 or 13:00PM).
?Invalid width You specified a width that was less than 1 or greater than 254.
?Invalid with network file specification You gave a network file specification in one of the commands that accepts them (RENAME, COPY, and so on), but you also gave a qualifier that can be used only with local operations.
?I/O to detached keyboard This message can result from one of two actions: • You tried to perform I/O with a terminal line that is used for dial-up terminals, but nobody was dialed in. • Your job became detached (perhaps because you were dialed in and your line was later hung up) and then tried to perform I/O with the terminal. The second situation either causes the job to hibernate or causes this error condition, after which the job hibernates. You see this message when you subsequently attach to the job.
?Keyword required You typed nonalphanumeric characters when a keyword is needed instead. (If you type alphanumeric characters without a valid keyword, you receive the error message, ?Invalid keyword.)
?Label already defined A label was defined more than once in an indirect command procedure.
?Label too long The label you specified in the indirect command procedure was longer than 255 characters.
?Log file already open A log file was already currently open when you attempted to open another log file using the OPEN/LOG_FILE command.
?Log file not currently open You attempted to enable or disable logging using the SET LOG_FILE command, and a log file did not exist.
?Log file print queue closed The print queue required for a SUBMIT command's /LOG_QUEUE qualifier is closed or marked for deletion. Use a different print queue for the log file.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?Log file print queue does not exist The print queue you specified with the /LOG_QUEUE qualifier in a SUBMIT command does not exist. Specify a different queue.
%Logical name not assigned This warning can be displayed for one of two reasons: • You had INSTAL privilege and specified a logical name that was not assigned as a system logical. (This means that neither the alternate logical name nor the pack-id label were assigned.) • You did not have INSTAL privilege and specified an alternate logical name when attempting to mount a disk. The mount succeeds in both cases, but the logical name is not assigned.
?Magtape record length error When performing input from magnetic tape, the record on tape was found to be longer than the buffer designated to handle the record.
?Magtape select error When access to a magnetic tape drive was attempted, the selected unit was found to be off line. This error can occur when you transfer data to or from a tape.
?Map or executable file required With LINK, you specified /NOEXECUTABLE and did not specify a map file.
??Maximum memory exceeded This is a nonrecoverable RSTS/E error caused by the following conditions: • While loading a program into memory, the job's private maximum memory size was reached. • While executing a program, the system required more memory for string or I/O buffer space, and the job's private maximum memory size or the system maximum was reached.
?Missing closing apostrophe The user did not specify matching apostrophes ('... ') when requesting apostrophe substitution.
?Missing closing bracket This error can occur in a local or remote file specification. There is a left bracket ([) or left angle bracket (<), but no right bracket (]) or right angle bracket (>).
?Missing closing quote A quotation mark (") is not matched with another quotation mark. (This error can occur in remote file specifications.)
?Missing device or file name The command must contain either a device specification or a device and file name specification. If you use the /SI:n switch, a file name must be present.
?Missing open parenthesis You did not specify an open parenthesis [(] when one was expected.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
%More than 16 speeds specified for <terminal> While processing the TERSPD.SYS file, a terminal was found that had more than 16 speeds specified. Only the first 16 speeds are used.
?Name or account now exists You attempted either to COPY to or RENAME an existing file. This error can occur with RENAME if you do not specify /REPLACE and the output file already exists. It can also occur with COPY if you specify /NOREPLACE and the output file already exists.
?Network node names must be the same Different node names were specified on input file specifications.
?No buffer space available The system is overloaded and cannot complete your command because small buffers are currently unavailable. Try the command again later.
?No channels available A DCL I/O or command file channel is not available. You may have issued an OPEN or at (@) command, but a channel is not available. Use the CLOSE command to close one or more files or use the EXIT command to unnest a level and try again.
?No default (Print,Batch) queue You did not specify a queue name with the PRINT or SUBMIT command, and no default queue exists.
?No default print queue for log file The /LOG_QUEUE qualifier in a SUBMIT command was specified without an explicit queue name; however, no default print queue exists. Use an explicit queue name for the log file.
?No file name or type permitted A device:ppn syntax was expected and you supplied a full file specification which included either a file name or type.
?No logins This message can be displayed when you try to log in, for one of two reasons: • The system is full, so it cannot accept additional users. • The system manager has disabled logins. (Possibly, logins are disabled because the system will be shut down shortly.)
?Non-executable file This error occurs if the file you are trying to run is a source file; for example, .BAS. You need to compile and link the file before you run it. (Note that an executable file includes the value 64 in its protection code.)
?Non-printable character You typed a control character.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
??Non-res run-time system This message generally indicates hardware problems. For example, the run-time system referenced has not been loaded into memory and cannot be loaded for some reason, and is therefore nonresident.
%No owner rights to (print,batch) entry <entry-spec> The entry shown is not your own and you do not have sufficient privilege to delete or modify it.
?NO prefix not allowed You used the /NO prefix improperly.
%No (Print,Batch) entry <entry-spec> found No entries matching the entry-spec you specified in a SHOW ENTRY or SET ENTRY command were found.
%No (Print,Batch) queue <queue-name> found No queues matching the queue name you specified in a SHOW QUEUE or SET QUEUE command were found.
%No (Print,Batch) server <server-name> found No servers matching the server name you specified in a SHOW SERVER or SET SERVER command were found.
?No qualifiers allowed The user specified a qualifier on a command that does not allow qualifiers.
?No (read,write,or read/write) access to file <file-spec> [by owner] The file specifications in a PRINT or SUBMIT command requires access that you or the specified owner does not have.
?No room for user on device In attempting to create a file on a device, or write information to a device, you exceeded some size limit. If it was a nondisk device, this error indicates that the device was full. If it was a disk device, this error indicates one of three conditions: • The disk as a whole is full. • You attempted to create a contiguous file, and there is not enough space on the disk. (However, the /CONTIGUOUS qualifier for the commands COPY and CREATE produce only a warning if there is not enough contiguous space.) If you attempted to create a contiguous file but were unable to, try to create a noncontiguous file of the same size. If you are able to create a noncontiguous file then you can conclude the problem is lack of contiguous space. • If you have eliminated the first two conditions, then the disk directory has reached its capacity. This capacity is independent of your disk quota. The number varies, depending on the directory cluster size that the system manager assigned when creating the directory and the sizes of the files in it. Large files fill up a directory faster than small files, especially if the large files have small cluster sizes. Also, if the directory itself has a large cluster size, it can hold more files and larger files.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?No run-time system This error can occur if the program you are trying to run requires a run-time system that is not installed.
?Not a valid device The device name that you gave is invalid for any of the following reasons: • It is not assigned as a system-wide or user logical name. • It is not a physical device name of any device that is installed on your system. • The device name is valid, but not with this command. For example, "INITIALIZE TT:" (you cannot initialize a terminal).
?Not enough available memory An attempt was made to load a executable program that is too large to run, given the job's private maximum memory size. Either the program must be allowed to expand above a private maximum memory size, or the system manager must increase the job's private memory size maximum to accommodate the program.
?Number not in range <low> or <high> You typed a number where one is allowed, but the number is not in the valid range. The valid range is from <low> to <high>.
?Number too big You typed a number where one is allowed, but the number is too large. Refer to the command description to find out the largest acceptable value.
?Number too small You typed a number where one is allowed, but the number is too small. Refer to the command description to find out the smallest acceptable value.
?Pack-id labels don't match The identification code for the specified disk pack does not match the identification code already on the pack. This message can occur when you try to mount or dismount a disk.
?Page limit exceeds queue's maximum You specified a /PAGE_LIMIT value in a PRINT command larger than the maximum allowed for the queue.
?Parameter or argument too long This message occurs if a file specification, text string, or qualifier argument exceeds 127 characters.
?PPN does not exist The project-programmer number (PPN) you specified as part of the job specification does not exist.
?PPN needed The command you typed requires that a PPN be specified.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
Previous logical name assignment replaced An informational message to indicate that an ASSIGN command redefined a previously defined user logical.
(Print,Batch) entry <entry-number> <entry-spec> created Acknowledgment message to PRINT or SUBMIT command.
(Print,Batch) entry <entry-spec> - <error text> For SET ENTRY, indicates that an entry was not modified for the reason stated in the error text.
?Print/Batch Services already started You issued the START/QUEUE/MANAGER command; however, the Print/Batch Services (PBS) package is already started.
?Print/Batch services not running You issued a command that requires action by PBS; however, the package is not running.
(Print,Batch) queue <queue-name> assigned to server <server-name> Acknowledgment message for ASSIGN/QUEUE command.
(Print,Batch) queue <queue-name> deassigned from server <server-name> Acknowledgment message for DEASSIGN/QUEUE command.
(Print,Batch) queue <queue-name> closed Acknowledgment message for CLOSE/QUEUE command.
(Print,Batch) queue <queue-name> deleted Acknowledgment message for DELETE/QUEUE command.
%(Print,Batch) queue <queue-name> - <error text> For SET QUEUE, indicates that a queue was not modified for the reason stated in the error text.
(Print,Batch) queue <queue-name> marked for deletion Acknowledgment message for DELETE/QUEUE command when the queue you specified still has entries on it. The queue is deleted as soon as it becomes empty.
(Print,Batch) queue <queue-name> opened Acknowledgment message for OPEN/QUEUE command.
(Print,Batch) queue <queue-name> started Acknowledgment message for START/QUEUE command.
(Print,Batch) queue <queue-name> stopped Acknowledgment message for STOP/QUEUE command.
(Print,Batch) server <server-name> (deleted,marked for deletion) Acknowledgment message for DELETE/SERVER command.
%(Print,Batch) server <server-name> - <error text> For SET SERVER, indicates that a server was not modified for the reason stated in the error text.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
(Print,Batch) server <server-name> initialized [non-]shareable [with forms <form-name>] Acknowledgment message for INITIALIZE/SERVER command.
(Print,Batch) server <server-name> modified Acknowledgment message for SET SERVER command.
(Print,Batch) server <server-name> started Acknowledgment message for START/SERVER command.
(Print,Batch) server <server-name> stopped Acknowledgment message for STOP/SERVER command.
Print/Batch Services started at <time> Acknowledgment message when you issue the START/QUEUE/MANAGER command to begin Print/Batch Services.
Print/Batch Services stopped at <time> Acknowledgment message for STOP/QUEUE/MANAGER command if no jobs are being processed.
Print/Batch Services will stop after (completing,aborting) <n> job[s] Acknowledgment message for STOP/QUEUE/MANAGER command if any jobs are currently in progress.
?Priority exceeds queue's maximum You specified a priority for a PRINT or SUBMIT entry larger than the maximum allowed for the queue.
?<privilege> privilege required You typed a command that requires some privilege, and you do not have it.
??Program failure in <program-name> This message reports a problem in the software. It is followed on the next line by an explanation of the problem. You should verify that the failing program is correctly installed. If necessary, you should then submit an SPR. The SPR should show the dialogue that preceded the message, the exact text of the message, and a list of patches that have been installed in the failing program.
?Program PBS$:PBS.TSK does not exist The Print/Batch Services (PBS) program was not found when you issued the START/QUEUE/MANAGER command.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?Protection violation This error can occur for reasons similar to the following: • You typed a CCL command that your system manager has protected against general use. • You tried to run a program to which you do not have execute access. • You tried to read a file to which you do not have read access. • You tried to write to or delete a file to which you do not have write access. If this message occurs because of a protection violation with one of your own files, you can use the SET PROTECTION command to change the file's protection code. (However, this error can also occur because of other conditions.)
%Public disk mounted as private A user with MOUNT privilege mounted a disk initialized as public without specifying /PRIVATE, /PUBLIC, /SHARE, or /NOSHARE. The system mounts the disk as private.
?Qualifier conflicts with file type You specified the /APPEND qualifier and the file has RMS attributes.
?Qualifier conflicts with parameter You typed a parameter and a qualifier that should not be present together. For example, with the DEASSIGN command, you specified the /ALL qualifier and a logical name.
?Queue already assigned to server You issued an ASSIGN/QUEUE for a queue already assigned to the server you specified. Use the SHOW QUEUE/FULL command to see which servers are assigned to a queue.
?Queue does not exist You specified the name of a queue that does not exist.
?Queue is (closed,deleted) A PRINT or SUBMIT command entry could not be created on a queue because the queue is closed or marked for deletion. Use a different queue.
?Queue is privileged The queue you specified in a PRINT or SUBMIT command requires one or more privileges that you do not have. Use the SHOW QUEUE/FULL command to display the privileges that the queue requires.
?Queue manager message queue full There is no room for additional messages to be sent to PBS at this time. Reissue the request.
?Quoted string expected A quoted string or character was expected and was either missing or was not enclosed in quotes.
%Record too long, line truncated The record on a READ command was too long. The assignment occurred; however, only the first 255 characters were processed.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?Reserved symbol name The user attempted to define a local or global symbol which began with a reserved prefix of $, F$ or f$, or delete a reserved global symbol from the symbol table that began with $.
?Server already exists You issued the INITIALIZE/SERVER command to define a server that already exists.
?Server does not exist You specified the name of a print or batch server that does not exist. Use the SHOW SERVER command to list the defined servers.
?Single character expected A single character inside quotes was expected and more than one character was supplied (for example, SET DATA/END_OF_DATA = "$$").
?Speed is not allowed for <terminal> The speed specified is not one of the speeds allowed for this terminal in the TERSPD.SYS file.
??Stack overflow This message indicates a system problem. The system manager should send in an SPR, giving the dialogue that preceded the message, the text of the message, and a list of patches that have been installed.
?String too long While in an expression, a string became too long to fit in the 255 available bytes.
?Substitution too complex The user's request for apostrophe substitution was rejected because the substitution was too complex (it reached the maximum number of substitution iterations allowed on one command).
?Symbol name conflicts with <symbol-name> The user specified an assignment whose symbol name or abbreviation point conflicted with an existing symbol definition in the same symbol table.
?Symbol name too long The symbol name specified by the user exceeded 255 characters.
%Symbol table almost full This warning is issued when your local or global symbol table has less than 100 free bytes.
??Symbol table full You attempted to define a label or symbol when the local or global symbol table was already filled up. Note that if you are in a command procedure when your symbol table gets full, the command procedure aborts.
?Syntax error The command has improper syntax. This occurs when there is not a more specific message describing the syntax error.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?THEN clause required The action clause was missing on the IF <expression> THEN command or on the ON <severity-level> THEN command.
?THEN keyword required The THEN keyword was missing on the IF <expression> THEN command or on the ON <severity-level> THEN command.
?Time limit exceeds queue's maximum You specified a /TIME_LIMIT value in a SUBMIT command larger than the maximum allowed for the queue.
?Too many arguments You used the notation /qualifier=(arg,arg,...) with a qualifier that accepts only a single argument.
?Too many elements in list In a list of file specifications or other items (separated by commas or plus signs), you indicated more file specifications than are allowed. For example, you exceeded one of the following limits: • The DIBOL, RENAME, DELETE, and SET PROTECTION commands allow six file specifications. • The COPY command allows six input file specifications and one output file specification. • The PRINT and SUBMIT commands allow up to 11 file specifications.
?Too many files or parameters You specified too many file specifications in a PRINT or SUBMIT command, or a parameter string in a SUBMIT command is too long to be handled by PBS. Reduce the number of file specifications or use a shorter parameter string.
?Too many logical names assigned With the ASSIGN command, you exceeded the maximum number of logical names. You can only assign up to four logical names (only three logical names if any of the logical assignments includes a PPN).
?Too many open files on unit You specified the same magnetic tape or DECtape drive both as input and output files on COPY or APPEND.
?Too many parameters This message occurs if you specify more command parameters than the command can accept. For example, you specified more than eight parameters with the /PARAMETERS qualifier.
?Too many printers initialized The device you specified could not be initialized because the maximum number of spooling devices has already been initialized.
?Unable to copy tape command file to disk The user specified an indirect command file on magnetic tape, and some error occurred when attempting to copy it to a temporary disk file for processing.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
??Unable to create DCL work file DCL's work file for storing the user's symbol tables could not be created on disk. The user may be over quota. Corrective action should be taken before the user continues processing.
?Unable to read DCL work file The user's symbol tables could not be read in from the DCL work file. If RSTS/E is processing an indirect command file, then control immediately returns to the interactive level.
?Unable to start Print/Batch Services <error message> Some external condition, as the error message on the second line describes, prevented PBS from being started.
??Unable to write DCL work file The user's symbol tables could not be written out to the DCL work file. The user may be over quota. Corrective action should be taken before the user continues processing. If RSTS/E is processing an indirect command file, then control immediately returns to the interactive level.
?Unbalanced parentheses Parentheses do not match up (for example, A=((5)+6).
?Undefined label <label> The label specified on the GOTO command does not exist in the indirect command procedure being executed.
?Undefined symbol The symbol name specified is not defined in the global or local symbol tables.
?Unexpected character A character was encountered in a qualifier or a parameter that was not expected. For example, in SET TERMINAL/WIDTH=80FOO, the "F" of "FOO" would be an unexpected character.
?Unit number required This message occurs when you specify a device-name without a device-number (for example, DM: instead of DM0:).
?User macros nested too deep There were more than 50 nested user macros or a user macro called itself.
?Wildcard entry name not allowed The command you specified does not allow wildcard entry names. Specify the name of a single entry instead.
?Wildcards not allowed You included a wildcard in a file specification, where wildcards are not allowed.
?Wildcard PPN not allowed The command you specified does not allow wildcard PPNs. Specify the name of a single PPN instead.

(continued on next page)

Table A-1: DCL Error Messages (Cont.)

Message and Meaning
?Wildcard queue name not allowed The command you specified does not allow wildcard queue names. Specify the name of a single queue instead.
?You need MOUNT privilege to dismount a public disk This message occurs when a user without MOUNT privilege tries to dismount a disk initialized as public. A user without MOUNT privilege can dismount only private disks.
?You need MOUNT privilege to mount a public disk This message occurs when a user without MOUNT privilege tries to mount a disk initialized as public. A user without MOUNT privilege can mount only private disks.
?You must have MOUNT privilege to rebuild the disk This message occurs when a user without MOUNT privilege attempts to rebuild a private disk with the /REBUILD qualifier of the MOUNT command. Have your system manager rebuild the disk.

More About Command Environments and RSTS/E File Specification

B

This appendix provides information primarily for users who are familiar with earlier versions (V8.0 and before) of RSTS/E, especially those users who work mostly within command environments other than DCL.

More About Command Environments

Chapter 1 discusses the four RSTS/E command environments: DCL, BASIC-PLUS, RT11, and RSX. Each offers its own advantages. Each command environment has its own prompt:

Prompt	Environment
$	DIGITAL Command Language (DCL) environment
Ready	BASIC-PLUS environment
.	RT11 environment
>	RSX environment

Since DCL is the default command environment on RSTS/E, this manual has already described many DCL commands to help you in your work. If you want to switch to one of the other command environments, see the appropriate documentation for complete descriptions of the commands you can use in BASIC-PLUS, RT11, and RSX.

Switching Between Command Environments

DCL is the default RSTS/E command environment. You can switch to another, however. Use the following command to switch to and from any command environment:

```
$ SET JOB/KEYBOARD_MONITOR=[command environment name]
```

The command environment name can be any one of the following:

- DCL
- BASIC
- RT11
- RSX

The system transfers control to that keyboard monitor, establishing it as your "job keyboard monitor." That is, you can type commands to the keyboard monitor you switched to until you switch again, or log off the system.

Note

You can issue DCL commands from any keyboard monitor by typing the $ character followed by the command. For example, if you are in the BASIC-PLUS command environment and you want to use the DCL SHOW USER command, type:

```
$ SHOW USER
```

DCL will execute the SHOW USER command, then automatically return to the BASIC-PLUS environment.

Note that if you type the $ character with no command, you will switch back into the DCL command environment.

BASIC-PLUS Keyboard Monitor Commands

The BASIC-PLUS command environment provides commands useful to the BASIC-PLUS programmer on RSTS/E systems. The BASIC-PLUS interpreter lets you type in a program using BASIC-PLUS statements that are translated and stored in your job area.

Many of the BASIC-PLUS commands (which are distinct from language statements) let you list, compile, store, and otherwise work with the BASIC-PLUS program you create. Note, however, that BASIC-PLUS has a very powerful feature, called "immediate mode" that allows you to type many language statements as though they were commands.

Table B-1 briefly describes BASIC-PLUS commands. See the *BASIC-PLUS Language Manual* for a complete description of BASIC-PLUS commands.

Table B-1: BASIC-PLUS Commands

Command	Description
APPEND	Includes the contents of a previously saved source program into the current program.
CAT or CATALOG	Displays the file directory for your account. Unless another device is specified following the term CAT or CATALOG, the public disk structure is the assumed device.
CCONT	Can be used by privileged users only. The command operates the same as the CONT command, but detaches the job from the terminal.
COMPILE	Allows you to store a translated version of your BASIC program. The file is stored on disk with the current name and the file type .BAC. Or, you can specify a new file name and type.
CONT	Allows you to continue execution of the program currently in memory following the execution of a STOP statement.
DELETE	Allows you to remove one or more lines from the program currently in memory. After the word DELETE, type the line number of the single line to be deleted or two line numbers separated by a hyphen (-) indicating the first and last line of the section of code to be removed. Several single lines or line sections can be indicated by separating the line numbers, or line number pairs, with a comma.
EXIT	Returns you to the job keyboard monitor.
KEY	Reenables the echo feature on the terminal after a TAPE command. Enter with the LINE FEED or ESCAPE key.
LENGTH	Displays the length of the current program in memory, in 1K increments.
LIST	Displays the program currently in memory at your terminal. You can list the entire program currently in memory, or one or more lines of that program.
LISTNH	Does the same thing as LIST, except that it does not print a header containing the program name and current date.
NEW	Clears your area in memory and allows you to input a new program from the terminal. You indicate a program name following NEW or when the system requests it.
OLD	Clears your area in memory and allows you to recall a saved program from a storage device. You indicate a program name following the word OLD or when BASIC-PLUS requests it.
RENAME	Lets you change the name of the program currently in memory.
RUN	Lets you begin execution of the program currently in memory. Or, if the RUN command is followed by a file specification, including device and account if desired, you can run any program.
RUNNH	Executes the program currently in memory, but does not print header information with the program name and current date.
SAVE	Saves the source program currently in memory in a file on the system disk, under its current name with the file type of .BAS. Or, you can give a complete file specification (including device and account).

(continued on next page)

Table B-1: BASIC-PLUS Commands (Cont.)

Command	Description
SCALE	Sets the scale factor to a designated value or displays the values currently in effect if no value is designated.
TAPE	Disables the echo feature on the terminal; this is useful if you are using a terminal with a paper-tape reader.
UNSAVE	Deletes the file specified.

RT11 Keyboard Monitor Commands

The RT11 Run-Time System has a keyboard monitor. If you switch control to RT11, a period (.) prompt displays, and you can type commands. You can use the standard RSTS/E commands ASSIGN, BYE, DEASSIGN, HELLO, and MOUNT.

Table B-2 briefly describes RT11 commands.

Table B-2: RT11 Commands

Command	Description
B/E/D	The commands B, E, and D let you examine memory locations, as follows: • B <address> — Sets a base address from which offsets in E and D commands are computed. Initially, the base is 0. • E <offset> — Prints the contents of location <address> + <offset>. • D <offset> <value> — Replaces the contents of location <address-> + <offset> with <value>. Any address from 0 to 177777 is valid for these commands; however, addresses outside your current job image area (as determined by the SIZE command) will give an error.
CCO[NTINUE]	Resumes the execution, as a detached job, of a program that was stopped by a CTRL/C. If, for any reason, the program cannot be restarted, the error message ?Can't continue displays on the terminal.
CO[NTINUE]	Resumes execution of a program stopped by a CTRL/C. If, for any reason, the program cannot be restarted, the error message ?Can't continue displays on the terminal.
CL[OSE]	Closes all open channels. Tentative files are made permanent.
DATE	Displays the current date on the job's terminal.
ER[ROR]	Prints the RSTS/E error message text associated with the last encountered error, if there was one. For example, any failure on file lookup gives an error message similar to ?Fil not fnd?. Typing CTRL/C to return to RT11's period prompt (.) and then typing ERR will print the full RSTS/E error message for the error encountered; for example, ?Can't find file or account.

(continued on next page)

Table B-2: RT11 Commands (Cont.)

Command	Description
GE[T]	Loads an RT11-executable file (with a file type of .SAV) into memory, but does not execute it. (The START command is used to run programs loaded with GET.) No automatic expansion of the job image area occurs with GET; you must use SIZE if necessary. Loading for GET differs from R or RUN in that certain low-core areas are preserved rather than loaded from block 0 of the .SAV file.
IN[ITIALIZE]	Resets all RT11 conditions, except for logical assignments, to the state they are in when RT11 is first entered as the job keyboard monitor.
LOCK	Denies access to a disk pack by users with insufficient privileges. The format is: LOCK dev: where dev: is the device designation for the disk.
MO[NITOR]	Exits to the job keyboard monitor. This will be the default keyboard monitor unless you have run the SWITCH utility to set a job keyboard monitor.
R	Runs a program from the system library account. The file can be associated with any run-time system. For example, R LINK has the same effect as RUN $LINK. The R command will automatically expand the user job image area, if necessary.
RE[ENTER]	Reenters or restarts a program already in memory whose execution has been halted. Bit 13 in the Job Status Word (see the *RSTS/E System Directives Manual*) must be set to allow a restart. If it is set, execution begins at the address specified by the contents of word 40 in the low 1000 bytes of virtual memory. All channels are reset (see .SRESET directive, *RSTS/E System Directives Manual*) unless the chain bit (bit 8 of the Job Status Word) is set. If restart is not possible, the message ?No restart is displayed on the terminal.
RN	Loads the file named (from the user's account if no project-programmer number is given) and begins execution. The program may be associated with any run-time system. If necessary, the user job image area is automatically expanded for the program to be loaded and run. The RN and R commands are identical except for the accounts searched for the file when no explicit project-programmer number is given.
RU	RU is the same as a GET followed by a START. You must use SIZE to expand the user job image area, if necessary; no automatic expansion occurs with RU.
RUN	This command is passed on by the RT11 run-time system to the RSTS/E monitor — it works the same in any command environment.
SH[UTUP]	If (1) only one job is running on the system, (2) logins are disabled, (3) no disks except the system disk are mounted, and (4) no files are open on the system disk, then the SHUTUP command logs the current job off the system and bootstraps the initialization code after the job is logged off.

(continued on next page)

Table B-2: RT11 Commands (Cont.)

Command	Description
SI[ZE]	Sets the size available for an RT11 program, in K words. When the RT11 run-time system is entered, the size is set to 2K words. To see the current size, type SIZE without a value. You must use SIZE to insure enough memory before typing a GET or RU command. The job image area is automatically expanded for R and RN commands, from information carried in the file itself. However, a size given with SIZE overrides any computed value (for R and RN) if the computed value is smaller than the SIZE value.
ST[ART]	Starts a program loaded with GET. The format of the command is: ST[ART] [addr] where addr is an even octal number specifying the address where program execution is to begin. If addr is omitted or 0, execution begins with the address in location 40 in the low 1000 bytes of memory.
TI[ME]	Displays the current time of day at the job's terminal.
UNLOCK	Unlocks a disk pack, so that programs running under nonprivileged accounts can open files on the disk. The format is: UNLOCK dev: where dev: is the device designation of the disk to be unlocked.
VE[RSION]	Prints the current emulator version number.

RSX Keyboard Monitor Commands

The RSX run-time system has a keyboard monitor. If you switch control to RSX, an angle bracket (>) prompt is displayed. You can use the standard RSTS/E commands ASSIGN, BYE, DEASSIGN, HELLO, and MOUNT.

Table B-3 briefly describes RSX commands.

Table B-3: RSX Commands

Command	Description
DISMOUNT	Prepares a device for dismounting, so that you can remove a disk pack of magnetic tape from a device. The format of the command is: DISMOUNT dev:
UNSAVE	Deletes a specifically named file from your account. The format of the command is: UNSAVE filename.type
SHUTUP	Shuts down the system if no other jobs are running on the system, all disks are dismounted, all other run-time systems are removed, and no logins are allowed.

More About RSTS/E File Specification

Many commands in the environments discussed in the previous section require that you identify a file to serve as either input or output to the command.

This section provides further information on specifying RSTS/E files in these command environments.

The general form for a RSTS/E file specification is:

device:[project,programmer]filename.type/switches

The following section discusses only the /switch element in the RSTS/E file specification. See Chapter 1 for a description of the other elements of specifying files.

RSTS/E File Specification Switch

A file specification switch may be included as the final element of the specification string. You may specify the following six switch options:

- /PROTECT — protection to be given to the file
- /FILESIZE — disk size (in blocks) to which the file is pre-extended
- /CLUSTERSIZE — minimum number of contiguous disk blocks forming a cluster
- /POSITION — specific block number on disk at which the file is to be placed
- /MODE — read/write mode in which the file's data is passed to the device driver
- /RONLY — read-only mode value for a disk file

If a file specification switch is not at the end of the file specification, is missing a colon, or is not a valid form, the system prints the error message ?Illegal switch usage. For example, either of the following specifications will produce the switch usage error message:

- ABC/SI:100 [1,2]
- ABC/SIQ

If an argument is missing, or contains an illegal character, the system generates an ?Illegal number error message.

/PROTECT Switch

The /PROTECT switch establishes a protection code for the file. The /PROTECT switch consists of:

- A slash (/)
- PROTECT (or a minimum abbreviation of PROT)
- A colon (:)
- A number specifying the protection code. Use a number sign (#) before octal numbers, and a decimal point (.) to terminate decimal numbers.

The protection code is a string of one to three digits. This string determines the file's degree of protection on two levels: the actions — reading, writing, and deleting — against which it is protected, and the user or class of users against whom it is protected.

See Chapter 3 for a complete description of how to specify protection codes.

/FILESIZE Switch

The /FILESIZE switch allows the creation of a disk file of the specified size, in blocks, before any read/write operations are performed. Thus, /FILESIZE reserves space on the disk for data to be placed in the file. This switch consists of:

- A slash (/)
- FILESIZE or SIZE (or a minimum abbreviation of FI or SI)
- A colon (:)
- An optional number sign (#) for octal conversion of the argument n
- The argument n, a decimal number which indicates the number of blocks in the pre-extended file
- An optional trailing decimal point (.) to ensure that n is interpreted as a decimal number.

The argument n specifies the length in disk blocks to which the file is pre-extended. That is, it reserves a specified portion of disk space for the file. The value of the /FILESIZE argument is dependent on the type of file. If the file's protection marks it as executable, the argument n must be in the range of 0 to 65535. If the capability to create large files is present on your system and the file is not marked as executable, the argument n can be in the range of 0 to 223-1 (assuming that the disk is large enough). Note that you cannot run an executable file that is larger than 65535 blocks. An attempt to pre-extend such a file beyond block 65535 returns a ?Protection violation error message.

The /FILESIZE switch may be minimally abbreviated to /FI or to /SI. The following list shows its valid forms:

/FI:[#]n[.]
/FIL:[#]n[.]
/FILE:[#]n[.]
.
.
.
/FILESIZE:[#]n[.]

or

/SI:[#]n[.]
.
.
.
/SIZE:[#]n[.]

/CLUSTERSIZE Switch

The /CLUSTERSIZE switch establishes the minimum cluster size for a disk file; a cluster is a number of contiguous blocks taken together as a unit. The /CLUSTERIZE switch is especially useful for large files; specifying a large cluster size speeds up random access to the data and prevents such files from crowding or filling your directory, whose size is limited. RSTS/E permits cluster sizes of 1, 2, 4, 8, 16, 32, 64, 128, or 256 blocks.

The /CLUSTERSIZE switch consists of:

- A slash (/)
- CLUSTERSIZE (or a minimum abbreviation of CL)
- A colon (:)
- An optional minus sign (-) to specify a negative cluster size (see the next paragraph)
- An optional number sign (#) for octal interpretation of the argument n
- The argument n specifying the cluster size in blocks
- A decimal point (.) to ensure decimal interpretation of n

The following list shows the valid forms of the option:

/CL:[-] [#]n[.]
/CLU:[-] [#]n[.]
/CLUS:[-] [#]n[.]
.
.
.
/CLUSTERSIZE:[-] [#]n[.]

Specifying a negative cluster size avoids certain errors associated with disk devices. A negative cluster size causes the system to use the absolute value of the cluster size, if the device on which the file is created allows that value. If the absolute value is less than the device cluster size at which the file is to be created, the system uses the device cluster size instead of returning an error.

/POSITION Switch

The /POSITION switch allows you to specify the location of a file on disk. The /POSITION switch consists of:

- A slash (/)
- POSITION (or a minimum abbreviation of PO)
- A colon (:)
- The argument n, a decimal number that specifies the desired placement of the file

The argument n indicates a device cluster number on the disk. The device cluster numbers vary from disk to disk; see the *RSTS/E System Generation Manual* for a table of device sizes.

If the value of n in /POSITION:n is 0, no placement is performed and the system determines the location of the file on the disk. If the value of n is a nonzero number, the system attempts to place the first block of the file at the specified device cluster number on the disk. If the file cannot be placed at that location, the system places the first block of the file at the first free cluster number that is greater than the specification.

When a file is placed, it is marked as such by the monitor. Note that no error is returned if the system is unable to place the file at the specified location. To determine the actual location of the file on disk, use the /PO, /FU, or /S options of the DIRECT program. For example:

```
DIR /PO
SY:[2,211]
RMSTST.B2S    18698
RMSTST.OBJ    18720
RMSTST.MAP    18723
CHTST .BAS     6020
CHTST1.TSK    16562
```

The following list shows the valid forms of the POSITION option:

/PO:n
/POS:n
/POSI:n
.
.
.
/POSITION:n

Note that the position of a file on disk may change if you backup and restore the file with the BACKUP program.

/MODE and /RONLY Switches

The /MODE switch enables the passing of up to 16 (decimal) bits of information to the device driver at file open time. The meaning of these bits (if any) is device dependent, and determines the read/write mode for data transfer. For explanations of the bits, see the *RSTS/E Programming Manual.*

The /MODE switch consists of:

- A slash (/)
- MODE (or a minimum abbreviation of MO)
- A colon (:)
- An optional number sign (#) for octal interpretation of the argument n
- The argument n specifying a mode setting between 0 and 32767 (decimal) inclusive
- An optional decimal point (.) to ensure decimal interpretation of n

The following list shows the valid forms of the option:

/MO:[#]n[.]
/MOD:[#]n[.]
/MODE:[#]n[.]

The /RONLY switch enables setting of the read-only MODE value for a disk file. The /RONLY switch consists of:

- A slash (/)
- RONLY (or a minimum abbreviation of RO)

The following list shows the valid forms of the option:

/RO
/RON
/RONL
/RONLY

Glossary

Absolute Time

A specific date or time provided in a command line. (Compare with Real Time.)

Account

What you log into with a Project-Programmer Number (PPN) and a password on a RSTS/E system.

Account Number

A number that identifies the project group and the programmer. It is also known as the Project-Programmer Number (PPN, such as [3,25]). You need an account number (which new users receive from the system manager) to log in to the system. The system uses the account to charge you for computer time and arrange, find, and protect your stored files.

Alphanumeric

A contraction of alphabetic-numeric; the set of characters that compose text. Characters include letters and numerals, and exclude special characters.

ANSI

American National Standards Institute. Creates standards to ensure consistency in all aspects of computer technology: command languages, keyboards, codes, and so forth.

ASCII code

American Standard Code for Information Interchange, a standard 7-bit code representing the 128 characters in which textual information is recorded.

Batch Processing

A way of scheduling programs that allows you to enter jobs to be run without your input at a terminal. Batch processing can be used, for example, in data processing operations that need no interaction.

Baud Rate

The speed at which information is transferred between devices on a system.

Bit

Contraction of "binary digit." A bit is the smallest unit of information in a binary system of notation.

Block

A set of consecutive machine words, characters, or digits handled as a unit, particularly in input and output operations. On the RSTS/E system, a block is equal to 256 16-bit words or 512 8-bit bytes.

BPI

Bits per inch; a measure of tape density. Common densities for magnetic tapes are 800 or 1600 bpi.

Buffer

A temporary storage area used to contain data. Buffers hold data being passed between processes or devices that operate at different speeds or times.

Byte

A group of eight binary digits (bits) processed as a unit. Each ASCII character is stored in one byte.

CCL

See Concise Command Language.

Central Processing Unit (CPU)

The portion of a computer system that controls the interpretation and execution of instructions.

Channel

A logical concept that helps you manage the flow of data to and from a running program.

Character

An element in a set of symbols. A human-readable symbol can be a letter from A to Z, a number from 0 to 9, or a special symbol. A machine intelligible symbol is made of a group of binary digits. Some machine-readable symbols are "non-printing" characters (such as tabs or control characters), because they cannot be displayed on a line printer listing or a video screen.

Cluster Size

The unit of size in which files are allocated. Cluster size is measured in blocks.

Command

A user-initiated order to a computer system that causes the performance of a predefined operation. A command is the part of an instruction that specifies the operation to be performed.

Command environment

An interface through which you can communicate with the computer. DCL and BASIC-PLUS are examples of command environments.

Command level

The part of a session at a terminal when the system awaits your input. For example, you can recognize DCL command level by its dollar sign ($) prompt.

Command line/command string

A line (or set of continued lines), normally terminated by pressing the RETURN key. A command line or string contains a command and (optionally) information modifying the command. The full form of the command string contains a command, its qualifiers, and its parameters (file specifications, for example) and their qualifiers.

Compile

To translate a source program (one written by a user) into machine language.

Concatenate

To unite in a series; to link together many items into one.

Concise Command Language (CCL)

A way to call a system program and provide it with input on a single command line. Many system programs (such as PIP on RSTS/E), use the name of the program as the command name, followed by the command line required by that program. CCLs can be different for each RSTS/E system.

CPU

See Central Processing Unit.

Crash

An unexpected computer system shut-down, caused by problems such as power failures.

Creation date

The date on which a file on RSTS/E is created or updated.

CRT

Cathode Ray Tube. A CRT is used in video display terminals, such as VT220s and VT100s. Sometimes video terminals are called "CRT devices."

CTRL (control key)

The keyboard character that causes a control action. It is used in combination with an alphabetic key. For example, if you hold down the control key and press the "U" key on a RSTS/E system, the system responds by deleting characters from the cursor to the beginning of the line.

DIGITAL Command Language (DCL)

The default command language on RSTS/E systems.

DECnet

A family of hardware/software products that creates distributed networks from DIGITAL computers and their interconnecting devices. DECnet allows you to do such operations as copying files from one computer (node) to another, and connecting your terminal to a remote computer.

Default

To omit information in a computer operation and let the computer make a predefined assumption about the operation; the assumption made by a program when you do not provide a value.

Density

The number of bits per inch (bpi) that can be stored on a magnetic tape.

Device

A peripheral hardware unit used to perform input and output operations.

Dial-Up Line

A communications circuit that connects a terminal and a computer over a telephone line.

Disk

A mass storage device that holds data on rotating magnetic platters.

Disk Quota

The capacity of a disk to store information. Each user of a computer system is allowed a finite amount of storage space. On RSTS/E, the system notifies you if you try to exceed your disk quota.

Display

The printing of text at a hard-copy terminal, or the output of text at a video screen.

DOS

Disk Operating System. DOS is used primarily to refer to a tape format.

Echo

The display of characters you type at a terminal. When you log in to a RSTS/E system, for example, generally the PPN is echoed, but the password is not.

Editing Session

The time you spend at a terminal using a text editor.

Editor

A program that allows you to modify data, programs, or files.

Elapsed Time

The amount of time you are logged in to a computer system.

Error Message

A notice from the computer that indicates an error and usually contains recovery information; a program message indicating the presence of a mistake. For example, RSTS/E displays an error message if you type in data that it cannot process.

Field

One or more characters treated as a unit; a specified area of a record used for a single type of data. For example, a file name is occasionally referred to as the "name field" of a complete file specification.

File

An ordered collection of stored text or data; a collection of related information. For example, you could create files on RSTS/E to contain data for a program or text for a report.

File Specification

The name used to identify the node, device, directory name, file name, and file type under which you store a file.

File Type

The part of a file specification that follows the file name. A file type (sometimes called an "extension" on RSTS/E) consists of a period (.) followed by up to three alphanumeric characters.

Hard copy

Computer output printed on paper, such as from a line printer or a hard-copy terminal.

Hardware

The physical equipment or machinery of a computer system. Devices are examples of hardware. (Compare with Software.)

Help Facility

A set of online messages that describes the functions of available programs on a system.

Host

The computer system that processes your commands and responds to them. The network node that you are currently logged in to.

Input

Information that is transmitted to a computer; information to be processed by the computer; a device involved in gathering data. For example, when you type in a program at a terminal that is connected to the computer, you are providing input. Among other things, the terminal is serving as an input device.

Installation

The location of a computer system. For example, this manual refers to your installation as the computer system at your site.

Interactive

The process of communication between a person and a computer system. For example, the computer prompts you for input and you type in a response.

Interface

A shared boundary between two elements of a computer system, such as between two programs or devices.

Job

Everything a user does from beginning to end of a terminal session, for example, commands, programs, processing, or output.

Job Number

The number RSTS/E assigns to your job when you log in. Your job number is deleted when you log out.

Journaling

The recording of user input during an editing session. For example, if you edit a file using EDT and the system crashes, journaling allows you to recover your work when the system resumes operation.

Keyboard

The set of keys that you use to type at a terminal to provide input to the computer.

Keyboard Monitor

A part of the system software that provides communication between a person at a keyboard and the operating system. The keyboard monitor interprets the commands you type at the keyboard.

Keyword

A word that is an integral part of a command or qualifier.

Link

To combine or connect two independent entities; a reference to another element in a set of elements. Linking is generally used to mean combining two or more object files into an executable file.

Literal

Characters that are interpreted literally. For example, in BASIC-PLUS programming, you can enclose a character string in quotation marks so that the characters are used as literals.

Local Node

The node a user first logs in to on the system. The local node is generally used in discussions of network operations. (Compare with Remote Node.)

Logical Name

A name that a user or system manager can assign to a physical device or to a PPN. (Compare with Physical Name.)

Log In

The process of gaining access to a computer system.

Log Out

The process of ending a session with the computer.

Magnetic Tape

A plastic tape coated with magnetic material on which data is recorded in narrow tracks.

Memory

Internal computer storage. Memory is also a device on which data can be stored and from which it can be accessed.

Merge

In files, to join the contents of one file to the contents of another file.

Microsecond

One millionth of a second.

Millisecond

One thousandth of a second.

Mnemonic

An aid to memory, as in mnemonic file names, where the name of the file reminds you of the contents of the file.

Mode

A state of being, such as "interactive mode," in which the system or a program prompts you for your input.

Network

A set of interconnected computer systems.

Node

An individual computer system in a network that can communicate with other computer systems in the network.

Non-File-Structured Device

A device that does not separate data into distinct files and does not maintain directories.

Null

An element that is explicitly left blank. For example, a null file type consists of no alphanumeric characters. In other words, the file specification includes a file type, which is null.

Operating System

Software that controls the execution of computer programs and performs system functions. An operating system is an integrated collection of programs that supervise computer operations.

Output

The device involved in transferring data from memory; the data created as the result of transfer; processed data. For example, when your terminal displays the contents of a file, the terminal serves as an output device.

Parameter

The object of a command. A parameter can be a file specification or a keyword option.

Parity Bit

A binary digit appended to a group of bits to make the sum of all the bits always odd (odd parity) or always even (even parity). (Parity pertains to the way data is transmitted between the system and its devices.) A parity bit is used to verify data storage.

Password

The character string assigned to each user that verifies user access privileges to a system; a way of assuring file confidentiality on time-sharing systems. You need a PPN and a password to log in to a RSTS/E system.

Physical Name

A name for a peripheral device; identifies the drive, control circuit, and medium together. (Compare with Logical Name.)

Priority

The level of importance assigned to an operation in computing. For example, you can set the priority of a print request so that your job is printed before or after other jobs in the print queue.

Programming

The process of planning, writing, testing, and correcting the steps required for a computer to solve a problem or perform a desired operation.

Project-Programmer Number (PPN)

See Account Number.

Prompt

A symbol indicating that the system is prepared for input from a user.

Protection Code

A feature that allows you to decide who can have access to your files.

Pseudo Keyboard

A part of RSTS/E software, primarily used in batch jobs. The pseudo keyboard operates like a real terminal, except that the "typist" is a program rather than a person.

Qualifier

A command language keyword that modifies the operation of a command. Qualifiers are always preceded by slash (/) characters. (On some systems, a qualifier is called a "switch.")

Queue

A line-up of jobs to be performed in sequence, as in a batch or print queue, in which certain jobs are given priority over others.

Read

A process that performs input from a file or a device.

Read-Only

The state a device or file is in when it can be used for input but not for output. (Same as Write-Locked.)

Real Time

Synonym for time as we know it, in 24-hour days. (Compare with Absolute Time.)

Remote Node

Any node in the network other than the local node. In other words, a remote node is any node except the one you originally logged in to. (Compare with Local Node.)

RSTS/E

An acronym for Resource-Sharing Timesharing System/Extended, a DIGITAL operating system.

Run Burst

The length of time, measured in ticks, that a job is allowed to run uninterrupted in the CPU of a computer.

Run Time

The time in which a program is executed. Run time is the actual amount of CPU time required for a program to complete execution.

Run-Time System

System software that manages a programming language. The number of run-time systems can differ from one system to another.

Software

The set of programs that controls the operation of a computer system.

SPR

Software Performance Report. Submitted by a user (generally the system manager) to report a possible problem in the software.

Swapping

The transfer of a job from memory to disk or vice versa. Lets the system run more jobs than its memory can hold at once.

Switch

See Qualifier.

Syntax

The rules governing the structure of statements in a computer language; the structure of a language.

System

A combination of hardware and software that performs processing operations; a collection of components that forms a functional unit.

System Manager

The person or group of people who schedule the access and use of a computer system.

Temporary File

A file that is deleted from your account when you log out.

Terminal

A device, consisting of a keyboard and display mechanism, used to enter and receive data to and from a computer. LA120s (hardcopy) and VT100s (video) are examples of terminals.

Tick

A fraction of a time slice. A tick is about $\frac{1}{60}$ of a second.

Timesharing

A method of computer operation in which more than one user shares the system at the same time. The central theory of timesharing is that the system processes each user's job in slices of time, so rapidly that the user is often unaware that processing time is shared.

Time Slice

The amount of time (in fractions of a second) that the computer assigns to each user's job during timesharing.

Translator

Software that converts human-readable programs into machine-readable code.

Utility

A program or set of programs that performs a set of commonly used functions, such as copying or deleting files. The PIP program and the EDT editor are examples of utilities on RSTS/E.

Wildcard

A symbol that allows you to refer to more than one file with a single file specification. The asterisk (*) is a commonly used wildcard symbol in DCL, as in *.DAT to specify all names of data files.

Word

A unit of storage in the computer. A word holds 16 bits, or two ASCII characters, of information.

Write

A process that performs output to a file or a device.

Write-locked

A protection feature that does not permit the output of data or changes to a file or device. (Same as Read-Only.)

Index

C

D

HOW TO ORDER ADDITIONAL DOCUMENTATION

DIRECT TELEPHONE ORDERS

In Continental USA and Puerto Rico call **800–258–1710**

In Canada call **800–267–6146**

In New Hampshire, Alaska or Hawaii call **603–884–6660**

DIRECT MAIL ORDERS (U.S. and Puerto Rico*)

DIGITAL EQUIPMENT CORPORATION
P.O. Box CS2008
Nashua, New Hampshire 03061

DIRECT MAIL ORDERS (Canada)

DIGITAL EQUIPMENT OF CANADA LTD.
940 Belfast Road
Ottawa, Ontario, Canada K1G 4C2
Attn: A&SG Business Manager

INTERNATIONAL

DIGITAL EQUIPMENT CORPORATION
A&SG Business Manager
c/o Digital's local subsidiary
or approved distributor

Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Northboro, Massachusetts 01532

*Any prepaid order from Puerto Rico must be placed with the Local Digital Subsidiary:
809–754–7575

Reader's Comments

Note: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement. _____

_____

_____

_____

_____

_____

_____

Did you find errors in this manual? If so, specify the error and the page number. _____

_____

_____

_____

_____

_____

_____

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) _____

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code or Country _____

Do Not Tear - Fold Here and Tape

digital

No Postage Necessary if Mailed in the United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN: Office Systems Publications MK01-2/E02
RSTS/E Documentation
DIGITAL EQUIPMENT CORPORATION
CONTINENTAL BOULEVARD
MERRIMACK, N.H. 03054

Do Not Tear - Fold Here and Tape