DNOS

System Command Interpreter (SCI) Reference Manual

Part No. 2270503-9701 *C
15 November 1983

TEXAS INSTRUMENTS

MANUAL REVISION HISTORY

DNOS System Command Interpreter (SCI) Reference Manual (2270503-9701)

Original Issue ..... 1 August 1981
Revision ..... 1 October 1982
Revision ..... 15 November 1983

The total number of pages in this publication is 676.

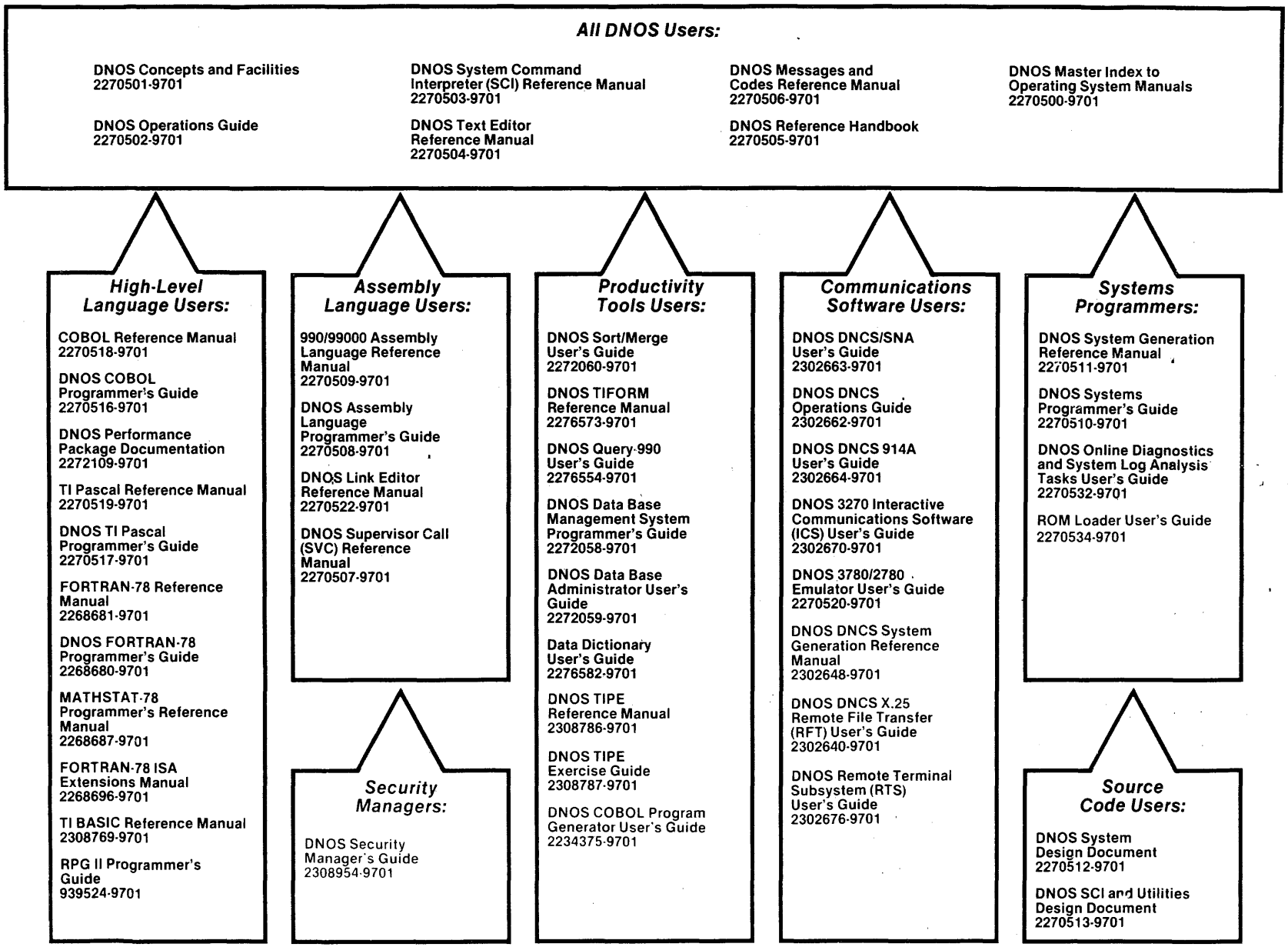
The computers offered in this agreement, as well as the programs that TI has created to use with them, are tools that can help people better manage the information used in their business; but tools—including TI computers—cannot replace sound judgment nor make the manager's business decisions.

Consequently, TI cannot warrant that its systems are suitable for any specific customer application. The manager must rely on judgment of what is best for his or her business.

2270503-9701

DNOS Software Manuals

This diagram shows the manuals supporting DNOS, arranged according to user type. Refer to the block identified by your user group and all blocks above that set to determine which manuals are most beneficial to your needs.



DNOS Software Manuals Summary

Concepts and Facilities
Presents an overview of DNOS with topics grouped by operating system functions. All new users (or evaluators) of DNOS should read this manual.

DNOS Operations Guide
Explains fundamental operations for a DNOS system. Includes detailed instructions on how to use each device supported by DNOS.

System Command Interpreter (SCI) Reference Manual
Describes how to use SCI in both interactive and batch jobs. Describes command procedures and gives a detailed presentation of all SCI commands in alphabetical order for easy reference.

Text Editor Reference Manual
Explains how to use the Text Editor on DNOS and describes each of the editing commands.

Messages and Codes Reference Manual
Lists the error messages, informative messages, and error codes reported by DNOS.

DNOS Reference Handbook
Provides a summary of commonly used information for quick reference.

Master Index to Operating System Manuals
Contains a composite index to topics in the DNOS operating system manuals.

Programmer's Guides and Reference Manuals for Languages
Contain information about the languages supported by DNOS. Each programmer's guide covers operating system information relevant to the use of that language on DNOS. Each reference manual covers details of the language itself, including language syntax and programming considerations.

Performance Package Documentation
Describes the enhanced capabilities that the DNOS Performance Package provides on the Model 990/12 Computer and Business System 800.

Link Editor Reference Manual
Describes how to use the Link Editor on DNOS to combine separately generated object modules to form a single linked output.

Supervisor Call (SVC) Reference Manual
Presents detailed information about each DNOS supervisor call and DNOS services.

DNOS System Generation Reference Manual
Explains how to generate a DNOS system for your particular configuration and environment.

User's Guides for Productivity Tools
Describe the features, functions, and use of each productivity tool supported by DNOS.

User's Guides for Communications Software
Describe the features, functions, and use of the communications software available for execution under DNOS.

Systems Programmer's Guide
Discusses the DNOS subsystems and how to modify the system for specific application environments.

Online Diagnostics and System Log Analysis Tasks User's Guide
Explains how to execute the online diagnostic tasks and the system log analysis task and how to interpret the results.

ROM Loader User's Guide
Explains how to load the operating system using the ROM loader and describes the error conditions.

DNOS Design Documents
Contain design information about the DNOS system, SCI, and the utilities.

DNOS Security Manager's Guide
Describes the file access security features available with DNOS.

Preface

This manual describes the features of the Distributed Network Operating System (DNOS) System Command Interpreter (SCI), which is the interface between the user and DNOS.

Read the material in the first three sections of this manual in order to gain a basic understanding of the functions and components of SCI before using the SCI commands. No attempt has been made in this manual to document SCI commands used for specific programming languages. For information regarding language SCI commands, refer to the appropriate DNOS language programmer's guide.

A brief description of the contents of this manual follows:

Section

1 Introduction — Defines special terms used throughout this manual, describes the structure of SCI commands, and discusses internationalization of SCI commands.

2 SCI Overview — Discusses the characteristics of SCI, how to activate SCI in different modes of operation, and discusses the SCI debugger.

3 SCI Command Groups and Menus — Discusses the categories of SCI commands. Commands in a given category are displayed on a terminal when the menu name is entered. Information on how to modify these menus is also presented.

4 DNOS SCI Commands — Discusses each SCI command, its prompts, and valid prompt responses. Examples in each SCI command discussion show the prompts of the command as they appear on the terminal.

Appendix

A Keycap Cross-Reference — Lists the generic key names and shows the corresponding keycap label on each terminal supported by DNOS. Throughout this manual, keys are referred to by generic names.

B Index of SCI Commands — Consists of a permuted index of the SCI commands discussed in Section 4, with accompanying page numbers to allow easy access to a specific command.

In addition to this manual, the DNOS software manuals shown on the support manual diagram (frontispiece) contain information related to SCI.

Contents

3 — SCI Command Groups and Menus

4 — DNOS SCI Commands

Appendixes

Index

Illustrations

Tables

1

Introduction

1.1 MANUAL ORGANIZATION

This manual provides a description of the System Command Interpreter (SCI), its functions and components, and the commands available through SCI. SCI is the user interface to the DNOS operating system. SCI prompts the user, accepts input, interprets commands, and directs activities to satisfy those commands.

Read the first three sections of this manual in the order presented. This will increase your ability to use the manual effectively when using it for future reference.

1.2 GENERIC KEY NAMES

The names used in this manual for individual keys on terminals are generic key names that apply to all terminals. In some cases the names on the keycaps of some terminals match the generic names, but in many cases they do not. You can identify which key on a particular terminal the generic name refers to by consulting Table A-1 in Appendix A. This table cross-references the generic key names with the keycaps of each terminal supported by DNOS. Appendix A also contains a reverse reference table showing commonly used keys on the Model 911 VDT and the corresponding generic key names.

1.3 SPECIAL TERMS AND WORD USAGE

This manual uses terms unique to DNOS during discussions of SCI functions and commands. Generally, terms used in different sections have the same meaning; however, several terms may have multiple meanings and will be discussed in the following paragraphs. For terms not defined in the following paragraphs, refer to the DNOS Glossary contained in the *DNOS Concepts and Facilities Manual*.

Although an *operator* and *user* both communicate with the DNOS operating system through terminals, these terms have different meanings when they appear in descriptions of SCI commands. The term *operator* denotes a person having system operator responsibilities, such as the person who controls system start and restart, places information media into the input devices, removes the output, and performs other related functions. The term *user* is defined as any person who makes use of the resources of a DNOS-based system.

The term *prompt* describes the SCI symbol ([ ]) and SCI command fields displayed on a terminal. The SCI prompt signifies that SCI is able to accept commands interactively. SCI displays its prompt when it is in the command mode and no foreground activity is in progress at the terminal. After a command is entered, field prompts are displayed to gather information used during command execution. This information is obtained from user responses to the field prompts.

The following are examples of the SCI prompt and the field prompt of the Show File (SF) command:

[ ]	SCI prompt.
SYSTEM*42	SCI prompt, with or without the symbol ([ ]). The value can be any set of characters you define and can be specified by the .OPTION primitive discussed in the *DNOS System Programmer's Guide*. The string SYSTEM*42 was chosen for this example.
[ ]SF SHOW FILE FILE PATHNAME:	Field prompt displayed when the SF command is entered through SCI. FILE PATHNAME is the field prompt for the SF command. The SF command is discussed in Section 4 of this manual.

The *command mode* specifies the mode in which a command processor interprets input as commands rather than as data. SCI is in the command mode when the SCI prompt is displayed at a terminal.

The term *command processor* denotes a task executed by an SCI procedure to perform a command.

A *command procedure* is a program in the SCI language that implements a command.

Primitives are the commands of the SCI language.

A *job* is an entity within the DNOS system that performs a user-defined function and is uniquely identified with a job ID. A job has access to certain resources such as input/output (I/O) devices and is made up of one or more cooperating tasks that execute concurrently or serially.

Foreground mode is a type of program execution in which a task interacts with the user's terminal.

Background mode is a type of program execution in which a task executes without interacting with the user's terminal. Other processing can take place in the foreground mode while a background activity proceeds.

A *batch stream* is a set of commands in which SCI bids a sequence of tasks and passes information from one task to the next. All parameters must be contained in the batch stream since execution is in the background mode and there is no interaction with the user's terminal.

The *Terminal Local File* (TLF) is a temporary file that is made available to command processors for communicating information to the user. If the command processor runs in foreground mode, the TLF is displayed upon completion of the command procedure. Enter the Show Background Status (SBS) command to display the TLF written by a processor running in background mode. When a command is executed from batch SCI, the TLF is written to the batch listing file.

Different privilege levels can be specified when user IDs are assigned to allow a user to execute commands having the same privilege level or lower. The term *privileged user* specifies that a user ID has been assigned a privilege level allowing execution of commands that modify critical system structures.

1.4 SCI COMMAND STRUCTURE

The SCI command names are in a verb/object format (that is, Kill Task, Map Disk, and so on). Since each command name describes the function that the command performs, you can easily recall the command that fits your need. The following are examples of SCI command names:

Command Name	Command Function
Show File	Show the contents of a file.
Backup Directory	Backup a file directory to another medium.
Copy KIF to Sequential File	Copy the contents of a key indexed file (KIF) to a sequential file.

Table 1-1 lists verbs used in SCI command names. By becoming familiar with these verbs, you can gain a general idea of which commands to reference to perform a specific function.

Table 1-1. Verbs of the SCI Command Names

ACTIVATE	LIST
ADD	MAP
ANALYZE	MODIFY
ANSWER	MOVE
APPEND	PATCH
ASSEMBLE	PRINT
ASSIGN	PROCEED
BACKSPACE	QUIT
BACKUP	READ
BEGIN	RECEIVE
BUILD	RECOVER
CALL	RELEASE
CHECK	REPLACE
CLEAR	RESPOND
COPY	RESTORE
COUNT	RESUME
CREATE	REWIND
DELETE	SAVE
DISPLAY	SCAN
END	SHOW
EXECUTE	SIMULATE
FIND	SNAPSHOT
FORWARD SPACE	TEST
HALT	TRANSFER
INITIALIZE	UNLOAD
INSERT	VERIFY
INSTALL	WAIT
KILL	WRITE

1.5 SCI INTERNATIONALIZATION

DNOS is designed to meet the international requirements of the United States as well as most Western European countries and Japan. All SCI commands can be modified by a system programmer to better meet the needs of the users in a particular country. If desired, SCI pathnames and SCI synonyms can contain local language characters. The details on how to internationalize SCI can be found in the *DNOS Systems Programmer's Guide*.

2

SCI Overview

2.1 DNOS ENVIRONMENT

DNOS is a general purpose, multitasking operating system designed to operate with the Texas Instruments Model 990 and Business Systems family of minicomputers using the memory mapping feature.

DNOS can be used to execute programs or utilities supplied by Texas Instruments or user-created programs or utilities. A variety of optional compilers, an assembler, and a debug program are available with DNOS to facilitate creating programs and utilities.

The System Command Interpreter (SCI) is the interface provided for user interaction with DNOS. SCI provides two types of jobs: interactive jobs that execute via an interactive terminal and batch jobs that are initiated from a terminal but execute independently from any terminal. If you desire, you can avoid using SCI by creating your own interface and all utilities needed.

2.2 SCI CONCEPTS

Commands can be entered through SCI from interactive jobs at terminals operating either in video display terminal (VDT) mode or in teletype (TTY) mode. SCI commands can also be used in batch jobs in the form of batch command streams.

Commands used at interactive terminals are entered in foreground mode. SCI responds to the entry of a command by displaying or printing field prompts associated with the command. In TTY mode, one field prompt at a time is displayed or printed, but a series of prompts is displayed simultaneously when the terminal is in VDT mode. After displaying or printing the field prompt, SCI waits for a user response. VDT mode is the recommended operating mode for the Texas Instruments video display terminals (VDTs). Teleprinter devices are always used in TTY mode.

Commands submitted in a batch command stream and executed by the Execute Batch (XB) or Execute Batch Job (XBJ) command operate without the aid of an interactive terminal. Required parameters for each SCI command must be supplied in the command stream. Since background activities do not interact with terminals you can start a process in background mode and perform other activities in foreground mode while the background task is active. Section 4 discusses the XB and XBJ commands.

SCI has several important characteristics in addition to the VDT and TTY terminal modes and the foreground/background capabilities. The following paragraphs discuss these additional characteristics.

2.2.1 SCI Interface Structure

DNOS is accessed through SCI commands or primitives that perform a desired function.

A command procedure, a program written with the primitives of the SCI language, implements an SCI command. Each command procedure can execute one or more tasks (command processors) to perform the command. Interaction with the user occurs through field prompts displayed on the terminal when you enter the command to SCI. You then enter appropriate responses to the field prompts, to determine the action taken by the command.

The available SCI commands can be displayed in command menus. Related SCI commands are grouped in command menus that allow you to view the options available for each major area of user interaction with DNOS.

Access to SCI in batch streams occurs through SCI commands (you must specify all responses to field prompts) and through the SCI language of primitive commands. It is also possible to execute SCI primitives interactively.

2.2.2 File and Directory Naming

The majority of DNOS users use disk files extensively. To do so effectively, you should be aware of the structure of the files on the disk.

Each disk volume has a specific file directory named VCATALOG where DNOS maintains a table of contents for the volume. The files described in VCATALOG can be data files or directory files as shown in Figure 2-1.

Any directory or file on a disk volume is referenced by its name (directory name or file name). The name of a directory is a concatenation of the volume name and directory name (excluding VCATALOG). A file name is a concatenation of the volume name, names of the directory levels (excluding VCATALOG) leading to the file, and a final component identifying the file. Each component of the directory name or file name must contain no more than 8 characters and must begin with an alphabetic character. Components of the name are separated by periods. The maximum length of the entire file name including periods cannot exceed 48 characters. If the volume name is not specified, the system disk is presumed. If the directory is not specified, VCATALOG is the presumed directory. The following examples are directory names and file names:

Directory Names	File Names
VOL2.BATCH	VOL2.BATCH.FILE1
.OUTPUT.LISTING	.OUTPUT.LISTING.RECORDS
MYDIR.MYDIRA	MYDIR.MYDIRA.MYFILE
EMPLOY01.USERA	EMPLOY01.USERA.PAYROLL

It is helpful to collect related files in a single directory. For example, all source files for a project might be placed in a directory named .PROJECT.SOURCE; all listings generated from assembly or compilation of source modules might be placed in a directory named .PROJECT.LISTING.

Avoid naming files which you might confuse with DNOS system files having file names and/or directory names beginning with S$.

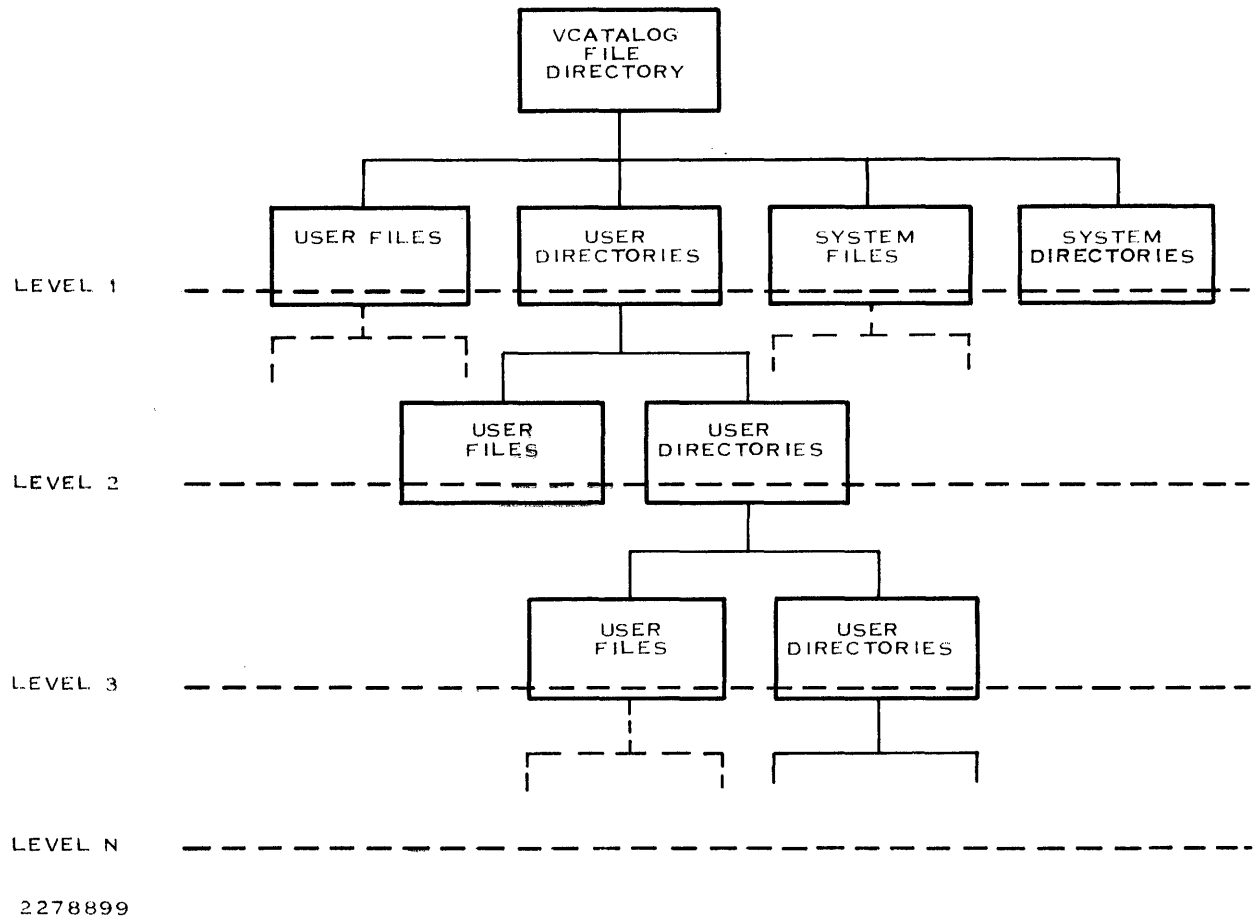


Figure 2-1. File and Directory Structure On a Disk

The file name DUMY can be used as a nonexistent file. All normal processing is done, but no output is generated. It is often convenient to use DUMY as a prompt response when you want to test some feature, but do not want to generate an output file.

When DNOS is used in a network of systems, disk file names can have an additional component. This component is a site name which appears as the first part of the file name. The site name is followed by a colon (:) and a file name as described above. For example, DALLAS:MYDIR.MYDIRA is a directory at site DALLAS on volume MYDIR. The file TOLEDO:.OUTPUT.LISTING.RECORDS resides on the system disk at the site named TOLEDO. If no site name is used, the user's own system is assumed to be the intended site. The user may specify his own site name; DNOS will use that as a local name.

2.2.3 Terminal Local File

All SCI commands transmit prompt responses to the command processor via an in-memory table of values. Some command processors return status and/or completion messages to you via a terminal local file (TLF). The system disk cannot be write-protected when executing any command that writes to the TLF.

A TLF is associated with SCI whenever SCI is activated at a terminal. SCI sends the results of a foreground operation to the foreground TLF if no other file or device is specified as the destination or output. The TLF is automatically displayed upon the completion of each foreground command execution. Press the F1 (roll up) key or the F2 (roll down) key on a VDT to examine output that takes up more than one display screen. (Function keys described with the Show File (SF) command in Section 4 of this manual can also be used with the TLF.) The TLF is cleared once it has been displayed and the Command key is pressed.

The background TLF is displayed at a terminal when a background task terminates and a Show Background Status (SBS) or a Wait (WAIT) command is executed. The TLF will not be displayed until the SBS or WAIT command is executed, even if the background task has terminated. Therefore, execute the SBS or WAIT command to ensure that the background TLF is displayed. If it is not displayed, subsequent background tasks that write to the TLF will have their output appended to the end of the current contents of the TLF.

2.2.4 Synonyms and Logical Names

Synonyms and logical names provide a means for referencing an I/O resource by a different name. The reference can be a file name or a more sophisticated resource, such as an interprocess communication channel or a multivolume file.

In essence, synonyms and logical names are sets of names and/or values. By substituting the value for the name, synonyms and logical names become useful referencing tools. In the case of logical names, the value can include a set of parameters in addition to the resource name. When you quit an interactive session with the Quit SCI (Q) command, the current synonyms and logical names are saved for use the next time you log on to SCI.

The Assign Synonym (AS) command allows you to assign synonyms; the Assign Logical Name (ALN) command allows you to assign logical names. Section 4 discusses the AS and ALN commands.

2.2.5 Logical Name Manipulation

With logical names, the associated value is always assumed to be an I/O resource. With synonyms, the value is treated as a text string (for example, file name, device name, and so on). Synonym names are always resolved under the control of the application program. On the other hand, by defining a logical name, you make it known to the operating system that all references to that logical name are to apply to the associated I/O resource. For example, when an application programmer needs the value of a synonym, a call to S$MAPS must be made, or an at sign (@) must be placed before the synonym name in an SCI command procedure. With logical names, the resolution automatically occurs each time it is used in context as an I/O resource.

Logical names can be defined to be available only for the user's job (job-level) or they can be available to all users of the system (global). Global logical names are especially convenient for referencing system-wide resources that may need to physically occupy different places at different times.

2.2.6 Environment and Scope of Name Definitions

At any given time, synonyms and logical names assigned by each task in a job are available to every task in that job. The effect that these name definitions have on an executing task can be thought of as the task's environment. When a background task is started, a snapshot of the bidding task's environment is made. The background task continues execution with the available synonyms and logical names of the bidding task. Definitions that are changed by either task, subsequent to the bid, do not affect the environment of the other.

Permanent copies of the synonyms and logical names are kept in files on the system disk. When a user logs off the terminal, the files are updated. Since several users may log on with the same user ID, each of them begins with the current set of synonyms and logical names from the current copy of the files. When the last user of a given user ID logs off the terminal, his current set of synonyms and logical names is saved to the files. Thus, multiple use of the same user ID requires the users' cooperation when assigning and deleting synonyms and logical names.

2.2.7 Error and Status Messages

When a special condition occurs, SCI displays an error or status message on the bottom line of the terminal screen. Further operation is inhibited until you acknowledge the message by pressing the Command or Return key. If you press the Command or Return key shortly before the message is displayed on the terminal, you may not see it since it is displayed only for an instant.

Messages can be generated within SCI while completing an SCI command or by a processor activated by an SCI command.

The messages displayed have three parts: the source indicator, a unique identifier, and the actual message. The source indicator is one of the following characters or character combinations:

Characters	Meaning
I	Informative Message
W	Warning Message
U	User Error
S	System Error
H	Hardware Error
US	User or System Error
UH	User or Hardware Error
SH	System or Hardware Error
USH	User, System, or Hardware Error

The unique identifier is a code containing the category of the message, such as SCI, COBOL, or UTILITY. This code may be followed by an identifier for the message within that category.

The actual message is a phrase describing the special condition that has occurred. For example, if you attempt to access a nonexistent file, the following message appears on the terminal:

```
U SVC-0315 <filename> DOES NOT EXIST
```

where:

< filename> is the name of the file you tried to access.

If the short message provided is not sufficient to enable you to understand the problem or warning, you can obtain further information in several ways. If your system supports expanded message information online and your terminal is in VDT mode, you can type a question mark (?) for further explanation. The same information can also be gathered by use of the Show Expanded Message (SEM) command, described in Section 4 of this manual. If your system does not support online expanded messages, the *DNOS Messages and Codes Reference Manual* can be referenced for further information on error codes and messages.

Several SCI commands display status messages to inform you what action is being taken during the steps of command execution. These status messages are also displayed on the bottom line of the terminal screen. Again, you must acknowledge the message by pressing the Command or Return key before further operation can continue; further information can be obtained in the same way as for error messages.

The following example is a status message displayed by the SBS command while a batch stream is executing:

```
I STATUS-1227 WAITING FOR COROUTINE ACTIVATION
```

2.2.8 Privileged Command Levels

Privilege levels are assigned to user IDs when a system manager or a privileged user executes the Assign User ID (AUI) command. Nonprivileged users of DNOS are limited in the commands they can enter due to the privilege levels associated with their user IDs (if operating a terminal requires user identification) or because of the privilege level assigned to the terminal itself (if the terminal used does not require user identification). Table 2-1 lists the DNOS privilege levels and Table 4-5 lists the privilege levels associated with each SCI command.

Table 2-1. Command Privilege Levels

Level	Meaning
0	Lowest level of access privilege (for example, Create File (CF))
1	System Manager defined
2	System access level (for example, Kill Task (KT))
3	System Manager defined
4	Management access level (for example, Assign User ID (AUI))
5	System Manager defined
6	Combination of System and Management (for example, Execute System Generation Utility (XSGU))
7	System Manager defined

The log-on process requires that a valid user ID be supplied since it involves the creation of a new job. You can supply the ID (where required user identification has been specified) or DNOS can supply it (where user identification has been specified as being not required). Identification that is not required means that DNOS uses a default user ID to determine the privilege level associated with the terminal. Define the default user IDs by using the Modify Terminal Status (MTS) command described in Section 4 of this manual.

Regardless of the privilege level assigned to your user ID, the SCI operator commands can be executed only by a system operator or through a terminal designated for use by a system operator. The Execute Operator Interface (XOI) command, discussed in Section 4 of this manual, can be executed to specify a terminal as the system operator's terminal. Refer to Table 3-1 for a list of DNOS operator commands.

2.2.9 File Security

When a system is using file security, each user ID is a member of one or more access groups. The access groups are allowed to use only certain files for certain types of access. Since SCI command procedures reside in files, they can be secured for use by only certain access groups. The system security manager determines which commands are secured. If there are commands described in this manual which you cannot access, check with your security manager.

Your security manager may also be responsible for determining which files have specific access rights for particular groups. Access rights, in addition to privilege levels, are used to determine who can use which commands. The access rights available are read, write, delete, execute, and control. The read access right is required for a file accessed by a command if that command shows data in the file or examines the file for input. The write access right is required for a file accessed by a command that modifies or updates that file. To issue a command that deletes a file, you must have the delete access right to that file. A command that executes a task from a program file requires that you have the execute access right for that program file. The control access right is required for any command that changes the access rights to a file. In most cases, an understanding of the access rights is sufficient to predict the required access rights to use a particular command. Where such an understanding is not sufficient, security-related information is included in the description of the command.

2.3 INTERACTIVE USE OF SCI

To use DNOS, you must activate SCI at a terminal. You may be required to enter your user ID and passcode before SCI can be accessed through that terminal. To deactivate SCI at any terminal and log off, enter the Quit SCI (Q) command in response to the SCI prompt as follows:

```
[] Q
```

The MTS command allows you to define parameters associated with a terminal, including those requiring entry of user identification. The MTS and Q commands are discussed in Section 4 of this manual.

2.3.1 SCI Activation

The following procedure shows the steps to activate SCI at your video display terminal. Some of the following prompts do not appear if they have been previously defined for the terminal in use.

1. Turn on the terminal if it is not already on.

2. Press the Attention key.

3. Enter the exclamation mark (!).

4. If this is the first terminal to log on after loading DNOS, enter the year, month, day, hour, and minute as requested.

5. DNOS responds by displaying or printing the following message:

```
DNOS 1.2.00
```

   where:

   1.2.00 is the current released version of DNOS.

6. If user identification is required, DNOS displays the following prompts:

```
 USER ID:
PASSCODE:
```

   Type in your assigned user ID and press the Return key to signal DNOS that an entry has been made. The user ID must be all uppercase characters. Next, type in your assigned passcode and press the Return key to signal DNOS that another entry has been made. To preserve passcode security, the characters of the passcode are not displayed.

7. DNOS may respond by displaying the following prompt (if it is not predefined by the Modify Terminal Status (MTS) command).

```
ACCOUNT ID:
```

   Type in the assigned account ID and press the Return key to signal that an entry has been made. Your account ID must be a valid number found in the file .S$ACCVAL.

8. DNOS may respond by displaying the following prompt (if it is not predefined by the MTS command):

```
JOB NAME:
```

   Type in a job name and press the Return key to signal DNOS that an entry has been made. A job name can be any alphanumeric string (maximum of eight characters) that starts with an alphabetic character or $ and consists of only uppercase characters.

9. DNOS may respond by displaying the following prompt (if it is set for display by the MTS command):

```
SYNONYM FILE PATHNAME:
```

   Enter the pathname of the file that contains the synonyms and logical names you want to use, or press the Return key to accept the default pathname (the file in which the synonyms for your user ID are normally kept). If a file is supplied, it must have been originally established by the Snapshot Name Definitions (SND) command.

10. If the job name entered is already in use with the same user ID, DNOS may respond with the following prompt:

```
RECONNECT?:
```

    Type in YES or NO and press the Return key to signal that an entry has been made. YES specifies that the terminal is also to be associated with the job in use. NO specifies that the terminal is to be associated with a new job. An ACCOUNT ID validation is not performed when YES is specified.

11. If the log-on is successful, DNOS either displays the main menu and the SCI prompt ([ ]) or it displays the news file if one exists. If a news file is displayed, SCI waits for the Command key to be pressed. After the Command key is pressed, SCI displays the main menu and SCI prompt ([ ]) as shown in Figure 2-2. The main menu can be changed at the option of the systems programmer.

12. Enter the SCI commands that are available to you to begin operating the terminal. If you enter a command that is not authorized for your user ID, SCI will display an appropriate error message.

13. While executing SCI commands, do not turn off the terminal. If the terminal is turned off, device errors are written to the system log and the system may loop in an attempt to complete the command.

2.3.2 Entry of SCI Commands in VDT Mode

To enter an SCI command in VDT mode, type the characters of the command in uppercase letters and press the Return key. Lowercase letters can be used only if the .OPTION primitive has been specified with the lowercase option. (Refer to the *DNOS Systems Programmer's Guide* or a detailed description of the .OPTION primitive.) Upon entry of a command, SCI displays the full name of the command entered and all of the field prompts associated with the command. Field prompts provide information and request parameters to complete command execution. For example, the following field prompt asks you to identify an output pathname:

```
OUTPUT PATHNAME:
```

2.3.2.1 Responses to Field Prompts. After you enter a command in VDT mode and the field prompts associated with the command are displayed, SCI places the cursor after the first field prompt and waits for your response. To enter a response to a field prompt, type the appropriate response or accept the displayed value by pressing the Return key to signal SCI that the response is complete.

```
**************************************************
**       T E X A S   I N S T R U M E N T S      **
**       D N O S   S Y S T E M   1 . 2 . 0      **
**************************************************

Command Groups:

/DEBUG  - Interactive Debugger          /OPER   - Operator Interface
/DEVICE - I/O Devices                   /PREXEC - Program Execution
/DIR    - Directories                   /PRFILE - Program Files
/EDIT   - Text Editor                   /SECURE - File Security
/FILE   - File Management               /SPOOL  - Spooler
/JOB    - Job Management                /STATUS - Status Reports
/LANG   - Language Support              /SYSCON - System Configuration
/LUNO   - Logical Unit Numbers          /SYSGEN - System Generation
/MSG    - Message Facilities            /SYSMGT - System Management
/NAME   - Synonyms and Logical Names    /TPD    - Teleprinter Devices
/NET    - Networking Support            /VOLUME - Disk Volumes


[]
```

Figure 2-2. SCI Main Menu

Upon entry of a response to the first field prompt, the cursor moves to the field following the next field prompt and waits for you to enter another response. After you enter a response to the last field prompt, the command executes.

The command executes if you press the Enter key after a response to a field prompt, since it is assumed that the responses to the remaining field prompts are to be accepted without changes.

2.3.2.2 Initial Values. When you enter a command, some of the prompts may show responses. These responses are known as initial values. They may be specified in the command procedure as exact values or as synonym values. You can accept the initial value by pressing the Return key.

2.3.2.3 Default Values. Many field prompts allow you to specify no response. The command processor then uses a value it has defined for this case. This value is known as a default.

2.3.2.4 Response Entry Control. Table 2-2 lists the keys that SCI uses for terminal functions during the time it is accepting user responses to field prompts when the terminal is in VDT mode. Table 2-3 shows the special terminal control keys that can be used when an SCI command is executing at a terminal in VDT mode.

Table 2-2. VDT Mode Command Entry Keys

Key	Function
Home	Positions the cursor at the beginning of the response field of the first field prompt displayed on the screen. You can reenter responses to all of the field prompts. The Home key does not allow you to go back to a previously displayed set of field prompts of the same command.
Return	Used to accept the current response to the field prompt. It is also used to position the cursor in the response field of the next field prompt or to start command processing if the command has no more field prompts.
Skip	Deletes all characters to the right of the cursor when it is positioned in the response field of the current field prompt and moves the cursor position to the start of the response field of the next field prompt.
Next Field	Moves the cursor position to the start of the response field of the next field prompt. If there are no more field prompts, the command executes.
Previous Field	Moves the cursor position to the start of the previous response field.
Erase Field	Deletes an entire response to a field prompt including any initial value supplied by the system. You can then enter a new response to the same field prompt.
Erase Input	Resets any initial values and goes to the first field prompt displayed on the terminal. The initial values are reset only for the set of field prompts currently displayed.
Command	Aborts a command currently accepting user responses and returns the terminal to command mode.
F4	Causes duplication of the corresponding field from the previous line. This key does not cause field termination. You can either accept the duplicated field by pressing the Return key or modify the duplicated field before accepting it.
Back Tab	Positions the cursor to the beginning of the response field for the current field prompt.
Forward Tab	Moves the cursor to the next pathname component in the current response field. If the cursor is positioned at the end of the pathname, pressing this key moves the cursor position to the beginning of the current response field.
Previous Line	Positions the cursor to the same character position in the response field of the previous field prompt. If there is no previous field prompt, the cursor position is unchanged.
Next Line	Positions the cursor to the same character position in the response field of the next field prompt. If there is no next field prompt, the cursor position is unchanged.
Previous Character	Moves the cursor one position to the left in the response field of the current field prompt.

Table 2-2. VDT Mode Command Entry Keys (Continued)

Key	Function
Next Character	Moves the cursor one position to the right in the response field of the current field prompt.
Enter	Enters responses to all field prompts as displayed. If a syntax error occurs, the cursor is positioned to the first character position of the field prompt response field containing the syntax error.
Repeat	Used in conjunction with the Next Line, Previous Line, Next Character, or Previous Character to repeat the functions of the key pressed. The Repeat key can also be used with all other keys.
Insert Character	Allows you to insert new characters, beginning at the current cursor position in the response field of the current field prompt. Any characters currently in the response field are shifted to the right. Characters can be inserted until the response field is full.
Delete Character	Deletes a character beginning at the cursor position and shifts the remaining characters in the response field of the field prompt one character position to the left.

Table 2-3. Control Keys for Commands Executing in VDT Mode

Key	Function
Attention	Temporarily stops output at a terminal. Pressing any alphabetic or numeric key continues output.
Attention then Return	Returns an error to the executing task. Most DNOS tasks programmed by Texas Instruments abort when the error is returned to the task. However, you can program a task to continue execution if you desire.
Attention then Control and X	Aborts the executing commands that can be aborted to return control to the user. One task is killed each time the break key sequence is used. If only SCI is active, it is killed, the job at the terminal terminates, and the terminal is again available for use. If tasks other than SCI are active, the order in which tasks are killed is: foreground tasks, background tasks, and SCI. If this control key sequence is issued to a terminal not logged on to SCI, the log-on prompts appear, but no task is bid and the log-on prompts remain on the screen. (The normal log-on procedures must be performed to log on SCI by pressing the Attention key followed by the exclamation point.)

2.3.2.5 Example of Entry of SCI Commands in VDT Mode. The following example shows the entry of the Execute Macro Assembler (XMA), Execute Link Editor (XLE), and the Install Task (IT) commands when a terminal is in VDT mode. Default and initial values are noted.

```
[] XMA

EXECUTE MACRO ASSEMBLER
         SOURCE ACCESS NAME: SYS2.KC0017.SOURCE
         OBJECT ACCESS NAME: SYS2.KC0017.OBJECT
        LISTING ACCESS NAME: SYS2.KC0017.LIST
          ERROR ACCESS NAME:                  default value
                   OPTIONS:                   default value
    MACRO LIBRARY PATHNAME:                   default value
       PRINT WIDTH (CHARS): 80                initial value
       PAGE LENGTH (LINES): 60                initial value


[] XLE

EXECUTE LINK EDITOR
        CONTROL ACCESS NAME: SYS2.KC0017.CONTROL
 LINKED OUTPUT ACCESS NAME: SYS2.KC0017.LNKOUT
        LISTING ACCESS NAME: SYS2.KC0017.LNKLST
         PRINT WIDTH(CHARS): 80              initial value
        PAGE LENGTH (LINES): 59              initial value


[] IT

INSTALL TASK SEGMENT
     PROGRAM FILE OR LUNO: SYS2.KC0017.PROG
                TASK NAME: MYTASK
                  TASK ID:                   default value
  OBJECT PATHNAME OR LUNO: SYS2.KC0017.LNKOUT
                 PRIORITY: 3
       DEFAULT TASK FLAGS?:                  initial value
YES
       ATTACHED PROCEDURES?: YES
```

The following optional field prompts are displayed because the response to the ATTACHED PROCEDURES? field prompt was YES.

```
ATTACH TASK PROCEDURES
        1ST PROCEDURE ID: >42
P1 FROM TASKS PROGRAM FILE?: YES           initial value
           2ND PROCEDURE ID: 0             initial value
P2 FROM TASKS PROGRAM FILE?: YES           initial value
```

2.3.3 Entry of SCI Commands In TTY Mode

To enter an SCI command in TTY mode, type the characters of the command in uppercase letters and press the Return key. Lowercase letters can be used only if the .OPTION primitive has been specified with the lowercase option. Upon entry of a command, SCI displays the full name of the command entered and the first field prompt associated with the command. Field prompts provide information and request parameters from you to complete command execution. For example, the following field prompt asks you to identify an output pathname:

```
OUTPUT PATHNAME:
```

2.3.3.1 Responses to Field Prompts. After entry of a command in TTY mode and the display of the first field prompt associated with the command, SCI places the cursor after the first field prompt and waits for a response. To enter a response to a field prompt, proceed as follows:

1. Type the appropriate response, or accept the displayed value by not typing anything in the field.

2. Press the Return key to signal SCI that the response is complete.

Upon entry of a response to the first field prompt, SCI displays the next field prompt. The cursor moves to the response field of the field prompt and waits for you to enter a response. After you enter a response to the last field prompt, the command executes.

If you press the Enter key after the response to any field prompt, it is assumed that the responses to the remaining field prompts not seen are to be accepted without changes.

2.3.3.2 Response Entry Control. Table 2-4 shows the keys that SCI uses for terminal control during the time that it is accepting user responses to field prompts when the terminal is in TTY mode. Table 2-5 shows the special terminal control keys that can be used when an SCI command is executing while the terminal is in TTY mode.

Table 2-4. TTY Terminal Mode Command Entry Keys

Key	Function
Return	Can be used to accept the currently displayed initial value as the response to the field prompt. It can also be used to terminate the current entry and go on to the next field prompt or start command processing if the command has no more field prompts.
Skip	Deletes all characters in the response field of the current field prompt to the right of the cursor and positions the cursor at the start of the response field of the next field prompt.
Next Field	Depending on the position of the cursor, pressing this key moves the cursor position forward to the start of the response field of the next field prompt. If there are no more field prompts, the command executes.
Previous Field	Redisplays, or causes the reprinting of, the previous field prompt.

Table 2-4. TTY Terminal Mode Command Entry Keys (Continued)

Key	Function
Erase Field	Deletes an entire response to a field prompt. You can then enter a new response to the same field prompt.
Erase Input	Resets initial values and goes to the first field prompt displayed on the terminal. The initial values are reset only for the field prompts currently displayed.
F9	Redisplays or causes the reprinting of the current field prompt without the initial value that was first supplied by the system. You can then enter a new response to the field prompt.
Command	Aborts a command currently accepting user responses and returns the terminal to command mode.
F4	Causes duplication of the corresponding field in the previous line. This key does not cause field termination. You can either accept the duplicated field by pressing the Return key or modify the duplicated field before accepting it.
Back Tab	Positions the cursor to the beginning of the response field for the current field prompt.
Previous Line	Positions the cursor to the first character position in the response field of the previous field prompt.
Next Line	Positions the cursor to the first character position in the response field of the next field prompt.
Previous Character	Moves the cursor one position to the left in the response field of the current field prompt.
Next Character	Moves the cursor one position to the right in the response field of the current field prompt.
Enter	Enters responses to all field prompts as displayed. If a syntax error occurs, the cursor is positioned to the first character position of the response field containing the syntax error.
Repeat	Used in conjunction with the Next Line, Previous Line, Next Character, or Previous Character to repeat the functions of the key pressed.
Insert Character	Allows you to insert new characters, beginning at the current cursor position in the response field of the current field prompt. Any characters currently in the response field are shifted to the right. Characters can be inserted until the response field is full.
Delete Character	Deletes a character beginning at the cursor position and shifts remaining characters in the response field of the field prompt one character position to the left.

Table 2-5. Control Keys for Commands Executing in TTY Mode

Key	Function
Attention	Temporarily stops output at a terminal. Pressing any alphabetic or numeric key continues output at the terminal.
Attention then Return	Returns an error to the executing task. Most DNOS tasks programmed by Texas Instruments abort when the error is returned to the task. However, you can program a task to continue execution if you desire.
Attention then Control and X	Aborts those executing commands that can be aborted to return control to the user. One task is killed each time the break key sequence is used. If only SCI is active, the task is killed, the job at the terminal terminates, and the terminal is again available for use. If tasks other than SCI are active, the order in which tasks are killed is foreground tasks, background tasks, and SCI. If this control key sequence is issued to a terminal not logged on to SCI, the log-on prompts appear, but no task is bid and the log-on prompts remain on the screen. Normal log-on procedures must be performed to log on SCI.

2.3.4 Executing Commands via Expert Mode

Entering commands in expert mode is comparable to issuing a command in batch mode, except that commands are entered at the interactive terminal. The advantages of expert mode over normal command entry are that it takes less time, uses fewer keystrokes, and saves paper for teleprinter devices.

To utilize expert mode effectively, you need to be familiar with the characteristics of the command. Specifically, it is assumed you know all field prompts and initial values for the command and its required prompts. You enter the command and its field prompt assignments by entering a comma (,) or an equal sign (=) as the last character of each line to designate continuation to the next line.

NOTE

The assignments which are made apply only to field prompts that occur in the primary command. For example, if the command procedure calls another command procedure, the assignments are not automatically passed to the called command.

SCI formats the field prompts on the screen and suppresses the display of the full name that normally appears on the top line of the screen. All values to required field prompts must be supplied and syntactically correct; any prompt that already has an initial value can be omitted. For example:

```
[] SF FILE=.S$NEWS
```

shows the news file to the interactive terminal without prompting for FILE PATHNAME. Note that the field prompt FILE PATHNAME can be abbreviated, as in batch streams, to FILE.

If you type in the SF command again, the initial value for FILE PATHNAME is .S$NEWS. Since prompts with initial values can be omitted and no other prompts are to be entered, a period following the command enters all prompts with their initial values. For example:

```
[] SF.
```

shows the news file since .S$NEWS is the initial value.

If anything appears after a command name in response to the SCI command prompt, it is assumed that you are in expert mode. In the first example the assignment FILE = .S$NEWS and in the second example the period (.) caused SCI to execute in expert mode.

Field prompt assertion is another feature of SCI that is especially useful in expert mode. For instance, when the value of a field prompt assignment is YES, it is semantically equivalent to list the field prompt without the assignment. For example:

```
[] QE ABORT=YES
```

is equivalent to

```
[] QE ABORT
```

or simply

```
[] QE A
```

In this case, the command directs the Text Editor to abort. If the currently edited file is to be replaced, the command:

```
[] QE REPLACE
```

quits the Text Editor and directs the Text Editor to replace the file. The semantic result is that you assert this quality or condition. In expert mode, SCI treats this condition as if you had typed =Y after the field prompt. Special care must be taken in the use of this feature because some field prompts have a negating qualifier at the end. For example, if you assert the field prompt DELETE, the full name of the field prompt may be DELETE PROTECT.

Expert mode has one additional quality that makes it especially useful for a TTY device. When using expert mode, if you do not supply a required value, and an initial value is not supplied, then SCI will prompt you for that value. For example:

```
[] PF FILE=.S$NEWS
```

If a PF was previously executed, then the LISTING DEVICE field prompt has an initial value; however, if this is the first PF executed for this session, SCI prompts for:

```
LISTING DEVICE OR CLASS:
```

After you respond to this prompt, the command executes. In VDT mode, this can be confusing because LISTING DEVICE OR CLASS is not the last prompt. In this case, SCI does not ask for the subsequent field prompts that appear on the screen. Nevertheless, in TTY mode the value of this feature is the number of lines of saved paper. Note that this mechanism places the burden of knowing the unseen assignments or initial values on you.

2.3.5 News File

The news file is a sequential file .S$NEWS that is displayed by SCI immediately after SCI is activated at a terminal. The news file is displayed only if the terminal is in VDT mode. If you press the Command key, SCI will display the main menu. The news file can be used to notify users of changes to or problems with the system, as well as to provide information to all system users. You can edit the file .S$NEWS by using the Text Editor. You can also view the file at any time, in VDT or TTY mode, by executing the Show File (SF) command, described in Section 4 of this manual.

Refer to the *DNOS Text Editor Reference Manual* for information concerning text editing a file.

2.3.6 User and Operator Communications

You can create a message at a terminal with the Create Message (CM) command and send that message to any terminal in the system. The message is displayed at the destination terminal when SCI determines that the terminal is available to receive the message.

Communication with the system operator occurs through the operator interface of DNOS. You can send a message to the operator by using the Create Operator Message (COM) command. The system operator can initiate the operator interface using the Execute Operator Interface (XOI) command, respond to requests using the Respond to Operator Interface Request (ROR) command, cancel a request in the current set by using the Kill Operator Interface Request (KOR) command, and cancel the effect of the XOI command by using the Quit Operator Interface (QOI) command.

If the Receive Operator Messages (ROM) command was entered, either the operator or any user can examine the current set of messages for the operator by using the List Operator Messages (LOM) command. The operator or any user can also receive a copy of any message sent to the operator by specifying the ROM command and cancel the effect of the ROM command by using the Kill Operator Messages (KOM) command.

The operator interface commands and the CM command are discussed in Section 4 of this manual.

2.4 BATCH USE OF SCI

SCI can be used in batch mode with the Execute Batch (XB) command or the Execute Batch Job (XBJ) command. Both commands execute batch streams; the difference between them is the job in which they execute. Jobs initiated by the XBJ command run independently of the terminal. The XBJ command allows you to initiate several jobs at one terminal without affecting the foreground or background mode of the interactive job. The XB command executes one program in background mode at your terminal, inhibiting the use of background mode for other commands.

The following paragraphs discuss the format of commands used in batch streams and the execution of batch SCI.

2.4.1 Batch Stream Format

Batch streams are text edited using the Execute Text Editor (XE) command discussed in Section 4 of this manual. (Refer to the *DNOS Text Editor Reference Manual* for additional information.) The first and last commands should be the BATCH and EBATCH commands, respectively. The BATCH command initializes the batch SCI environment. EBATCH specifies that there are no more commands in the batch stream for SCI to process. The SCI commands to be executed in batch format must appear between these two commands.

Each SCI command in a batch stream must include the following information:

- The SCI command
- All required field prompts associated with the command
- The parameter values (your responses) assigned to the field prompts

The format of a batch stream command to supply the preceding information is as follows:

command f1 = value,f2 = value,...,fn = value

where:

command	Characters of the SCI command
f1,f2,...,fn	Field prompts for the first, second, and other parameters associated with the command. A field prompt is either the full field prompt associated with each parameter or an abbreviation of the field prompt.
value	Response that you enter

For example, the Execute Macro Assembler (XMA) command can be entered in a batch stream as follows (field prompts with optional responses need not be specified):

```
XMA  S=SYS2.KC0017.SOURCE,OB=SYS2.KC0017.OBJECT,
     L=SYS2.KC0017.LIST,OP=(XREF)
```

Batch streams can have comment statements in them. A comment line begins with an asterisk in column one. A comment can also be added to any line of a batch stream by preceding the comment with an exclamation mark.

2.4.2 Field Prompts

A field prompt is a character string which requests a valid response to execute an SCI command. This character string contains a maximum of 28 characters, including embedded blanks. When used in a batch stream, the field prompt can be abbreviated according to the near-equality algorithm. This algorithm considers a field prompt and its abbreviation *nearly equal* if all of the following conditions are met:

1. The field prompt and the abbreviation must have the same first character. Using the field prompt SOURCE ACCESS NAME as an example:

Acceptable	SRC
Not acceptable	OURCE ACCESS NAME

2. All characters in the abbreviated field prompt must appear in the order that they are found in the field prompt. Using the field prompt OBJECT ACCESS NAME as an example:

Acceptable	OJTAS
Not acceptable	OJB or OBJZ

3. If the abbreviated field prompt includes any character of a word in the field prompt, then the abbreviation must include the first character of that word and the first character of all preceding words. Using the field prompt OUTPUT ACCESS NAME as an example:

Acceptable	OAN
Not acceptable	ON or OUTPUTCC

4. A special character contained in a field prompt can be excluded or included in the abbreviation. If the abbreviation terminates before the special character, the abbreviation must be nearly equal to the characters preceding the special character of the field prompt. Using ANY*KID*CAN as a hypothetical field prompt:

Acceptable	A or AN or AY
Not acceptable	AK or ANYK

5. There is one exception to the near-equality algorithm. Any numeric digit in the field prompt must be included in the abbreviation. The following example uses the field prompt OBJECT ACCESS NAME1:

Acceptable	OAN1
Not acceptable	OAN

If the digit is preceded by a special character and the special character is not included in the abbreviation, then the digit need not be included in the abbreviation. Using the field prompt OBJECT ACCESS NAME(1), the first parenthesis encountered is considered a special character:

Acceptable	O or OAN
Not acceptable	OAN1

2.4.2.1 Initial Values, Default Values, and Optional Prompts. Initial values, default values, and optional field prompts for an SCI command in a batch stream are essentially the same as when SCI is accessed interactively. However, since you are not prompted for field prompt responses, your supplied values must be specified in the batch stream prior to execution. Initial values need not be specified in the batch stream. Also, you must ensure that appropriate optional field prompts and responses are included. For instance, if the Install Task (IT) command is to be executed in a batch stream and you want to specify a task name and a priority, the field prompts and responses for TASK NAME and PRIORITY must be included. Notice that in the XMA example for VDT mode, the default and initial values were accepted for five of the field prompts.

You must be aware of initial values that are set in the batch stream through synonyms. If you do not want the synonym value as a response to a particular field prompt at any point within the batch stream, you must set the field prompt to the desired value. Command descriptions in Section 4 of this manual indicate which prompts have initial values, which prompts are defined or set by synonyms, and which prompts have defaults. As in interactive SCI, default values for field prompts need not be specified.

2.4.2.2 Example Batch Stream of SCI Commands. The following example shows the XMA, XLE, and the IT commands when used in a batch stream executed with the XB command. Default and initial values, and optional field prompts are specified. A message specifying the number of errors encountered, if any, during batch stream execution is displayed on your terminal when execution completes as a result of the Error Counter (EC) and Create Message (CM) commands within the batch stream. The BATCH command sets the synonym EC to zero and the EC command adds to the value of EC when errors occur. The CM, EC, IT, XMA, and XLE commands are described in Section 4 of this manual.

```
BATCH
XMA  SOURCE=SYS2.KC0017.SOURCE,
     OBJECT=SYS2.KC0017.OBJECT,
     LISTING=SYS2.KC0017.LIST ,
     ERROR ACCESS NAME="",
     OPTIONS=(),
       MACRO="",
     PRINT=80,
     PAGE=60
EC
XLE  CONTROL=SYS2.KC0017.CONTROL,
     LINKED=SYS2.KC0017.LNKOUT,
     LISTING=SYS2.KC0017.LNKLST,
     PRINT=80,
     PAGE=59
```

```
EC
IT   PROGRAM=SYS2.KC0017.PROG,
     TN=MYTASK,
     TID=0,
     OBJECT=SYS2.KC0017.LNKOUT,
     PRIORITY=3,
     DEFAULT=YES,
     ATTACHED=YES,
     1ST=>42,
     P1=YES,
     2ND=0,
     P2=YES
EC
CM   TEXT="NUMBER OF ERRORS ENCOUNTERED IS @$E$C"
EBATCH
```

2.4.3 Execution of Batch Streams and Batch Jobs

You can execute batch streams with the XB command or the XBJ command. The XB command executes the batch stream in background mode, allowing you to continue working in the foreground mode. However, the XB command inhibits other background activity until the batch stream is complete. When the batch stream completes, a completion message appears the next time you press the Command or Return key. You can also enter the SBS command from time to time to allow you to monitor batch stream execution.

After the batch stream completes executing in background mode, the following message will appear on your terminal:

```
BATCH SCI HAS COMPLETED
```

If you do not want to perform any foreground activity while the batch stream executes, you can enter the WAIT command. This allows the batch completion message to be displayed without pressing the Command key. However, the Command key can be pressed at any time to activate foreground SCI. The following is an example of how to issue the XB and WAIT commands.

```
[] XB

EXECUTE BATCH
       INPUT ACCESS NAME: MY.BATCH
    LISTING ACCESS NAME: MY.LIST

[] WAIT
```

The XBJ command allows you to execute a batch stream as a job which is independent of the terminal. This allows you to continue to execute SCI commands interactively in foreground and background modes. Since the batch job is independent of the terminal, the SBS and WAIT commands are not applicable and you will not receive a message when the batch job is complete. Refer to the discussion of XBJ in Section 4 of this manual for detailed information concerning the execution of an SCI batch job.

2.5 USE OF THE INTERACTIVE DEBUGGER

Read the following information concerning the SCI debugger only if you are interested in debugging assembly language programs; otherwise, bypass this portion of the manual. In addition to supporting assembly language debugging, DNOS also supports COBOL and Pascal language debugging as follows:

- The compilers and interpreters for these languages provide error messages that pinpoint syntax errors in the source programs.
- The runtime packages provide error-tracing information in addition to the error messages that describe the nature of the error. Refer to the *DNOS COBOL Programmer's Guide* and the *DNOS TI Pascal Programmer's Guide* for information on debugging COBOL and Pascal programs, respectively.

2.5.1 SCI Debugging Support

Flaws in software are commonly called bugs, and debugging is the process of identifying flaws. Structured programming techniques can reduce the number of bugs in a program; however, the bugs which remain tend to be subtle and hard to find. Several levels of debugging support are provided by DNOS.

- Several SCI commands provide debugging capabilities without requiring a special mode of operation.
- A special mode of operation allows a detailed examination of a single program (task) during the execution process.

All of the debug commands interact with the terminal. Therefore, care must be taken when debugging a program that uses the terminal, since having two processes which request terminal support can cause confusion. If the program being debugged requires the use of a terminal, use two terminals — one for the program and one for debugging.

Information about the SCI debugger and the special mode of operation is provided in the following paragraphs. Refer to the *DNOS Assembly Language Programmer's Guide* for further information on debugging assembly language programs.

2.5.2 Modes of Debugging

There are two sets of debug commands. One set can only be used on controlled tasks which have been put into debug mode through the use of the Execute in Debug Mode (XD) command. The other set of commands can be used on any task. In either case, beware of tasks that unconditionally suspend themselves, since the debug commands can inadvertently reactivate tasks. Refer to Table 2-6 and Table 2-7 for lists of the debug commands for controlled tasks and debug commands for all tasks, respectively. These commands are described in detail in Section 4 of this manual.

Putting a task into controlled mode affects the execution of all debug commands as follows:

- Symbolic expressions can be used in place of integer expressions on commands involving the controlled task.
- Every command requires the controlled task to be unconditionally suspended.

- Every command keeps the controlled task unconditionally suspended.
- Pressing the Command key automatically suspends the controlled task during execution of the following commands: Proceed from Breakpoint (PB), Delete and Proceed from Breakpoint (DPB), and Resume Task (RT).

Table 2-6. SCI Debug Commands for Controlled Tasks

Command	Meaning
ASB	Assign Simulated Breakpoint
DSB	Delete Simulated Breakpoints
LSB	List Simulated Breakpoints
QD	Quit Debug Mode
RST	Resume Simulated Task
ST	Simulate Task
XD	Execute in Debug Mode

Table 2-7. SCI Debug Commands for All Tasks

Command	Meaning
AB	Assign Breakpoint
AT	Activate Task
DB	Delete Breakpoints
DPB	Delete and Proceed from Breakpoint
FB	Find Byte
FW	Find Word
HT	Halt Task
LB	List Breakpoints
LM	List Memory
LSM	List System Memory
MIR	Modify Internal Registers
MM	Modify Memory
MSM	Modify System Memory
MWR	Modify Workspace Registers
PB	Proceed from Breakpoint
RT	Resume Task
SIR	Show Internal Registers
SP	Show Panel
SV	Show Value
SWR	Show Workspace Registers
XHT	Execute and Halt Task

2.5.3 Unconditional Suspend

Most of the debugging commands require the unconditional suspension of the task being debugged either before or during the debug command. The unconditionally-suspended task state under DNOS (task state 6) is the state in which the task is dormant until activated by a command. There are several ways for a task to become unconditionally suspended:

- The task is bid with the suspend option selected by the .DBID primitive, the Execute and Halt Task (XHT) command, or the supervisor call (SVC).

  The .DBID primitive is used for tasks that interface through SCI, such as command processors (which are normally bid using the .BID and .QBID primitives) and high-level language programs (for example, COBOL programs). When the .DBID primitive is executed through SCI, the task is bid and immediately placed in a suspended state. The run ID of the task is saved in the synonym $$BT, or it can be obtained by issuing the Show Task Status (STS) command.

  The XHT command is used for tasks that are normally executed directly by an Execute Task (XT) command. XHT places the task in a suspended state for debugging and displays the run ID of the task to the user. If you want to execute and halt the task and simultaneously place it in controlled mode, the XD command can be used with no input for the RUN ID prompt. The XD command performs the XHT and saves the run ID as the default value for the debugger commands.

  Refer to the *DNOS Systems Programmer's Guide* for information on .BID, .DBID, and .QBID primitives. The STS, XD, and XHT commands are discussed in Section 4 of this manual.

  Refer to the *DNOS Supervisor Call (SVC) Reference Manual* for information on supervisor calls.

- The task suspends itself by an SVC.

- The task executes a breakpoint (XOP 15,15).

- The task is suspended by an SCI debug command. Once the task has been placed in a suspended state, the debugger can be used to assign breakpoints, simulate execution, display memory, and perform other debugging functions. When the debugging session is over, the task can be terminated via the Kill Task (KT) command. If the task was put into controlled mode by an XD command, it can be killed by responding YES to the KILL TASK? prompt of the Quit Debug Mode (QD) command. The KT and QD commands are discussed in Section 4 of this manual.

2.5.4 Symbols

The debug support provided allows for symbolic addressing which allows you to specify labels within the task being debugged rather than memory addresses. This allows for convenient debugging since the assembly language source code list can be used as reference for the symbolic labels used. Symbolic constants consist of the Link Edit phase name, a period (.), the module identifier name (IDT), a period (.), and the symbol, which is an assembly language label. The syntax is defined as:

phasename.IDTname.symbol

To have full symbolic capability, both the assembler and Link Editor must use the Symbolic Memory Table (SYMT) option. If the assembler does not use the SYMT option, but the Link Editor does, then symbols of the following form are available:

phasename.IDTname

If either the phase name or the IDT name of a symbol is omitted, the immediately previous corresponding value is used. The syntax is as follows:

.IDTname.symbol	(no phasename)
phasename..symbol	(no IDTname)
..symbol	(no phasename or IDTname)

The following are examples of debug syntax:

PHASE1.MOD1.XYZ	References:	Phase =	PHASE1
		IDT =	MOD1
		Label =	XYZ
.MOD2.MNO	References:	Phase =	PHASE1
		IDT =	MOD2
		Label =	MNO
..ABC	References:	Phase =	PHASE1
		IDT =	MOD2
		Label =	ABC

Eight bytes of memory per symbol are required to store symbol values.

If the task being debugged is a single routine that was installed without being linked, then a symbolic constant consists of a period (.), the module identifier name, a period (.), and the characters of the symbol as follows:

.IDTname.symbol

As with the linked module, the SYMT option of the assembler must be selected to have full symbolic capability. If the IDT name of a symbolic constant is omitted, the immediately previous corresponding value is used as shown below:

```
.PROG.XYZ
..SYM
```

NOTE

Use symbols only for commands affecting a task that has been placed in the controlled mode by the XD command.

The method used to encode the symbol does not guarantee unique representation of the symbols. An error message appears whenever two symbols are encoded to the same value. The second symbol cannot be used.

2.5.5 Expressions

Constants (and symbolic constants for tasks in the controlled mode) can be combined using the operators +, −, *, /, <, >, and ( ) to form expressions that can be used as command operands. The operators have the following meanings:

+	unary plus or addition
−	unary minus or subtraction
*	multiplication
/	division
( )	evaluation order
<>	the contents of the indicated memory location

NOTE

The right angle bracket (>), used by itself, indicates a hexadecimal number.

Expressions are evaluated according to the following rules:

- Subexpressions delimited by ( ) and < > are evaluated first, with the innermost expression evaluated before any other levels.
- Unless indicated otherwise by parentheses or angle brackets, unary + and − are evaluated first, multiplication and division are evaluated second, and addition and subtraction last.
- For operators at the same level, evaluation proceeds left to right.

For example, if .IDTNAM.BEGIN is memory address >7A and if memory address >7F contains >3B, then the expression:

```
>FF/(<.IDTNAM.BEGIN +5> + -2 +3 * >F)
```

is evaluated as follows:

```
>FF/(<.IDTNAM.BEGIN+5> + -2 +(3 * >F))
>FF/(< >7A +5> + -2 +(3 * >F))
>FF/(< >7F> + -2 + (3 * >F))
>FF/(>3B + (-2) + >2D)
>FF/(>39 +>2D)
>FF/>66
2
```

Several special symbols are allowed in full expressions when the debugger is in controlled mode. These special symbols are:

#PC — represents the contents of the program counter

#WP — represents the contents of the workspace pointer

#ST — represents the contents of the status register

#Rn — represents the contents of the nth workspace register, with n having a value of 0 through 15

A full expression is composed of decimal or hexadecimal integers and the operators +, -, *, /, <, >, and ( ). When the debugger is in controlled mode, symbolic names and the special symbols are permitted in the full expression.

Character strings are also allowed in expressions. A character string is of the form 'XXXX' where X is any valid ASCII character. Use a pair of apostrophes to represent one apostrophe in a character string. A character string can be any length, but only the leftmost four characters are significant. Strings shorter than four characters are right-justified and zero filled to the left. The value of a character string is an expression in the ASCII hexadecimal representation of the characters expressed as a 32-bit number. The following are examples of character strings and their values:

String	Value
'ABCD'	41424344
'A'	00000041
'ABCDE'	41424344
''	00000000
'A''B'	00412742

These symbols can be used in expression lists in the same way as constants or symbolic constants. For example, the following is a valid expression:

```
#PC + NAME.IDT - #R15
```

3

SCI Command Groups and Menus

3.1 INTRODUCTION

DNOS SCI commands can be grouped by their general functions. The SCI default main menu displays the names of the command groups. By entering a command group menu name in response to the SCI prompt symbol ([ ]), you can view the associated commands of a group.

The following paragraphs discuss command groups and menus.

3.2 SCI COMMAND GROUPS

Most SCI commands are unique to one command group, but several commands are duplicated in different groups. Command menus display the command groups and the commands which make up a group. Figure 3-1 shows the predefined SCI main menu, and Table 3-1 lists the commands for each group.

```
************************************************
**       T E X A S   I N S T R U M E N T S    **
**       D N O S   S Y S T E M   1 . 2 . 0    **
************************************************

Command Groups:

/DEBUG  - Interactive Debugger          /OPER   - Operator Interface
/DEVICE - I/O Devices                   /PREXEC - Program Execution
/DIR    - Directories                   /PRFILE - Program Files
/EDIT   - Text Editor                   /SECURE - File Security
/FILE   - File Management               /SPOOL  - Spooler
/JOB    - Job Management                /STATUS - Status Reports
/LANG   - Language Support              /SYSCON - System Configuration
/LUNO   - Logical Unit Numbers          /SYSGEN - System Generation
/MSG    - Message Facilities            /SYSMGT - System Management
/NAME   - Synonyms and Logical Names    /TPD    - Teleprinter Devices
/NET    - Networking Support            /VOLUME - Disk Volumes

       []
```

Figure 3-1. SCI Main Menu

Table 3-1. Commands of SCI Groups

Command Group	Command Group Menu Name	SCI Command	Command Name
Interactive Debugger	/DEBUG	AB	Assign Breakpoint
		ABP	Assign Breakpoint—Pascal
		ASB	Assign Simulated Breakpoint
		AT	Activate Task
		DB	Delete Breakpoints
		DBP	Delete Breakpoint—Pascal
		DPB	Delete and Proceed from Breakpoint
		DPBP	Delete and Proceed from Breakpoint—Pascal
		DSB	Delete Simulated Breakpoints
		FB	Find Byte
		FW	Find Word
		HT	Halt Task
		LB	List Breakpoints
		LBP	List Breakpoints—Pascal
		LM	List Memory
		LPS	List Pascal Stack
		LSB	List Simulated Breakpoints
		LSM	List System Memory
		MIR	Modify Internal Registers
		MM	Modify Memory
		MSM	Modify System Memory
		MWR	Modify Workspace Registers
		PB	Proceed from Breakpoint
		PBP	Proceed from Breakpoint—Pascal
		QD	Quit Debug Mode
		RT	Resume Task
		RST	Resume Simulated Task
		SIR	Show Internal Registers
		SP	Show Panel
		SPS	Show Pascal Stack
		ST	Simulate Task
		SV	Show Value
		SWR	Show Workspace Registers
		XD	Execute in Debug Mode
		XHT	Execute and Halt Task

Table 3-1. Commands of SCI Groups (Continued)

Command Group	Command Group Menu Name	SCI Command	Command Name
I/O Devices	/DEVICE	CC	Copy/Concatenate
		CIC	Create IPC Channel
		CSM	Copy/Verify Sequential Media Utility
		DIC	Delete IPC Channel
		LDC	List Device Configuration
		MAD	Modify Absolute Disk
		MADU	Modify Allocatable Disk Unit
		MD	Map Disk
		MDS	Modify Device State
		RCRU	Read Contents of Specified CRU Address
		SAD	Show Absolute Disk
		SADU	Show Allocatable Disk Unit
		STI	Show Terminal Information
		WCRU	Write Value to Specified CRU Address
Directories	/DIR	BD	Backup Directory
		BDD	Backup Directory to Device
		CB	Create Batch Stream
		CD	Copy Directory
		CFDIR	Create Directory File
		DD	Delete Directory
		LD	List Directory
		MD	Map Disk
		RD	Restore Directory
		VB	Verify Backup
		VC	Verify Copy
Text Editor	/EDIT	CL	Copy Lines
		DL	Delete Lines
		DS	Delete String
		FS	Find String
		IF	Insert File
		MHR	Modify Horizontal Roll
		ML	Move Lines
		MR	Modify Roll
		MRM	Modify Right Margin
		MT	Modify Tabs
		QE	Quit Edit
		RE	Recover Edit
		RS	Replace String
		SL	Show Line
		SVL	Save Lines
		XE	Execute Text Editor
		XES	Execute Text Editor with Scaling

Table 3-1. Commands of SCI Groups (Continued)

Command Group	Command Group Menu Name	SCI Command	Command Name
File Management	/FILE	AA	Add Alias to Pathname
		AF	Append File
		CC	Copy/Concatenate
		CF	Create File
		CFIMG	Create Image File
		CFKEY	Create Key Indexed File
		CFPRO	Create Program File
		CFREL	Create Relative Record File
		CFSEQ	Create Sequential File
		CKR	Copy KIF Randomly
		CKS	Copy KIF to Sequential File
		CSK	Copy Sequential File to KIF
		CSM	Copy/Verify Sequential Media Utility
		DA	Delete Alias from Pathname
		DF	Delete File
		IBMUTL	IBM Diskette Conversion Utility
		LLR	List Logical Record
		MD	Map Disk
		MFN	Modify File Name
		MFP	Modify File Protection
		MKF	Map Key Indexed File
		MKL	Modify KIF Logging
		MRF	Modify Relative to File
		PF	Print File
		SF	Show File
		SRF	Show Relative to File
		TFFPC	Transfer File from PC
		TFTPC	Transfer File to PC
Job Management	/JOB	HJ	Halt Job
		KJ	Kill Job
		LJ	List Jobs
		MJP	Modify Job Priority
		Q	Quit SCI
		RJ	Resume Job
		SJS	Show Job Status for < user ID>
		XBJ	Execute Batch SCI Job
		XJ	Execute Job
		XJM	Execute Job Monitor

Table 3-1. Commands of SCI Groups (Continued)

Command Group	Command Group Menu Name	SCI Command	Command Name
Language Support	/LANG	Assembler:	
		XMA	Execute Macro Assembler
		Basic:	
		BASIC	Execute BASIC
		COBOL:	
		Commands in Menu /COBOL	
		COBOL Program Generator:	
		Commands in Menu /CPG	
		DBMS:	
		Commands in Menu /DBMS	
		Data Dictionary:	
		Commands in Menu /DD	
		FORTRAN-78:	
		Commands in Menu /FORTRN	
		Link Editor:	
		XLE	Execute Link Editor
		QUERY:	
		Commands in Menu /QUERY	
		RPG:	
		Commands in Menu /RPG	
		Sort/Merge:	
		XBSM	Execute Batch Sort/Merge
		XSM	Execute Sort/Merge

Table 3-1. Commands of SCI Groups (Continued)

Command Group	Command Group Menu Name	SCI Command	Command Name
		TIFORM:	
		FORMTSTR	Form Tester
		XFDLC	Execute Forms Description Language Compiler
		XISGE	Execute Interactive Screen Generator and Editor
		TIP: Commands in Menu /TIP	
		TIPE: Commands in Menu /TIPE	
Logical Unit Numbers	/LUNO	AGL	Assign Global LUNO
		AL	Assign LUNO
		BL	Backspace LUNO
		FL	Forward Space LUNO
		MLP	Modify LUNO Protection
		RAL	Release All Logical Units
		RGL	Release Global LUNO
		RL	Release LUNO
		RWL	Rewind LUNO
		SIS	Show I/O Status
		WEOF	Write EOF to LUNO
Message Facilities	/MSG	BEMF	Build Expanded Message File
		BMF	Build Message File
		CM	Create Message
		COM	Create Operator Message
		MSG	Display Message
		SEM	Show Expanded Message
Synonyms and Logical Names	/NAME	ALN	Assign Logical Name
		AS	Assign Synonym Value
		LLN	List Logical Names
		LS	List Synonyms
		MS	Modify Synonym
		Q$SYN	Clear Synonyms
		RLN	Release Logical Name
		SGND	Snapshot Global Name Definitions
		SND	Snapshot Name Definitions

Table 3-1. Commands of SCI Groups (Continued)

Command Group	Command Group Menu Name	SCI Command	Command Name
Networking Support	/NET	TI PC File	Transfer Package:
		TFFPC	Transfer File from PC
		TFTPC	Transfer File to PC
		DNCS Products: Commands in Menu /DNCS	
		3270 ICS: Commands in Menu/ICS	
Operator Interface	/OPER	COM	Create Operator Message
		KOM	Kill Operator Messages
		KOR	Kill Operator Interface Request
		LOM	List Operator Messages
		QOI	Quit Operator Interface
		ROM	Receive Operator Messages
		ROR	Respond to Operator Interface Request
		XOI	Execute Operator Interface
Program Execution	/PREXEC	AT	Activate Task
		BATCH	Begin Batch Execution
		CB	Create Batch Stream
		EBATCH	End Batch Execution
		EC	Error Counter
		HT	Halt Task
		KBT	Kill Background Task
		KT	Kill Task
		RT	Resume Task
		WAIT	Wait for Background Task to Complete
		XB	Execute Batch
		XHT	Execute and Halt Task
		XT	Execute Task
		XTS	Execute Task and Suspend SCI

Table 3-1. Commands of SCI Groups (Continued)

Command Group	Command Group Menu Name	SCI Command	Command Name
Program Files	/PRFILE	CFPRO	Create Program File
		CPI	Copy Program Image
		DO	Delete Overlay
		DP	Delete Procedure Segment
		DPS	Delete Program Segment
		DT	Delete Task Segment
		IO	Install Overlay
		IP	Install Procedure Segment
		IPS	Install Program Segment
		IRT	Install Real-Time Task Segment
		IT	Install Task Segment
		MOE	Modify Overlay Entry
		MPE	Modify Procedure Segment Entry
		MPF	Map Program File
		MPI	Modify Program Image
		MSE	Modify Program Segment Entry
		MTE	Modify Task Segment Entry
		MTSA	Modify Task Security Attribute
		SPI	Show Program Image
File Security	/SECURE	User Commands:	
		LAG	List Access Groups
		LAGFR	List Access Group File Rights
		LSAR	List Security Access Rights
		MPC	Modify Passcode
		MSAR	Modify Security Access Rights
		SCAG	Set Creation Access Group
		Access Group Leader Commands:	
		CAG	Create Access Group
		DAG	Delete Access Group
		LAGM	List Access Group Members
		MAG	Modify Access Group
		Security Manager Commands:	
		AUI	Assign User ID
		DUI	Delete User ID
		LUI	List User IDs
		MTSA	Modify Task Security Attribute
		MUI	Modify User ID
		MVS	Modify Volume Security

Table 3-1. Commands of SCI Groups

Command Group	Command Group Menu Name	SCI Command	Command Name
Spooler	/SPOOL	HO	Halt Output at Device
		KO	Kill Output at Device
		MO	Modify Output at Device
		MSD	Modify Spooler Device
		PF	Print File
		RO	Resume Output at Device
		SOS	Show Output Status
Status Reports	/STATUS	LSC	List Software Configuration
		LTS	List Terminal Status
		SBS	Show Background Status
		SCS	Show Channel Status
		SDT	Show Date and Time
		SIS	Show I/O Status
		SJS	Show Job Status for < User ID>
		SMM	Show Memory Map
		SMS	Show Memory Status
		SOS	Show Output Status
		STI	Show Terminal Information
		STS	Show Task Status for < User ID>
		SVS	Show Volume Status
		XJM	Execute Job Monitor
		XPD	Execute Performance Display
System Configuration	/SYSCON	ISL	Initialize System Log
		LDC	List Device Configuration
		MCC	Modify Country Code
		MCDT	Modify Command Definition Table
		MDC	Modify Device Configuration
		MDS	Modify Device State
		MSP	Modify Scheduler/Swap Parameters
		MST	Modify System Table Sizes
		QSCU	Quit Configuration Utility Session
		SCC	Show Country Code
		SCDT	Show Command Definition Table
		XSCU	Execute System Configuration Utility

Table 3-1. Commands of SCI Groups (Continued)

Command Group	Command Group Menu Name	SCI Command	Command Name
System Generation	/SYSGEN	ALGS	Assemble and Link Generated System
		CP	Create Patch
		IGS	Install Generated System
		PGS	Patch Generated System
		TGS	Test Generated System
		XSGU	Execute System Generation Utility
System Management	/SYSMGT	Disk Recovery:	
		CKD	Check Disk for Consistency
		IDS	Initialize Disk Surface
		RVI	Recover Volume Information
		SD	Scan Disk
		Initialization:	
		CSF	Create System Files
		IDT	Initialize Date and Time
		MDS	Modify Device State
		MTS	Modify Terminal Status
		System Crash Analysis:	
		XANAL	Execute Crash Analysis Utility
		User ID Handling:	
		AUI	Assign User ID
		DUI	Delete User ID
		LUI	List User IDs
		MPC	Modify Passcode
		MUI	Modify User ID
Teleprinter Devices	/TPD	CALL	Call Terminal
		ANS	Answer Incoming Call
		DISC	Terminal Disconnection
		MHPC	Modify Hardcopy Terminal Port Characteristics
		LHPC	List Hardcopy Terminal Port Characteristics

Table 3-1. Commands of SCI Groups (Continued)

Command Group	Command Group Menu Name	SCI Command	Command Name
Disk Volumes	/VOLUME	CKD	Check Disk for Consistency
		CRV	Check and Reset Volume
		CV	Copy Volume
		CVD	Copy and Verify Disk
		DCOPY	Disk Copy/Restore
		IDS	Initialize Disk Surface
		INV	Initialize New Volume
		IV	Install Volume
		MD	Map Disk
		MVI	Modify Volume Information
		MVS	Modify Volume Security
		RVI	Recover Volume Information
		SD	Scan Disk
		UV	Unload Volume

3.3 SCI COMMAND GROUP PURPOSE

The following paragraphs discuss the purpose of each SCI command group and give references to other manuals where additional information on the command groups is available. Refer to Table 3-1 for a list of commands contained in each group.

3.3.1 Interactive Debugger Group

The interactive debugger commands provide support for correcting a program. This command group includes commands which execute and terminate the debug mode, display the debug panel for specific tasks, and evaluate expressions for debugging purposes. Commands in this group enable you to modify and manipulate breakpoints, tasks, system memory and registers, and search for specific values in memory areas. Discussions on this command group can be found in the *DNOS Assembly Language Programmer's Guide*.

3.3.2 I/O Devices Group

The I/O device commands pertain to the use and operation of online input and output devices. These commands allow you to perform operations on mounted media, manipulate I/O devices, and display and modify system device information. Discussions on this command group can be found in this manual.

3.3.3 Directories Group

The directory commands enable you to perform copy, verify, and restore functions at the directory level. This command group also allows you to manipulate directories and list information contained on the disk volume or within specific directories. Discussions on this command group can be found in the *DNOS Operations Guide*.

3.3.4 Text Editor Group

The Text Editor commands allow you to modify the contents of source files and manipulate the terminal display with horizontal and vertical rolls of the screen. These commands are accessible only after activating the Text Editor. Discussions on this command group can be found in the *DNOS Text Editor Reference Manual.*

3.3.5 File Management Group

The file management commands maintain user data with the ability to create relative record files, program files, image files, key indexed files, and sequential files. Files and filenames can be copied, modified and manipulated to meet your needs. This command group also contains SCI commands which enable you to display and print files and list the file information in a disk map. Discussions on this command group can be found in the *DNOS Operations Guide.*

3.3.6 Job Management Group

The job management commands control jobs executing on the DNOS system. These commands allow you to list and manipulate job priority and job execution. Discussions on this command group can be found in the *DNOS Operations Guide.*

3.3.7 Language Support Group

The language support commands enable you to perform assembly, compile, and link operations, to execute programs and tasks, and to use various productivity tools. Several languages have command menus which can be displayed in place of the DNOS main menu while using the language commands. Refer to the language manuals for details of commands in specific language command groups.

3.3.8 Logical Unit Numbers Group

The logical unit numbers commands allow DNOS to perform input and output to logical units (LUNOs) rather than directly to the resources the LUNOs represent. These commands assign, release, and manipulate the resources to which the LUNOs are assigned. Discussions on this command group can be found in the *DNOS Operations Guide.*

3.3.9 Message Facilities Group

Message facilities include user interfaces to the system operator, other users, and the error handling facilities. Discussions on this command group can be found in the *DNOS Operations Guide* and the *DNOS Messages and Codes Reference Manual.*

3.3.10 Synonyms and Logical Names Group

The DNOS system enables the use of synonyms and logical names which access I/O resources. This command group assigns and lists synonyms and logical names. Synonyms can also be modified and deleted; logical names must be released. Discussions on this command group can be found in the *DNOS Operations Guide.*

3.3.11 Networking Support Group

The networking support group lists commands and submenus of commands for networking products on DNOS. Refer to the specific networking product manual for additional information.

3.3.12 Operator Interface Group

The operator interface commands send messages to the system operator when operator intervention is necessary. The responsibility for responding to the operator messages can be the responsibility of a designated system operator or shared between the system users. Discussions on this command group can be found in the *DNOS Operations Guide.*

3.3.13 Program Execution Group

The program execution group contains commands associated with batch streams during creation, execution, and termination. Other commands display and control program activity and errors encountered during activation. Discussions on this command group can be found in the *DNOS Operations Guide*.

3.3.14 Program Files Group

Program file commands create and maintain DNOS program files. These file commands can modify, delete, and install procedures, program segments, tasks, and overlays within a program file. Discussions on this command group can be found in the *DNOS Assembly Language Programmer's Guide*.

3.3.15 File Security Group

Three general groups of commands are available. A set of user commands allow any user to examine access rights and set up his or her own security environment. Commands for access group leaders allow those users to create and manage access groups. Commands in the security manager group allow the security manager to control the system's overall security. Discussions on this command group can be found in the *DNOS Operations Guide* and the *DNOS Security Manager's Guide*.

3.3.16 Spooler Group

The spooler commands pertain to the use and operation of online output devices. These commands allow you to print files, manipulate output devices, examine output status, and modify output status at a device. Discussions on this command group can be found in the *DNOS Operations Guide*.

3.3.17 Status Reports Group

Status report commands enable you to obtain the status of programs executing on a DNOS system, the status of the whole operating system, and the status of devices attached to the system. Discussions on this command group can be found in the *DNOS Operations Guide*.

3.3.18 System Configuration Group

System configuration commands allow you to examine and modify a system configuration. Discussions on this command group can be found in the *DNOS System Generation Reference Manual*.

3.3.19 System Generation Group

System Generation commands allow you to generate, patch, test, and install a new system. Discussions on this command group can be found in the *DNOS System Generation Reference Manual*.

3.3.20 System Management Group

System management commands allow you to perform required functions prior to system operation, to analyze a system crash, to maintain user IDs, and to recover disks. Discussions on this command group can be found in the *DNOS Operations Guide* and the *DNOS Systems Programmer's Guide*.

3.3.21 Teleprinter Devices Group

The teleprinter device utilities commands establish connections, monitor completion of connections, and terminate connections between terminals. Other commands list the port characteristics of the hardcopy terminal ports and allow modifications to those characteristics.

3.3.22 Disk Volumes Group

Disk volume commands inspect and modify the contents of a disk medium, prepare a disk medium for storing data, and copy the contents of one disk to another disk. Discussions on this command group can be found in the *DNOS Operations Guide*.

3.4 SCI MENUS

The main menu is the list of commands or submenus that appears on your terminal after you log on to the system. Submenus listed on this menu are names of submenus which can also be displayed on your terminal. Although DNOS supplies its own main menu, you can alter this menu or create a new one to meet your requirements.

The following paragraphs explain how to modify, create, and display main menus and submenus.

3.4.1 Displaying Menus

The default SCI main menu is displayed when the terminal is in VDT mode, SCI is in command mode, and there is no foreground activity in progress.

If the text of a menu is to be displayed, enter only the menu name in response to the SCI prompt symbol ([ ]). For example, to display the commands for the /VOLUME (Disk Volumes) submenu (Table 3-1), respond to SCI as follows:

```
[] /VOLUME
```

3.4.2 Creating Menus

The DNOS SCI menus shown in Table 3-1 are located within the .S$CMDS system directory. The .S$CMDS directory is the SCI primary command procedure library with the default main menu in the file named .M$LC. The .M$LC file is composed of text that displays submenus. Access to show and/or edit these files is by .S$CMDS. M$name, where name is a menu name contained in the .S$CMDS library without the slash (/). By editing the .M$LC file, you control what is displayed as the default main menu. To edit the text of a submenu (for example, PREXEC from Table 3-1) enter .S$CMDS.M$PREXEC in response to the FILE ACCESS NAME prompt of the XE command.

If you want to set up your own menu scheme of main and submenus to be displayed in place of the ones supplied by DNOS, create a directory to be used as the primary command procedure library. This directory should contain .M$LC and other .M$ files, as in the SCI primary command procedure library .S$CMDS, to remain consistent with DNOS SCI menu file names. For example, the text of the .M$LC file should consist of the file names which contain the submenus.

Enter the .USE primitive interactively or within a user-defined command procedure to specify that the primary command procedure library is not .S$CMDS. If the default main menu is not contained in a file named .M$LC within the your library, enter the .OPTION primitive to specify the file to be used for that purpose. The following example enters the .USE and .OPTION primitives to specify a user-defined primary command procedure library and the file to be used as the default main menu, respectively:

```
[] .USE .MYDIR.PROC,.S$CMDS
[] .OPTION MENU=MAIN
```

In this example, the default main menu is located in the file .MYDIR.PROC.M$MAIN.

The .USE primitive specifies that the first library entered is used as the primary command procedure library instead of .S$CMDS, and that .S$CMDS is used as the secondary library. As many as five command procedure libraries can be specified with .USE.

The .OPTION primitive specifies the file used as the default main menu. The user-defined menu is then displayed instead of the SCI default main menu. This menu is displayed when the terminal is in command mode until the .USE and/or .OPTION primitives are entered. These primitives can be entered interactively or in a command procedure, specifying that another library is to be used as the primary command procedure library, and/or another file is to be used as the default main menu, respectively.

The .OPTION primitive can also be used to specify that lowercase characters are to be accepted for command entry in interactive mode. This is specified by saying .OPTION LOWERCASE = YES. The .USE and .OPTION primitives can be entered in the preceding format in any command procedure. Further details on this primitive and on others available with SCI can be found in the *DNOS Systems Programmer's Guide.*

3.5 SPECIAL SCI COMMAND PROCEDURES

There are three command procedures that are invoked at specific points during SCI execution:

- M$00 - invoked at terminal log-on
- M$01 - invoked at terminal log-off
- M$02 - invoked when a question mark (?) is entered in response to an error message displayed on a terminal

These command procedures are not required for proper SCI execution, however, they can be adapted to fulfill your particular needs. The following paragraphs discuss these command procedures and their functions.

3.5.1 M$00 Command Procedure

SCI invokes the M$00 command procedure after the terminal log-on requirements are satisfied and the .S$NEWS file is displayed. It is also invoked each time a batch SCI is started by the XB command or the XBJ command. You can use this procedure to establish a particular SCI environment prior to command execution.

For example, the M$00 command procedure can bid an accounting program that executes at each user's terminal after log-on. M$00 can also check the status of mounted disks at log-on and return the information to the user. Another use of the M$00 command procedure is to determine the different primary command libraries accessible with each user ID. On the basis of the user ID entered during log-on, M$00 can contain the .OPTION and .USE primitives to set up appropriate environments.

3.5.2 M$01 Command Procedure

The M$01 command procedure is invoked when terminal log-off occurs or when a batch stream terminates. This command procedure can be used for any process that you want.

You can use the M$01 procedure to gather a variety of information. For example, M$01 can bid an accounting program (as mentioned with the M$00 command procedure) which records information for each user's terminal from the time of log-on to log-off.

3.5.3 M$02 Command Procedure

SCI invokes the M$02 command procedure when you enter a question mark (?) in response to an error message displayed on your terminal. If you do not enter the question mark after the error message, this command procedure is not invoked.

After you enter the question mark, the M$02 procedure calls the Show Expanded Message (SEM) command processor and displays detailed error information on your terminal.

Do not delete the content of this procedure as delivered if you are going to use the question mark feature.

3.6 ASSIGNMENT OF COMMANDS TO FUNCTION KEYS

You can make use of the function keys on a VDT to quickly access an SCI command. You can set as many as 8 function keys on a 911 and as many as 12 on a 940 or 931 VDT. For each function key you wish to enable, assign the synonym F$$xy, where xy is the function key number. The value assigned should be a command name in .S$CMDS or in any of your command directories. For example, if you use the Modify Synonym (MS) command as follows:

```
MODIFY SYNONYM
      SYNONYM: F$$05
        VALUE: SF
```

Then while using SCI, you can press the F5 function key to issue the SF command.

4

DNOS SCI Commands

4.1 INTRODUCTION

This section describes the DNOS SCI commands in alphabetical order. An example is given for most commands in their description. The SCI commands allow you to interface with DNOS interactively or via batch streams. You can enter these commands via all DNOS interactive devices including teleprinter devices and video display terminals (VDTs).

Each command description specifies the legal response types that you can enter for each particular prompt. These response types are listed and defined in Table 4-1. Legal device names that can be entered for the devicename response type, from Table 4-1, are listed in Table 4-2.

Table 4-1. Response Type Indicators

Type	Definition
Pathname	Name used to access an I/O resource. A pathname can include channel name, device name, file name or station name. You can also specify a pathname by a synonym, a synonym followed by a pathname (synonym.pathname), a logical name, or a logical name followed by a pathname (logical name.pathname). Legal characters in pathnames include uppercase alphabetic characters (A through Z), numerals (0 through 9), periods (.), dollar sign ($), left bracket ([ ), right bracket ( ]), and back slash (\). The name must start with an alphabetic character.
Devicename	Name of a device (for example, DS01 and ST01).
Filename	Pathname of a file. A concatenation of a disk or volume name, any directory-level names leading to a file, and a final component name of a file. You can also specify a file by a synonym, a synonym followed by a file name (synonym.filename), a logical name, or a logical name followed by a file name (logical name.filename).
Site	Name of a computer installation in a DNOS network. A site name can be from 1 to 8 alphanumeric characters beginning with an alphabetic character. You must have the Distributed Network I/O (DNIO) package installed to be able to use a site name.
Stationname	A station ID in the range of ST01 through ST99. You can obtain the station ID of your station by entering the Show Terminal Information (STI) command.
YES/NO	The response to a prompt can be YES, NO, Y, or N.

Table 4-1. Response Type Indicators (Continued)

Type	Definition
Integer	Hexadecimal or decimal digits. Precede hexadecimal numbers by entering a right angle bracket (>) or a leading zero.
Integer exp	Decimal or hexadecimal value or expression composed of decimal or hexadecimal integers and the operators add (+), subtract (−), multiply (*), and divide (/).
Integer exp list	A list of decimal or hexadecimal values or expressions, separated by commas.
Full exp	Integer expression with the additional operators ( ) to indicate evaluation order and < > to show contents of indicated memory address registers. String operands are also permitted. In the debugger controlled mode, symbolic names and the symbols #PC, #WP, #ST, and #R0, #R1, #R2, . . .,#R15 are permitted. This type is unique to the SCI debugger.
Full exp list	More than one full expression separated by commas.
Alphanumeric	A string of alphabetic and/or numeric characters or a dollar sign ($), starting with an alphabetic character. (Used with user IDs, volume names, and so on.)
Character(s)	Letters, digits, punctuation marks, or other symbols restricted in format or content by the context.
Line number	Numeric expression composed of an integer or integers and the operators plus (+) and minus (−).

Table 4-2. Legal Device Names

Device Name	Device
CMxx	A communication device specified by a device name in the range of CM01 through CM99.
CRxx	A card reader specified by a device name in the range of CR01 through CR99.
CSxx	A cassette unit of a Model 733 ASR terminal specified by a device name in the range of CS01 through CS99.
DSxx	A disk unit or an FD1000 diskette unit, specified in the range of DS01 through DS99.
DUMY	A dummy device to which output can be sent. All normal processing is done, but no output is generated.
ME	The terminal at which a command is entered. You can find the station ID of your terminal by entering the Show Terminal Information (STI) command.
MTxx	A magnetic tape unit specified by a device name in the range of MT01 through MT99.
LPxx	A line printer specified by a device name in the range of LP01 through LP99.
SDxx	A user-configured special device in the range of SD01 through SD99.
STxx	A terminal specified by a device name in the range of ST01 through ST99. You can find the station ID of your terminal by entering the Show Terminal Information (STI) command. The device name STxx, where xx is your terminal ID number, is equivalent to entering ME.

For convenience, many commands have initial values and default values for various prompts. An initial value is a value that SCI automatically displays as a response to a field prompt. You can accept the initial value by pressing the Return key or delete it by pressing the Erase Field or Skip key. You can also replace the initial value by entering a different value.

The initial value for a field prompt can be set in the command procedure so that the same value always appears as the initial value for that field prompt. In other cases, SCI saves the value entered with a command and displays it as the initial value for a later entry of the same command or for the entry of a related command.

A default value is a value that the command processor automatically supplies when you do not enter a response to a field prompt. The command processor often provides default values to speed up entry of responses to field prompts. This is especially true of optional user responses.

When a default value exists, you can press the Return key without entering a response. Such an entry is called a null entry. When you supply a null entry to an optional user response prompt, the default value is used as the field prompt response.

If you do not want the default value or the initial value, you can supply a value as the response for a field prompt.

The notations shown in Table 4-3 accompany some prompt responses in the command descriptions to define how DNOS expects the response types to be entered.

Table 4-3. Field Prompt Notation

Notation	Meaning
Uppercase	Enter the response as listed.
Lowercase	Enter a response of this type.
No marks	The response is required.
[ ]	The response is optional.
{ }	The response must be exactly one of the enclosed items or must be of a type that is one of the enclosed items (the choices are separated by a slash).
Item...item	More than one item of this type can be entered to the response. Separate items by commas.
@	Synonyms are allowed as responses.
( )	Represents the initial value. If (*) is shown, the value can be supplied from a synonym set by a previously used command procedure.
/	Indicates alternate response types.
-	When a hyphen (-) precedes a valid field prompt type, the initial and user responses are not echoed to the screen. In a batch stream the response is replaced by four dashes.

The format of command descriptions is consistent from command to command. Each description is broken into categories to aid you in referencing particular command information. The categories of command descriptions are listed in Table 4-4.

Table 4-4. Command Description Format

Category	Description
Command	This category includes the command keyword and its full descriptive name.
Prompts	This category includes the set of prompts for the command, the types of responses accepted, and an indication of any initial values supplied by the command procedure.
Purpose	This category includes a statement of the function performed by the command, its general uses, and rules of use.
Prompt Details	For each of the command prompts, this category describes what is required for the command, how the response can be specified, and any default values used when you do not supply a response.
Example	This category presents an example of how to use the command in a common way, show responses to the prompts, and describes the outcome of the command.
Special Cases	This category is included for commands which have nonobvious applications, such as using the Backup Directory command to copy a single file. Few commands need this category of explanation.
Assumptions	This category includes assumptions not obvious from the statement of purpose or from the prompts of the command. The fact that a certain LUNO must be assigned or that the file or device must be in a certain state are examples of assumptions.
Messages	This category documents special completion messages, output formats, or other nonerror messages. This category does *not* include standard error messages but it does include a description of other messages you must notice.
Notes	This category includes special warnings, cautions, or reminders about the command.
Related Commands	This category lists the commands that perform opposite operations from the command being described (Create File versus Delete File), perform similar operations to the same types of data, or in some fashion are similar to the described command.

Where a category is not needed for a particular command description, that category has been excluded.

Table 4-5 lists each SCI command and the associated privilege level. To execute an SCI command, your user ID must have been assigned a privilege equal to or greater than the command's privilege level. Refer to the Assign User ID (AUI) command for information on assigning user privilege levels.

Table 4-5. Privilege Levels of SCI Commands

Command Keyword	Command Name	Command Privilege Level
AA	Add Alias to Pathname	0
AB	Assign Breakpoint	2
ABP	Assign Breakpoint—Pascal	2
AF	Append File	0
AGL	Assign Global LUNO	0
AL	Assign LUNO	0
ALGS	Assemble and Link Generated System	6
ALN	Assign Logical Name	0
ANS	Answer Incoming Call	0
AS	Assign Synonym Value	0
ASB	Assign Simulated Breakpoint	2
AT	Activate Task	0
AUI	Assign User ID	4
BATCH	Begin Batch Execution	0
BD	Backup Directory	0
BDD	Backup Directory to Device	2
BEMF	Build Expanded Message File	2
BL	Backspace LUNO	0
BMF	Build Message File	2
CAG	Create Access Group	2
CALL	Call Terminal	0
CB	Create Batch Stream	0
CC	Copy/Concatenate	0
CD	Copy Directory	0
CF	Create File	0
CFDIR	Create Directory File	0
CFIMG	Create Image File	2
CFKEY	Create Key Indexed File	0
CFPRO	Create Program File	0
CFREL	Create Relative Record File	0
CFSEQ	Create Sequential File	0
CIC	Create IPC Channel	2
CKD	Check Disk for Consistency	6
CKR	Copy KIF Randomly	0
CKS	Copy KIF to Sequential File	0
CL	Copy Lines	0
CM	Create Message	0
COM	Create Operator Message	0

Table 4-5. Privilege Levels of SCI Commands (Continued)

Command Keyword	Command Name	Command Privilege Level
CP	Create Patch	2
CPI	Copy Program Image	0
CRV	Check and Reset Volume	2
CSF	Create System Files	6
CSK	Copy Sequential File to KIF	0
CSM	Copy/Verify Sequential Media Utility	0
CV	Copy Volume	2
CVD	Copy and Verify Disk	2
DA	Delete Alias from Pathname	2
DAG	Delete Access Group	2
DB	Delete Breakpoints	2
DBP	Delete Breakpoint—Pascal	2
DCOPY	Disk Copy/Restore	2
DD	Delete Directory	2
DF	Delete File	2
DIC	Delete IPC Channel	2
DISC	Terminal Disconnection	0
DL	Delete Lines	0
DO	Delete Overlay	2
DP	Delete Procedure Segment	2
DPB	Delete and Proceed from Breakpoint	2
DPBP	Delete and Proceed from Breakpoint—Pascal	2
DPS	Delete Program Segment	2
DS	Delete String	0
DSB	Delete Simulated Breakpoints	2
DT	Delete Task Segment	2
DUI	Delete User ID	4
EBATCH	End Batch Execution	0
EC	Error Count	0
FB	Find Byte	2
FL	Forward Space LUNO	0
FS	Find String	0
FW	Find Word	2
HJ	Halt Job	0
HO	Halt Output at Device	0
HT	Halt Task	2
IBMUTL	IBM Diskette Conversion Utility	2
IDS	Initialize Disk Surface	2
IDT	Initialize Date and Time	4
IF	Insert File	0
IGS	Install Generated System	6
INV	Initialize New Volume	2
IO	Install Overlay	2
IP	Install Procedure Segment	2
IPS	Install Program Segment	2
IRT	Install Real-Time Task Segment	2
ISL	Initialize System Log	4
IT	Install Task Segment	2

Table 4-5. Privilege Levels of SCI Commands (Continued)

Command Keyword	Command Name	Command Privilege Level
IV	Install Volume	0
KBT	Kill Background Task	2
KJ	Kill Job	0
KO	Kill Output at Device	0
KOM	Kill Operator Messages	0
KOR	Kill Operator Interface Request	0
KT	Kill Task	2
LAG	List Access Groups	0
LAGFR	List Access Group File Rights	0
LAGM	List Access Group Members	2
LB	List Breakpoints	2
LBP	List Breakpoints—Pascal	2
LD	List Directory	0
LDC	List Device Configuration	2
LHPC	List Hardcopy Terminal Port Characteristics	0
LJ	List Jobs	4
LLN	List Logical Names	0
LLR	List Logical Record	2
LM	List Memory	2
LOM	List Operator Messages	0
LPS	List Pascal Stack	2
LS	List Synonyms	0
LSAR	List Security Access Rights	0
LSB	List Simulated Breakpoints	2
LSC	List Software Configuration	2
LSM	List System Memory	2
LTS	List Terminal Status	0
LUI	List User IDs	4
MAD	Modify Absolute Disk	6
MADU	Modify Allocatable Disk Unit	6
MAG	Modify Access Group	2
MCC	Modify Country Code	6
MCDT	Modify Command Definition Table	6
MD	Map Disk	0
MDC	Modify Device Configuration	6
MDS	Modify Device State	6
MFN	Modify File Name	2
MFP	Modify File Protection	2
MHPC	Modify Hardcopy Terminal Port Characteristics	0
MHR	Modify Horizontal Roll	0
MIR	Modify Internal Registers	2
MJP	Modify Job Priority	4
MKF	Map Key Indexed File	0
MKL	Modify KIF Logging	2
ML	Move Lines	0
MLP	Modify LUNO Protection	2
MM	Modify Memory	2
MO	Modify Output at Device	0

Table 4-5. Privilege Levels of SCI Commands (Continued)

Command Keyword	Command Name	Command Privilege Level
MOE	Modify Overlay Entry	2
MPC	Modify Passcode	0
MPE	Modify Procedure Segment Entry	2
MPF	Map Program File	0
MPI	Modify Program Image	2
MR	Modify Roll	0
MRF	Modify Relative to File	6
MRM	Modify Right Margin	0
MS	Modify Synonym	0
MSAR	Modify Security Access Rights	0
MSD	Modify Spooler Device	0
MSE	Modify Program Segment Entry	2
MSG	Display Message	0
MSM	Modify System Memory	6
MSP	Modify Scheduler/Swap Parameters	6
MST	Modify System Table Sizes	6
MT	Modify Tabs	0
MTE	Modify Task Segment Entry	2
MTS	Modify Terminal Status	4
MTSA	Modify Task Security Attribute	4
MUI	Modify User ID	4
MVI	Modify Volume Information	2
MVS	Modify Volume Security	4
MWR	Modify Workspace Registers	2
PB	Proceed from Breakpoint	2
PBP	Proceed from Breakpoint—Pascal	2
PF	Print File	0
PGS	Patch Generated System	6
Q	Quit SCI	0
QD	Quit Debug Mode	2
QE	Quit Edit	0
QOI	Quit Operator Interface	2
QSCU	Quit Configuration Utility Session	6
Q$SYN	Clear Synonyms	0
RAL	Release All Logical Units	2
RCRU	Read Contents of Specified CRU Address	6
RD	Restore Directory	0
RE	Recover Edit	0
RGL	Release Global LUNO	0
RJ	Resume Job	0
RL	Release LUNO	0
RLN	Release Logical Name	0
RO	Resume Output at Device	0
ROM	Receive Operator Messages	0
ROR	Respond to Operator Interface Request	0
RS	Replace String	0
RST	Resume Simulated Task	2
RT	Resume Task	2

Table 4-5. Privilege Levels of SCI Commands (Continued)

Command Keyword	Command Name	Command Privilege Level
RVI	Recover Volume Information	6
RWL	Rewind LUNO	0
SAD	Show Absolute Disk	2
SADU	Show Allocatable Disk Unit	2
SBS	Show Background Status	0
SCAG	Set Creation Access Group	0
SCC	Show Country Code	0
SCDT	Show Command Definition Table	2
SCS	Show Channel Status	2
SD	Scan Disk	6
SDT	Show Date and Time	0
SEM	Show Expanded Message	0
SF	Show File	0
SGND	Snapshot Global Name Definitions	2
SIR	Show Internal Registers	2
SIS	Show I/O Status	0
SJS	Show Job Status for < user ID>	2
SL	Show Line	0
SMM	Show Memory Map	2
SMS	Show Memory Status	0
SND	Snapshot Name Definitions	2
SOS	Show Output Status	0
SP	Show Panel	2
SPI	Show Program Image	2
SPS	Show Pascal Stack	2
SRF	Show Relative to File	6
ST	Simulate Task	2
STI	Show Terminal Information	0
STS	Show Task Status for < user ID>	0
SV	Show Value	0
SVL	Save Lines	0
SVS	Show Volume Status	0
SWR	Show Workspace Registers	2
TFFPC	Transfer File from PC	0
TFTPC	Transfer File to PC	0
TGS	Test Generated System	6
UV	Unload Volume	0
VB	Verify Backup	0
VC	Verify Copy	0
WAIT	Wait for Background Task to Complete	0
WCRU	Write Value to Specified CRU Address	6
WEOF	Write EOF to LUNO	0
XANAL	Execute Crash Analysis Utility	2
XB	Execute Batch	0
XBJ	Execute Batch SCI Job	0
XD	Execute in Debug Mode	2
XE	Execute Text Editor	0
XES	Execute Text Editor with Scaling	0

Table 4-5. Privilege Levels of SCI Commands (Continued)

Command Keyword	Command Name	Command Privilege Level
XHT	Execute and Halt Task	2
XJ	Execute Job	0
XJM	Execute Job Monitor	4
XLE	Execute Link Editor	2
XMA	Execute Macro Assembler	2
XOI	Execute Operator Interface	2
XPD	Execute Performance Display	2
XSCU	Execute System Configuration Utility	6
XSGU	Execute System Generation Utility	6
XT	Execute Task	0
XTS	Execute Task and Suspend SCI	2

4.2 FILE SECURITY

In a DNOS system that has been generated with the file security option, there are two factors that affect how you can access a file. These factors are the access groups to which you belong and the access rights for those groups for any particular file you wish to use. The *DNOS Security Manager's Manual* describes how to set up a secure environment. In most cases, your security manager will determine what access groups exist in your environment and will assign you to one or more access groups. The security manager or some other group leader may also be responsible for determining which files have what access rights for particular groups.

The commands for creating access groups and allowing various groups to access particular files can be available to you or they can be restricted to the security manager or some select group of users. The access rights to the command procedures, in addition to their privilege level, determine who can use which commands.

In the descriptions that follow, no file security is presumed for access to the commands. If you are unable to issue some command, ask your security manager if that command has been secured for use by a particular group. If you need to use that command, negotiate with your security manager for access.

While using the commands, if you have file security, you will need the appropriate access to files you manipulate with the commands. The access rights available are read, write, delete, execute, and control.

In general, the read access right is needed for a file accessed by a command if that command shows data in the file or examines the file for input. For example, the Show File (SF) command requires that you have read access to the file being shown. If you do not have read access, you will receive an error message from SF.

The write access right is needed for a file accessed by a command that modifies or updates a file. For example, the Append File (AF) command requires access to the file used for OUTPUT PATHNAME. AF also requires read access to the file used as INPUT ACCESS NAME(S).

If you issue a command that deletes a file, you must have the delete access right to that file. Delete File (DF), for example, requires that you have delete access to the file(s) specified for PATHNAMES(S). Since the text editor replaces an existing file with a new one, you need delete access to the file specified for FILE ACCESS NAME if you are replacing that file when using the Execute Text Editor (XE) and Quit Editor (QE) commands.

A command that executes a task from a program file requires that you have the execute access right for that program file. The Execute Task (XT) command, for example, requires that you have execute access to the file specified for PROGRAM FILE OR LUNO.

The control access right is required for any command that changes the access rights to a file. If you want to use the Modify Security Access Rights (MSAR) command, for example, you must have control access to the file specified for FILE NAME.

In the command descriptions that follow, the notes sections tell if there are any unexpected security requirements for files used by the command. In most cases, an understanding of the access rights is sufficient to predict the required access rights to use a particular command.

4.3 NETWORK ACCESS WITH SCI COMMANDS

The Distributed Network I/O (DNIO) package must be installed for you to access networks of DNOS systems. When you are using DNOS in a network of systems, you can log on to another system using a feature called network log-on. Network log-on allows you to execute any SCI command at the other site, just as though you were physically logged on there.

You can also use a feature called network I/O while logged on to your local system. To access a network I/O resource, you include the site name as the first part of the I/O resource name. For example, to show a file named .S$NEWS at the site named DALLAS, you would issue the following Show File (SF) command:

```
[] SF

SHOW FILE
   FILE PATHNAME: DALLAS:.S$NEWS
```

Most commands that use I/O pathnames can use network pathnames. The exceptions are as follows:

- No task execution can be done from a remote program file (includes most of the commands in the /PREXEC menu).
- No physical volume commands can be used (includes all commands in the /VOLUME menu).
- The directory utilities CD, BDD, and VC cannot be used with network site names.
- No system configuration utility commands can be used (/SYSCON).
- No absolute disk or CRU commands can be used (MAD, SAD, MADU, SADU, RCRU, WCRU).
- No resource but I/O resources can be preceded by a site name. That is, site names cannot be appended to task run IDs or job names or other items that are not I/O resources.

As an aid to tell you which commands can specify site names, each of the command prompts that allows a network pathname indicates this in the prompt description by [site:] or by [site] if only the site name is required.

AA (ADD ALIAS TO PATHNAME)

4.4 AA (ADD ALIAS TO PATHNAME)

Prompts:

```
ADD ALIAS TO PATHNAME
                    PATHNAME:  [site:]pathname@          (*)
              ALIAS PATHNAME:  [site:]pathname@
```

Purpose:

The AA command assigns aliases (alternate names) to files or directories.

Prompt Details:

PATHNAME:
The pathname of the file or directory to which the alias is to be assigned.

ALIAS PATHNAME:
The same pathname as for the PATHNAME prompt, except the last component which is the alias name by which the file or directory can be identified.

Example:

The following example assigns the alias FILEB to a file named FILE1:

```
[] AA

ADD ALIAS TO PATHNAME
                PATHNAME: VOL1.PROGDIR.DIRA.FILE1
          ALIAS PATHNAME: VOL1.PROGDIR.DIRA.FILEB
```

Related Commands:

DA (Delete Alias from Pathname)

4.5 AB (ASSIGN BREAKPOINT)

Prompts:

ASSIGN BREAKPOINT

RUN ID:	integer	(*)
ADDRESS(ES):	full exp list	

Purpose:

The AB command replaces the specified contents of the address(es) of a task with a breakpoint (an XOP 15,15 instruction), which stops execution of the task at that location. Thus, the task can be suspended at any location in its execution and need not be memory resident to be assigned a breakpoint.

The task is temporarily suspended while the breakpoints are inserted, then restored to its original state. Unless the task is being simulated, you must monitor the task with the Show Internal Registers (SIR) or Show Panel (SP) command to determine when it reaches a breakpoint. When the breakpoint occurs, the task is placed in task state 6 (unconditional suspend). To proceed, the Proceed from Breakpoint (PB), Delete and Proceed from Breakpoint (DPB), or the Delete Breakpoint (DB) and Resume Task (RT) commands must be executed.

The contents of the address(es) are saved and can be restored by the DB command. A maximum of 32 breakpoints per job can be in effect at any one time; an attempt to use more than this number of breakpoints generates an error message. If the runtime ID specifies a system task, your user ID must have a privilege level of two or higher, or the command is aborted.

The DB, DPB, PB, RT, SIR, and ST commands are discussed later in this section.

Prompt Details:

RUN ID:
A valid run ID in your current job. Obtain current run IDs by executing the Show Task Status (STS) command.

ADDRESS(ES):
The integer value(s) of the address(es) within the task where the breakpoint(s) is to occur. Addresses must be separated by a comma.

Example:

In the following example, the AB command assigns breakpoints at locations within a task whose run ID is >A0.

```
[] AB

ASSIGN BREAKPOINT
            RUN ID: >A0
      ADDRESS(ES): >200,>30C,>41A
```

AB (ASSIGN BREAKPOINT)

Assumptions:

The task can be put into task state 6.

Notes:

In VDT mode, breakpoints are displayed as part of the debug panel before SCI is reactivated. The debug panel is not displayed when in TTY mode.

Related Commands:

DB	(Delete Breakpoints)
DPB	(Delete and Proceed from Breakpoint)
LB	(List Breakpoints)
PB	(Proceed from Breakpoint)
SP	(Show Panel)

ABP (ASSIGN BREAKPOINT — PASCAL)

4.6 ABP (ASSIGN BREAKPOINT — PASCAL)

The ABP command assigns a breakpoint at entry to, return from, or both entry to and return from a Pascal routine. Refer to the *DNOS TI Pascal Programmer's Guide* for further information on the ABP command and the following prompts:

```
ASSIGN BREAKPOINT - PASCAL
                        RUN ID:  integer exp                (*)
                 ROUTINE NAME:  character(s)
WHERE (ENTRY/RETURN/BOTH):  {ENTRY/RETURN/BOTH}
```

AF (APPEND FILE)

4.7 AF (APPEND FILE)

Prompts:

```
APPEND FILE
        INPUT ACCESS NAME(S): [site:]pathname@...[site:]pathname @
           OUTPUT PATHNAME: [site:]pathname@
  MAXIMUM RECORD LENGTH: [integer]
```

Purpose:

The AF command appends the contents of one file to another file or appends input from a sequential input device to a file. You can append sequential disk files, relative record disk files, or input from sequential devices to a sequential disk file or a relative record disk file. Input appended to a file is added immediately after the last record of the file, writing over the last end-of-file mark in the file. A new end-of-file mark is added after the appended information. You cannot append directories, program files, image files, or key indexed files using the AF command.

Prompt Details:

INPUT ACCESS NAME(S):
One or more pathnames or device names (separated by commas) that identify the source of information to be appended to the output file or to be copied to a magnetic tape unit. The number of files that can be appended is limited by the number of pathnames, device names, or synonyms that can fit in the 50-character space provided.

If you enter ME or your station ID, AF expects you to enter new data from your terminal. Enter data from your terminal and press the Return key to append the data to the output file. Press the Enter key to terminate input.

OUTPUT PATHNAME:
A pathname that identifies the file to which the specified input file or files, and/or the information input from a sequential device should be appended. If the specified file does not exist, it is automatically created.

MAXIMUM RECORD LENGTH:
The maximum length of the records to be copied. The default value is 512 bytes.

Example:

The following example shows the contents of four files, the appending of three of those files onto the first file, and the listing of the new contents of the first file.

```
[] SF

SHOW FILE
        FILE PATHNAME: .TEST
```

AF (APPEND FILE)

```
RECORD NUMBER ONE
RECORD NUMBER TWO
RECORD NUMBER THREE

[] SF

SHOW FILE
        FILE PATHNAME: V1.FOUR

RECORD NUMBER FOUR

[] SF

SHOW FILE
        FILE PATHNAME: V1.FIVE

RECORD NUMBER FIVE

[] SF

SHOW FILE
        FILE PATHNAME: V1.SIX

RECORD NUMBER SIX

[] AF

APPEND FILE
     INPUT ACCESS NAME(S): V1.FOUR,V1.FIVE,V1.SIX
          OUTPUT PATHNAME: .TEST
    MAXIMUM RECORD LENGTH: 80

[] SF

SHOW FILE
            FILE PATHNAME: .TEST

RECORD NUMBER ONE
RECORD NUMBER TWO
RECORD NUMBER THREE
RECORD NUMBER FOUR
RECORD NUMBER FIVE
RECORD NUMBER SIX
```

AGL (ASSIGN GLOBAL LUNO)

4.8 AGL (ASSIGN GLOBAL LUNO)

Prompts:

```
ASSIGN GLOBAL LUNO
                    LUNO:  [integer]
             ACCESS NAME:  pathname@
          PROGRAM FILE?:  YES/NO          (NO)
```

Purpose:

The AGL command assigns a global logical unit number (LUNO) to a device or file. Since the LUNO is global, it is accessible by more than one job. If the LUNO is currently assigned, an error is returned. The system selects an unused global LUNO if a null response is entered for the LUNO prompt. In either case, the following message is displayed at the station where the AGL command is executed:

```
ASSIGNED LUNO: >XX
```

where:

```
>XX is the LUNO that was assigned.
```

Prompt Details:

LUNO:
A LUNO in the range from 1 through >FE. If you do not enter a response, the system assigns an unused global LUNO by default.

ACCESS NAME:
A valid device name of a physical device or the pathname of a file to which the LUNO is to be assigned.

PROGRAM FILE?:
Enter YES if the LUNO is to be assigned to a program file, image file, or directory file; otherwise, enter NO.

Example:

The following example assigns global LUNO >30 to a line printer (LP04):

```
[] AGL

ASSIGN GLOBAL LUNO
                  LUNO: >30
           ACCESS NAME: LP04
        PROGRAM FILE?: NO

ASSIGNED LUNO: >30
```

AGL (ASSIGN GLOBAL LUNO)

Notes:

The hidden prompt, DISPLAY, is not displayed in interactive mode but can be specified in interactive expert mode or batch mode. The DISPLAY prompt can be used to control whether or not the assigned LUNO is displayed on your terminal. If you respond NO to this prompt, the assigned LUNO is not displayed. The default is YES.

Related Commands:

RGL (Release Global LUNO)

AL (ASSIGN LUNO)

4.9 AL (ASSIGN LUNO)

Prompts:

ASSIGN LUNO

LUNO:	[integer]	
ACCESS NAME:	[site:]pathname@	
PROGRAM FILE?:	YES/NO	(NO)

Purpose:

The AL command assigns a job-local logical unit number (LUNO) to a device or file accessible to the job for I/O operations. If the LUNO is currently assigned, an error is returned. The system selects an unused job-local LUNO if a null response is entered for the LUNO prompt. In either case, the following message is displayed at the station where the AL command is executed:

ASSIGNED LUNO: >XX

where:

>XX is the LUNO that was assigned.

Prompt Details:

LUNO:
A LUNO in the range from 1 through >FE. If you do not enter a response, the system assigns an unused job-local LUNO.

ACCESS NAME:
A valid device name of a physical device or the pathname of a file to which the LUNO is to be assigned.

PROGRAM FILE?:
Enter YES if the LUNO is to be assigned to a program file, image file, or directory file; otherwise, enter NO.

AL (ASSIGN LUNO)

Example:

The following example assigns job-local LUNO > 30 to a line printer (LP04):

```
[] AL

ASSIGN LUNO
                  LUNO: >30
          ACCESS NAME: LP04
        PROGRAM FILE?: NO

ASSIGNED LUNO: >30
```

Notes:

The hidden prompt, DISPLAY, is not displayed in interactive mode but can be specified in interactive expert mode or batch mode. The DISPLAY prompt can be used to control whether or not the assigned LUNO is displayed on your terminal. If you respond NO to this prompt, the assigned LUNO is not displayed. The default is YES.

Related Commands:

RAL	(Release All Logical Units)
RL	(Release LUNO)

ALGS (ASSEMBLE AND LINK GENERATED SYSTEM)

4.10 ALGS (ASSEMBLE AND LINK GENERATED SYSTEM)

Prompts:

```
ASSEMBLE AND LINK GENERATED SYSTEM
          DATA DISK/VOLUME:  [site:]{devicename/volumename}@      (*)
        TARGET DISK/VOLUME:  [site:]{devicename/volumename}@      (*)
           SYSTEM NAME:      alphanumeric                         (*)
```

Purpose:

The ALGS command assembles and links a newly generated system. You must execute this command after an Execute System Generation Utility (XSGU) session and before an initial program load (IPL) is performed.

Prompt Details:

DATA DISK/VOLUME:
The device name or name of the volume that contains the system's linkable object.

TARGET DISK/VOLUME:
The disk device or volume name that is to contain the newly generated system.

SYSTEM NAME:
A one- to eight-character alphanumeric string that is the name of the system to be generated.

Example:

In the following example, a new system has been generated and is contained in disk drive DS01. The name of the system being assembled and linked is S$SYS1. The ALGS command assembles and links the newly defined system with the following prompt responses:

```
[] ALGS

ASSEMBLE AND LINK GENERATED SYSTEM
        DATA DISK/VOLUME: DS01
      TARGET DISK/VOLUME: DS01
             SYSTEM NAME: S$SYS1
```

The ALGS command execution time is approximately 30 minutes. If the ALGS command terminates with an error, a message is displayed on the terminal indicating where the error occurred in the ALGS batch stream.

After you press the Return key, the following message is also displayed:

```
BATCH SCI HAS COMPLETED
```

ALGS (ASSEMBLE AND LINK GENERATED SYSTEM)

Assumptions:

The specified target disk is the same as that specified when the system was generated. The specified system name is the same as the output configuration name specified when the system was generated.

Messages:

If the ALGS command did not terminate normally, see the last line of the batch SCI listing file to determine which error occurred. The pathname of the listing file is:

targetdisk.SSGU.systemname.ALGSLIST

where:

targetdisk and systemname are those responses entered for the ALGS prompts.

If too much system table area is requested, the following message appears in the system link map named SYSMAP:

OVERLAY TOOBIG OVERWRITES NEXT HIGHER PHASE

In this case, you must generate a new system specifying less table area.

Related Commands:

IGS	(Install Generated System)
PGS	(Patch Generated System)
TGS	(Test Generated System)

ALN (ASSIGN LOGICAL NAME)

4.11 ALN (ASSIGN LOGICAL NAME)

Prompts:

```
ASSIGN LOGICAL NAME
            LOGICAL NAME: alphanumeric
          RESOURCE TYPE: [SPOOL/NONE/TEMP/PERM]
```

If you enter NONE or a null value in response to RESOURCE TYPE, the following prompts are displayed:

```
ASSIGN LOGICAL NAME
          ACCESS NAME(S): {[site:]pathname/pathname...pathname}@
           GLOBAL NAME?: YES/NO                                  (NO)
```

If you enter SPOOL in response to RESOURCE TYPE, the following prompts are displayed:

```
ASSIGN SPOOLER PARAMETERS
              ANSI FORMAT?: YES/NO                               (NO)
     LISTING DEVICE OR CLASS: [site:]devicename
     NUMBER OF LINES/PAGE: integer                              (62)
                     FORM: character(s)                         (STANDARD)
          NUMBER OF COPIES: integer                             (1)
            BANNER SHEET?: YES/NO                               (NO)
             GLOBAL NAME?: YES/NO                               (NO)
```

If you enter TEMP in response to RESOURCE TYPE, the following prompts are displayed:

```
TEMPORARY FILE PARAMETERS
        DISK OR VOLUME NAME: {devicename/volumename}            (*)
                  FILE TYPE: {SEQUENTIAL/RELATIVE RECORD}       (SEQUENTIAL)
                CREATE NOW?: YES/NO                             (NO)
          JOB ACCESS LEVEL: [EA/EW/SH/RO]
```

If you enter PERM in response for RESOURCE TYPE, the following prompts are displayed:

```
PERMANENT FILE PARAMETERS
              FILE PATHNAME: [site:]pathname
                  FILE TYPE: {SEQ/REL/KEY/DIR}                  (SEQUENTIAL)
                CREATE NOW?: YES/NO                             (NO)
          JOB ACCESS LEVEL: [EA/EW/SH/RO]
```

ALN (ASSIGN LOGICAL NAME)

If you specify SEQ for FILE TYPE, the following prompts are displayed:

```
SEQUENTIAL FILE PARAMETERS
      LOGICAL RECORD LENGTH:  [integer]
   PHYSICAL RECORD LENGTH:  [integer]
         INITIAL ALLOCATION:  [integer]
    SECONDARY ALLOCATION:  [integer]
             EXPANDABLE?:  YES/NO                    (YES)
          BLANK SUPPRESS?:  YES/NO                   (NO)
            FORCED WRITE?:  YES/NO                   (NO)
```

If you specify REL for FILE TYPE, the following prompts are displayed:

```
RELATIVE RECORD FILE PARAMETERS
      LOGICAL RECORD LENGTH:  [integer]
   PHYSICAL RECORD LENGTH:  [integer]
         INITIAL ALLOCATION:  [integer]
    SECONDARY ALLOCATION:  [integer]
             EXPANDABLE?:  YES/NO                    (YES)
            FORCED WRITE?:  YES/NO                   (NO)
```

If you specify KEY for FILE TYPE and PERM for RESOURCE TYPE, the following prompts are displayed:

```
KEY INDEXED FILE PARAMETERS
      LOGICAL RECORD LENGTH:  integer
   PHYSICAL RECORD LENGTH:  [integer]
         INITIAL ALLOCATION:  [integer]
    SECONDARY ALLOCATION:  [integer]
            MAXIMUM SIZE:  integer
```

After you respond to these prompts, you receive additional prompts for defining each key. Refer to the description of the Create Key Indexed File (CFKEY) command for details on these prompts.

If you specify DIR for FILE TYPE and PERM for RESOURCE TYPE, the following prompts are displayed:

```
  DIRECTORY FILE PARAMETERS
                MAX ENTRIES:  integer
DEFAULT PHYSICAL RECORD SIZE:  [integer]
```

Purpose:

The ALN command associates a logical name with an I/O device, file, or files, and optionally associates parameters with the logical name. Your response to the RESOURCE TYPE prompt determines what type of logical name and parameters are created.

ALN (ASSIGN LOGICAL NAME)

Specify SPOOL in response to the RESOURCE TYPE prompt to associate a logical name with a spooler device and spooler parameters. Spooler logical names are used as responses to SCI prompts and within programs to provide easy access to spooler devices. After you assign a logical name to a spooler device you can write output to the logical name as though it were the actual device. All of your output will be collected and written to a temporary file created for this purpose. When you are finished writing, the temporary file will be queued to the specified device to wait its turn to print, then deleted after all the output has been written to the device. Spooler parameters are created from your responses to the prompts that appear after the RESOURCE TYPE prompt and control how the output is printed on the actual device.

Specify TEMP in response to the RESOURCE TYPE prompt to associate a logical name with a job local temporary file and temporary file parameters. You can create sequential and relative record temporary files. You can also specify whether the file is to be created when the ALN command is executed or when the logical name is first accessed. In either case, the file will exist until a release logical name operation is performed and all LUNOs are released, or the job terminates. If the system crashes, all temporary files are deleted by the next system IPL. Therefore, temporary files should be used for work files only and not for files that you want to preserve.

Specify PERM to associate a logical name with a permanent file and permanent file parameters. Using PERM, you can create sequential, relative record, key indexed, and directory files. You can also specify whether the file is to be created when the ALN command is executed or when the logical name is first accessed. In either case, permanent files are not deleted when the jobs that created them terminate.

Specify NONE to associate a logical name with one or more I/O resources. If only one pathname is supplied for the ACCESS NAME(S) prompt, the logical name can be used much like a synonym is used. By specifying a series of pathnames, you can create concatenated files and multifile sets. These types of files allow you to access a series of files as though they were one. Since individual components of concatenated files and multifile sets can be on separate disks, this option gives you a way to create files that are too large for one disk to store.

A concatenated file is created by supplying a series of all sequential or all relative record files. The files must be either all blocked or all unblocked, and for concatenated relative record files, each file must have the same logical record size. A multifile set is created by supplying a series of key indexed files that have the same physical record size and the same key definitions. Initially, all but the first of these files must be empty. In subsequent definitions of a multifile set, the same files as in the first definition must be associated in the same order.

Through the ALN command, logical names can be used in a command procedure or user program, with DNOS resolving the logical name to the associated resource.

ALN (ASSIGN LOGICAL NAME)

Prompt Details:

LOGICAL NAME:
A one- to eight-character name by which the resource is to be known. Do not use a name which is also a device name on your system. Using such a name can cause unpredictable results.

RESOURCE TYPE:
The characters specifying the type of parameters to be assigned to the logical name. SPOOL specifies spooler parameters, NONE or a null response specifies there are no parameters to be assigned, TEMP specifies temporary file parameters, and PERM specifies permanent file parameters.

ACCESS NAME(S):
The pathname(s) of the resource(s) to associate with the specified logical name. If more than one pathname is specified, each pathname must be unique. Restrictions on concatenating files are discussed in the *DNOS Systems Programmer's Guide*.

GLOBAL NAME?:
A YES response specifies that the logical name assigned is available to all system users. A NO response assigns the logical name locally, making it unavailable to other users. The Snapshot Global Name Definitions (SGND) command assigns current global names permanently.

ANSI FORMAT?:
A YES response specifies that ANSI control characters are to be recognized. A NO response specifies that ANSI control characters are to be ignored.

LISTING DEVICE OR CLASS:
A specific output device or class of devices to which the data is written. The device or class must be previously identified to the spooler via the Modify Spooler Device (MSD) command.

Example Class of Devices:

FLP — fast line printer

Example Specific Output Devices:

CSxx — cassette
LPxx — line printer
MTxx — magnetic tape
STxx — station

where xx = unit number

If you specify a class of devices, the first available device in that class is used.

ALN (ASSIGN LOGICAL NAME)

NUMBER OF LINES/PAGE:
The number of lines per page.

FORM:
A one- to eight-character string which specifies the form type on which the output is to be placed.

NUMBER OF COPIES:
The number of copies to be generated.

BANNER SHEET?:
A YES response specifies that a banner sheet be output prior to the actual file. NO specifies that no banner sheet is to be output. The banner sheet includes the job name, user ID, and the date and time the file is printed.

DISK OR VOLUME NAME:
The device or volume name that will contain the temporary file.

FILE PATHNAME:
The name of the file to be associated with the logical name.

FILE TYPE:
One of the following file type abbreviations can be entered:

SEQ — Sequential File
REL — Relative Record File
KEY — Key Indexed File
DIR — Directory

If you specify TEMP for RESOURCE TYPE, only SEQ and REL can be specified for FILE TYPE.

CREATE NOW?:
A YES response specifies that the file does not exist and is to be created during ALN processing. A NO response specifies that the file already exists or must be created when the usage is first specified.

JOB ACCESS LEVEL:
The type of access required at the job level. The following are valid responses:

EA — exclusive all
EW — exclusive write
SH — shared
RO — read only

ALN (ASSIGN LOGICAL NAME)

LOGICAL RECORD LENGTH:

An even number of bytes that specifies the average logical record size. If records are to be blank-suppressed, blank strings must not be included in the size estimate.

For key indexed files, this prompt determines the length to which records are expanded and the logical record length returned on a Read File Characteristics suboperation of the I/O SVC. It is not used to compute storage requirements.

Refer to the *DNOS Supervisor Call (SVC) Reference Manual* for information on the Read File Characteristics suboperation.

PHYSICAL RECORD LENGTH:

An even number of bytes that specifies the physical record size. The default record size is specified in response to the DEFAULT RECORD SIZE prompt during system generation.

For key indexed files, the physical record size should be at least three times the logical record size.

INITIAL ALLOCATION:

An integer that specifies the size, in logical records, of the initial disk space allocated to the file.

SECONDARY ALLOCATION:

An integer that specifies the size, in logical records, of subsequent disk space allocated if the file grows larger than the initial size.

EXPANDABLE?:

A YES response makes the file expandable beyond initial allocation.

BLANK SUPPRESS?:

A YES response specifies that the file is to be blank suppressed; however, a blank-suppressed file cannot support rewrite operations.

FORCED WRITE?:

Disk writes to sequential files are ordinarily deferred until memory is needed. A YES response specifies that logical records be written immediately.

MAX ENTRIES:

An integer number specifying the maximum number of file names to be kept in this directory. The number entered is rounded up to the nearest prime number. The maximum number of entries is 512.

DEFAULT PHYSICAL RECORD SIZE:

An even number of bytes to be used as the default record size for files in this directory.

ALN (ASSIGN LOGICAL NAME)

Example:

In the following example, the logical name MSG is assigned to the printer device, LP01. After execution of the ALN command, any output to logical name MSG is printed on LP01.

```
[]ALN

ASSIGN LOGICAL NAME
             LOGICAL NAME: MSG
            RESOURCE TYPE: SPOOL

ASSIGN SPOOLER PARAMETERS
              ANSI FORMAT?: NO
 LISTING DEVICE OR CLASS: LP01
    NUMBER OF LINES/PAGE: 50
                    FORM:
         NUMBER OF COPIES: 1
            BANNER SHEET?: NO
             GLOBAL NAME?: NO
```

Notes:

The logical name associated with the file name should not be a legal device name if that device is also in use by the job.

Related Commands:

LLN	(List Logical Names)
RLN	(Release Logical Name)
SGND	(Snapshot Global Name Definitions)

4.12 ANS (ANSWER INCOMING CALL)

Prompts:

```
ANSWER INCOMING CALL
      TERMINAL ACCESS NAME:  stationname      (*)
                ANSWERBACK:  [character(s)]   (*)
```

Purpose:

The ANS command monitors a communication port for completion of a connection with a remote terminal when the call is initiated by the remote terminal. The ANS command alerts you that a specific incoming call has been received. Whether or not you issue the ANS command, the system continuously monitors for a ring signal on those ports which are available.

Prompt Details:

TERMINAL ACCESS NAME:

A valid system terminal name. The terminal must be logically turned on and not currently in use.

ANSWERBACK:

The ANSWERBACK prompt is a well-defined code used to identify a particular teleprinter device (TPD). This answerback code must be previously defined at the remote TPD. If an answerback is requested, upon completion of the connection, the system delays for a specified time to allow the terminal to auto-answerback. A null response specifies that the terminal answerback is unknown or undefined.

Example:

In the following example, the port established as ST21 during system generation is waiting to answer an incoming call from a TPD that has the answerback code BMT000.

```
[]ANS

ANSWER INCOMING CALL
   TERMINAL ACCESS NAME: ST21
             ANSWERBACK: BMT000
```

ANS (ANSWER INCOMING CALL)

Messages:

A completion message appears when a connection is established with a remote TPD. This message indicates whether answerback data is present and whether data is expected.

The received answerback consists of ASCII characters ranging from >20 through >7E. Any nongraphic characters are deleted from the answerback upon reception. The received answerback is compared with the user-supplied answerback and is saved as the value of the synonym ABM. If no answerback is received, this synonym is deleted.

Notes:

If the terminal answerback contains lowercase data, the user-supplied ANSWERBACK must contain the same lowercase data for a successful comparison.

During normal mode usage, no answerback is requested. If the terminal sends an auto-answerback, the answerback is stored up to the buffering capacity of the port's buffer (as defined during system generation).

Using the ANS command, you can request answerback if it is not automatically supplied. The answerback is optionally compared to a user-supplied answerback; if a match is not obtained, the system disconnects. If a successful connection is made, a message is displayed.

The user-supplied answerback is retained as a synonym and used as a default display on subsequent ANS commands.

Related Commands:

CALL	(Call Terminal)
DISC	(Terminal Disconnection)
LHPC	(List Hardcopy Terminal Port Characteristics)
MHPC	(Modify Hardcopy Terminal Port Characteristics)

4.13 AS (ASSIGN SYNONYM VALUE)

Prompts:

```
ASSIGN SYNONYM VALUE
                SYNONYM:  alphanumeric
                  VALUE:  [character(s)]
```

Purpose:

The AS command defines a string of one or more characters to substitute for another string of characters. You can use the substitute string as responses to prompts instead of the original string. The defined string is called a synonym, and the string of characters for which it substitutes is called the synonym value. For example, you can legally define the following synonyms:

Synonym	Synonym Value
PRINTER	LP01
PAYFILE	VOL1.DIR2.FILE13

Synonyms are usually shorter than their synonym value, although they can be longer. A common use of synonyms is to represent pathnames. A synonym can represent an entire pathname or only the first part of a pathname. For example, if the character A is a synonym with a value of V1.DIR.SUB1 and MYFILE is a synonym with a value of V1.DIR.SUB1.FIL1, the following character strings are equivalent responses to command prompts:

```
MYFILE
A.FIL1
V1.DIR1.SUB1.FIL1
```

When you create synonyms, they are automatically associated with the user ID under which you logged on. To use a synonym, you must be logged on under the user ID of the user who created the synonym. When you log off, the current set of synonyms is saved for the next log-on.

Prompt Details:

SYNONYM:
One or more alphanumeric characters representing the synonym to be assigned. The first character must be alphabetic.

VALUE:
The character string to which the synonym is being assigned. If you do not enter a value for the synonym, the current value assigned to the synonym is erased and the synonym is deleted.

AS (ASSIGN SYNONYM VALUE)

Example:

The following example assigns the synonym VJ to the pathname VOL1.PROGDIR.DIRB.FILE1:

```
[] AS

ASSIGN SYNONYM VALUE
          SYNONYM: VJ
            VALUE: VOL1.PROGDIR.DIRB.FILE1
```

Notes:

You should not define any synonyms beginning with a dollar sign ($) since synonyms reserved for system use begin with the $ character.

Related Commands:

LS	(List Synonyms)
MS	(Modify Synonym)

4.14 ASB (ASSIGN SIMULATED BREAKPOINT)

Prompts:

```
ASSIGN SIMULATED BREAKPOINT
          ON (A,C,P,R,S): {A/C/P/R/S}      (PC)
                   FROM:  full exp
                   THRU:  [full exp]
                  COUNT:  full exp          (1)
                DISPLAY:  [full exp]
```

Purpose:

The ASB command sets up a breakpoint on a range of address values for memory on one of the following conditions:

- The memory at the address is altered (A)
- A CRU access is made to the address (C)
- The PC value is the specified address (P)
- A memory reference is made to the address (R)
- The status value is set in the status register (S)

The breakpoints set with this command are valid only while task simulation is in progress. In this case, breakpoints are conditions which stop execution but allow execution to be resumed by an operator command, either by using the Resume Simulated Task (RST) command or by pressing the F3 function key. Each simulated breakpoint is assigned a number which is displayed at the completion of the ASB command. When a breakpoint occurs during simulation, a panel and the breakpoint number are displayed along with the display string.

Prompt Details:

ON (A,C,P,R,S):
The characters A, C, P, R, S are valid responses to this prompt and have the following meanings:

A = Memory alteration
C = CRU access
P = Program Counter (PC) value
R = Reference (memory)
S = Status Register (ST) value

FROM:
The integer expression that specifies the lower address limit for the breakpoint.

THRU:
The integer expression that specifies the upper address limit for the breakpoint. The default value is the value specified for the FROM prompt.

ASB (ASSIGN SIMULATED BREAKPOINT)

COUNT:
The integer expression that specifies the number of times this breakpoint is to be encountered before execution is halted. The initial value is one.

DISPLAY:
The integer expression that specifies the memory address to be displayed when this breakpoint is reached. The default value is the program counter (PC) value at the time the breakpoint is reached.

Example:

The following example causes task simulation to halt when data is stored in memory locations >6 through >1C0. This breakpoint is to be encountered one time, and, if it is encountered, the value in the PC is displayed.

```
[] ASB

ASSIGN SIMULATED BREAKPOINT
       ON (A,C,P,R,S): A
                 FROM: 6
                 THRU: >1C0
                COUNT:
              DISPLAY:
```

Special Cases:

A memory write operation that does not change the value in memory is not a memory alteration. The debugger does not detect breakpoints set on memory alteration which occurs through the execution of an Extended Operation (XOP).

Assumptions:

The program with breakpoints to be simulated has been placed in a controlled mode by the previous execution of the Execute in Debug Mode (XD) command.

Notes:

If the response to the DISPLAY prompt is specified to be the contents of a register (for Example, #R3), the current address in the register is used for the display address.

Related Commands:

DSB	(Delete Simulated Breakpoints)
LSB	(List Simulated Breakpoints)
QD	(Quit Debug Mode)
ST	(Simulate Task)
XD	(Execute in Debug Mode)

4.15 AT (ACTIVATE TASK)

Prompts:

```
ACTIVATE TASK
                RUN ID:   integer                (*)
```

Purpose:

The AT command activates an unconditionally suspended task.

Prompt Details:

RUN ID:
A valid task run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

Example:

In the following example, the AT command activates a task with a run ID of > A.

```
[] AT

ACTIVATE TASK
             RUN ID:   >A
```

Assumptions:

The task specified to be activated has or will place itself in a suspended state by using the Unconditional Wait SVC. If the task has already been suspended, it is then reactivated. If the task has not yet been suspended, it is reactivated immediately when the Unconditional Wait SVC is issued. If the task was bid via the Execute and Halt Task (XHT) command, a subsequent AT command has no effect. The Activate Task SVC processor has no effect if the task is debugged with the XHT command.

Related Commands:

HT	(Halt Task)
RT	(Resume Task)
STS	(Show Task Status)

AUI (ASSIGN USER ID)

4.16 AUI (ASSIGN USER ID)

Prompts:

```
ASSIGN USER ID
                USER DESCRIPTION:   character(s)
                     NEW USER ID:   alphanumeric
                    NEW PASSCODE:   [character(s)]
     USER PRIVILEGE CODE (0..7):    integer                 (0)
```

Purpose:

The AUI command allows authorized users to assign user IDs, passcodes, user privilege codes, and user descriptions.

Prompt Details:

USER DESCRIPTION:
A string of 1 to 20 characters. This is for information purposes only to identify the user.

NEW USER ID:
A string of one to eight alphanumeric characters. The user ID rules are the same as those for a pathname component, excluding the period.

NEW PASSCODE:
A string of one to eight characters that comprise a passcode to be associated with the user ID assigned in response to the NEW USER ID prompt. If no passcode is given, the default is eight blanks. Special characters (%, $, ^, and so on) are legal passcode components.

USER PRIVILEGE CODE:
A number in the range of zero through seven that specifies the privilege level at which the user can operate the system. Seven denotes the highest privilege level.

Example:

The following example assigns a user description, a user ID, a passcode, and a user privilege code:

```
[] AUI

ASSIGN USER ID
            USER DESCRIPTION: LANGUAGE TESTERS
                 NEW USER ID: TESTER
                NEW PASSCODE: LANG
 USER PRIVILEGE CODE (0..7): 0
```

AUI (ASSIGN USER ID)

Assumptions:

The user ID being added does not currently exist in the system.

Notes:

When AUI is used in a file security environment, you must have read and write access rights to the .S$CLF file.

Related Commands:

DUI	(Delete User ID)
LUI	(List User IDs)
MPC	(Modify Passcode)
MUI	(Modify User ID)

BATCH (BEGIN BATCH EXECUTION)

4.17 BATCH (BEGIN BATCH EXECUTION)

Purpose:

The BATCH command is used in a batch stream to remove unwanted synonyms and their values. It is recommended that this command be the first command in any batch stream and the End Batch Execution (EBATCH) command be the last command in any batch stream. A Show File (SF) command on the files .S$CMDS.Q$SYN and .S$CMDS.BATCH shows the synonyms that are deleted. The BATCH command cannot be executed in the interactive mode; it should only be specified in a batch stream used by the Execute Batch (XB) or Execute Batch Job (XBJ) command.

A number of options exist to format the batch listing. A SF command on the file .S$CMDS.BATCH shows six options. The most commonly used option is LS (List Synonyms). If YES is specified for this option, a list of all synonyms is placed on the batch listing file or device.

Example:

In the following example, the BATCH command removes unwanted synonyms. Synonyms are listed on the batch listing file. The EBATCH command specifies that there are no more commands to be processed.

```
BATCH  LS=YES
   *
   *
   *
EBATCH
```

Notes:

The BATCH command is not intended for interactive use. The following prompts are available in batch stream format when using the BATCH command:

```
BEGIN BATCH EXECUTION
                    USER ID:  [alphanumeric]          (*)
                 STATION ID:  [stationname]@          (*)
  BATCH INPUT ACCESS NAME:    [site:][filename]@      (*)
BATCH LISTING ACCESS NAME:    [site:][pathname]@      (*)
        LS (LIST SYNONYMS)?:  [YES/NO]                (NO)
        LIST LOGICAL NAMES?:  [YES/NO]                (NO)
```

The USER ID prompt indicates the user ID with which the batch stream is associated. The response to STATION ID is the station from which the batch stream is to be executed. The BATCH INPUT ACCESS NAME is the name of the batch input file. BATCH LISTING ACCESS NAME is the destination file for the listing. The LS(LIST SYNONYMS)? prompt gives you the option to list the synonyms in the listing file. The LIST LOGICAL NAMES? prompt gives you the option to list the logical names in the listing file.

BATCH (BEGIN BATCH EXECUTION)

Related Commands:

EBATCH (End Batch Execution)
XB (Execute Batch)
XBJ (Execute Batch SCI Job)

BD (BACKUP DIRECTORY)

4.18 BD (BACKUP DIRECTORY)

Prompts:

```
BACKUP DIRECTORY
        DIRECTORY PATHNAME:  [pathname]@                                    (*)
    SEQUENTIAL ACCESS NAME:  [[site:]pathname/[site:]device name... [site:]device
                             name]@                                         (*)

       CONTROL ACCESS NAME:  [site:][pathname]@
        LISTING ACCESS NAME:  [site:][pathname]@
                    OPTIONS:  [character(s)...character(s)]
     EXECUTION MODE(F,B):     {F/FOREGROUND/B/BACKGROUND}(FOREGROUND)
```

Depending on your responses, you might receive one or more additional prompts:

```
PATHNAME OF BACKUP FILE
                PATHNAME:  pathname@

BLOCKING FACTOR IN BYTES
          BLOCKING FACTOR:  [integer]                                       (9600)

DATE OF OLDEST FILE
                    YEAR:  integer
                   MONTH:  integer
                     DAY:  integer
                    HOUR:  [integer]                                        (0)
                  MINUTE:  [integer]                                        (0)
```

Purpose:

The BD command copies a directory into a sequential format. You can keep the copy on a disk file or on magnetic tape, and the backup file can span more than one disk or tape. Later, you can rebuild the directory by using the Restore Directory (RD) command. To make sure the directory and the backup file match, you can use the Verify Backup (VB) command. If you want to copy files from a directory to another directory, you can use the Copy Directory (CD) and Verify Copy (VC) commands. Alternatively, you can use the Backup Directory to Device (BDD) command to copy a directory onto a sequential medium. BDD is generally faster than BD, but it does not provide as many options.

The BD command provides numerous options that let you select the files you want to include in the copy. Unless you specify otherwise, BD copies all the files (and aliases) in the directory, including any subdirectories and the files they contain. BD cannot copy the system files .S$ROLLD.S$ROLLA, .S$DIAG, and .S$CRASH, or the spooler directory .S$SDTQUE, when backing up .VCATALOG unless the system files are specifically included in the backup using the INCLUDE directive within a control file.

BD (BACKUP DIRECTORY)

BD allows you to use a *control file* containing directives that identify the directory, backup file, and files involved in the operation. For information on the control file directives, see the notes for BD.

Do not have other file activity going to the directory while it is being backed up. Activities such as creating files can cause inconsistent file structures to be created if a backup is in progress.

Prompt Details:

DIRECTORY PATHNAME:
The pathname of the directory you want backed up. Unless you limit the command with options or control file directives, the backup file includes all files (and aliases) in this directory, including any subdirectories and files they contain. You can skip this prompt only if you specify the directory pathname with a MOVE directive in a control file.

SEQUENTIAL ACCESS NAME:
The access name for the backup file. You can enter a sequential file pathname, a device name, or a list of device names. If you enter the pathname of a sequential file, BD puts the backup copy there, automatically creating the file if necessary. If you enter the device name of a tape drive, BD puts the backup file on the tape mounted there. If you list more than one tape drive, BD fills them in the order given. If you enter the device name of a disk drive (or a list of device names DS01 through DS99), BD displays an additional prompt for the pathname of the backup file. For details on multiple volume backups, refer to the special cases for BD. You can skip this prompt only if you specify the sequential access name with a MOVE directive in a control file.

CONTROL ACCESS NAME:
The access name of a control file. The control file contains directives that specify which files in the directory are to participate in the operation. If you enter a pathname of a sequential file, BD uses the directives it finds there. If you enter ME, BD expects you to enter control file directives from your station. For details on control files, refer to the notes for BD. If you respond to both the DIRECTORY PATHNAME and SEQUENTIAL ACCESS NAME prompts, you do not need to specify a control file.

LISTING ACCESS NAME:
The device name or the file name where you want to receive listings from the backup operation. If you skip this prompt, BD assumes by default that you want the listings sent to the terminal local file of your station. If the listing file reports an error in copying a file, the file is not included in the backup copy and cannot be restored later.

BD (BACKUP DIRECTORY)

OPTIONS:

List of options to control the backup operation. Each option is expressed as a pair of alternatives, for example ALIAS and NOALIAS. You can choose one or the other, but not both. If you do not choose either, BD uses a default. Options specified in the control file override any options chosen here. The options listed after ORIGINAL OPTIONS are the default options modified by the response to the OPTIONS prompt. Options specified by a control file are not included at this point on the listing.

Option	Purpose
ALIAS	(Default) Copy aliases.
NOALIAS	Do not copy aliases.
BLOCK	Block the output into records on the magnetic tape. If you choose this option, BD displays the additional BLOCKING FACTOR prompt. You do not need this option for disk files.
NOBLOCK	(Default) Do not block the output for magnetic tape.
DATE	Copy only files updated later than a specified date. If you choose this option, BD displays additional prompts for the date and time. If the DATE option is used in a control file, additional date and time prompts must be answered. In batch or expert mode they can be answered directly. In interactive mode the DATE option must be included in the response to the OPTIONS prompt in order to answer the additional date and time prompts. If either the last update date or the creation date is after the date specified, the file is copied.
NODATE	(Default) Copy files regardless of their dates.
REWIND	Rewind the tape before performing the backup operation.
NOREWIND	(Default) Do not rewind the tape before performing the backup operation.
UNLOAD	Rewind and unload the reel after the backup operation.
NOUNLOAD	(Default) Do not rewind and unload the reel after the backup operation.

BD (BACKUP DIRECTORY)

Option	Purpose
SYSFILES	(Default) Include the following system files and directories in a backup operation on the system disk: .SCI990 .S$SYSLIB .S$CDT .S$ISBTCH .S$CMDS .S$ISLIST .S$EXPMSG .S$LANG .S$MSG .S$PWCS .S$SGU$ .S$SHARED .S$SYSTEM .S$SHIP .S$IPL .S$UTIL
NOSYSFILE	Do not include the system files in the backup.
FAST	Copy a program file all at once. If you choose this option, you need to make sure the disk that contains the directory and the disk where you later restore the directory have the same sector size.
NOFAST	(Default) Copy each task, procedure, program segment, and overlay image individually.
MULTI	No effect. This option is included for compatibility with existing BD control files.
NOMULTI	No effect. This option is included for compatibility with existing BD control files.
CMP	No effect. This option is included for compatibility with RD control files.
NOCMP	No effect. This option is included for compatibility with RD control files.

EXECUTION MODE(F,B):

A choice of whether to carry out the command in the foreground or background. If you enter F or FOREGROUND, the command executes in the foreground. If you enter B or BACKGROUND, the command executes in the background. Though the default is FOREGROUND, in a batch stream the BD command always executes in the background.

PATHNAME:

The pathname of the backup file, which can span one or more disks. The pathname consists of the volume name given to all of the disks involved in the backup, followed by a period and a single pathname component to identify the backup file. You cannot give the device name of a disk unit as part of the pathname—nor can you give the volume name by itself. This prompt appears only when you give a disk unit device name (or list of disk unit device names) as your response to the SEQUENTIAL ACCESS NAME prompt. The discussion of special cases for BD provides details on multiple volume backups.

BD (BACKUP DIRECTORY)

BLOCKING FACTOR:
The number of bytes per record on magnetic tape. This prompt appears only when you choose the BLOCK option. The default (and initial value) for the blocking factor is 9600 bytes per record. This improves tape utilization by reducing the number of interrecord gaps. The largest blocking factor allowed for this prompt is 9600 bytes per record.

YEAR:
The two- or four-digit number for a year. This prompt appears only when you choose the DATE option. Your response becomes the year portion of the cutoff date used by the DATE option to determine which files to copy.

MONTH:
The one- or two-digit number of a month. This prompt appears only when you choose the DATE option. Your response becomes the month portion of the cutoff date used by the DATE option to determine which files to copy.

DAY:
The one- or two-digit number of a day of the month. This prompt appears only when you choose the DATE option. Your response becomes the day portion of the cutoff date used by the DATE option to determine which files to copy.

HOUR:
The hour on the 24-hour clock. This prompt appears only when you choose the DATE option. Your response becomes the hour portion of the cutoff time used by the DATE option to determine which files to copy. If you skip this prompt, BD assumes 0 (the initial value).

MINUTE:
The minute of the hour. This prompt appears only when you choose the DATE option. Your response becomes the minute portion of the cutoff time used by the DATE option to determine which files to copy. If you skip this prompt, BD assumes 0 (the initial value).

Example:

This example creates and verifies the backup file BACKUP82.OCTOBER for the directory VOL2.DIR1. Listing the DATE option causes the BD command to issue additional prompts for selecting only files updated since a specified date, 10/01/82.

```
[] BD

BACKUP DIRECTORY
        DIRECTORY PATHNAME: VOL2.DIR1
   SEQUENTIAL ACCESS NAME: BACKUP82.OCTOBER
      CONTROL ACCESS NAME:
      LISTING ACCESS NAME: LP
                  OPTIONS: DATE
      EXECUTION MODE(F,B): BACKGROUND
```

```
      DATE OF OLDEST FILE
                     YEAR: 82
                    MONTH: 10
                      DAY: 1
                     HOUR: 0
                   MINUTE: 0
```

The BD command produces the following listing:

```
   BACKUP DIRECTORY        12:40:30 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       VOL2.DIR1
ORIGINAL DESTINATION: BACKUP82.OCTOBER
ORIGINAL OPTIONS:      ALIAS,DATE,NOBLOCK,NOREWIND,NOUNLOAD,SYSFILES,NOFAST
CONTROL FILE:

LIST FILE:              LP

   **       DIRECTORY  VOL2.DIR1
   **   FILE3
   **   FILE2
   **   PROGRAMS - PROGRAM FILE

   **   TASK       ID
   **   PROGA     >03
   **   PROGB     >0E
   **        END OF PROGRAM FILE

   **        END OF DIRECTORY  VOL2.DIR1


ELAPSED TIME = 0 MINUTES   25 SECONDS
SIZE OF INPUT = 1782 ADU'S

****** BACKUP DIRECTORY COMPLETED
```

The following VB command makes sure the backup file matches the original directory. It also uses the DATE option to include only files updated since 10/01/82 in the operation.

```
   [] VB

   VERIFY BACKUP
      SEQUENTIAL ACCESS NAME: BACKUP82.OCTOBER
           DIRECTORY PATHNAME: VOL2.DIR1
         CONTROL ACCESS NAME:
         LISTING ACCESS NAME: LP
                     OPTIONS: DATE
         EXECUTION MODE(F,B): BACKGROUND

   DATE OF OLDEST FILE
                        YEAR: 82
                       MONTH: 10
                         DAY: 1
                        HOUR: 0
                      MINUTE: 0
```

BD (BACKUP DIRECTORY)

The VB command produces the following listings:

```
   VERIFY BACKUP            12:39:53 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       BACKUP82.OCTOBER
ORIGINAL DESTINATION:  VOL2.DIR1
ORIGINAL OPTIONS:      ALIAS,DATE,NOREWIND,NOUNLOAD,SYSFILES,NOFAST
CONTROL FILE:
LIST FILE:             LP
 **        DIRECTORY  VOL2.DIR1
 **   FILE3
 **   FILE2
 **   PROGRAMS - PROGRAM FILE

 **   TASK        ID
 **   PROGA      >03
 **   PROGB      >0E
 **        END OF PROGRAM FILE

 **        END OF DIRECTORY  VOL2.DIR1


ELAPSED TIME = 0 MINUTES   24 SECONDS
SIZE OF INPUT = 1782 ADU'S

****** VERIFY BACKUP COMPLETED
```

Messages:

PROGRAM FILE BACKED UP WITH ERRORS
PRECEDING TASK AND ALL SUCCEEDING TASKS NOT RESTORED
: The hardware encountered a disk error on the source disk. The last task listed for this program file cannot be restored, but the other tasks listed can be restored. Any tasks in this program file not appearing on the listing file cannot be restored.

< filename> BACKED UP WITH ERRORS-CANNOT BE RESTORED
: The hardware encountered a disk error. The file named in the message cannot be restored.

ATTEMPT TO UNLOAD THE SYSTEM DISK
: You specified a backup to the system disk by volume name. The message means that BD could not unload the system disk at the completion of the operation. No error condition results.

Notes:

When BD is used in a file security system, you must have read access rights to the files which are to be backed up, and you must have write access rights to the file specified for SEQUENTIAL ACCESS NAME.

The control files used with BD, CD, RD, VB, and VC consist of directives that limit the actions of the commands. If you use a control file with BD, you should also use control files for subsequent RD and VB commands. If you use a control file for CD, you should use the same control file for a subsequent VC command. Directives, file names, pathnames, and options must all be in upper case in a control file.

You can shorten the names of the directives to the first few characters, if you like. For example, you can enter the MOVE directive as MOVE, MOV, MO, or M.

The control file must have END as its last directive. When you enter control file directives from your station, you can also terminate the control file by pressing the Enter key.

The control file consists of one or more of the following directives:

MOVE directory pathname,sequential access name (BD)
MOVE sequential access name, directory pathname (RD and VB)
MOVE input pathname,output pathname (CD)
MOVE master pathname,copy pathname (VC)

> For the BD, RD, and VB commands, the MOVE directive has the same function as your responses to the DIRECTORY PATHNAME and SEQUENTIAL ACCESS NAME prompts —identifying the directory and backup file involved in the operation. The only difference is that a MOVE directive cannot supply a list of devices for a multiple volume backup. If you want to use more than one device for a multiple volume backup, you have to use the prompt.
>
> For the CD command, the MOVE directive has the same function as your responses to the INPUT PATHNAME and OUTPUT PATHNAME prompts. For VC, it has the same function as your responses to the MASTER PATHNAME and COPY PATHNAME prompts.
>
> If you leave off the directory pathname, the command assumes you mean the last directory specified in either your response to the DIRECTORY PATHNAME prompt or the prior MOVE directive. Likewise, if you leave off the sequential access name, it assumes you mean the one specified in the prior MOVE directive or prompt response.

OPTIONS option...option

> The OPTIONS directive allows you to specify options in the control file. You can specify any of the choices described for the OPTIONS prompt. If you use BLOCK as an option in a control file, you must *not* specify *both* directory and sequential names in the prompt responses, and the OPTION containing the BLOCK must precede any MOVE in the file. If you use more than one OPTIONS directive, the most recent selection from each pair of choices has effect. If you also give a response to the OPTIONS prompt, that response is overridden by any corresponding options chosen in OPTIONS directives. You cannot use the OPTIONS directive between two EXCLUDE directives for the same directory.

BD (BACKUP DIRECTORY)

INCLUDE filename...filename

The INCLUDE directive allows you to specify which files are copied from the directory backed up. If you do not use INCLUDE directives, the command assumes you want to include all files not specifically excluded with EXCLUDE directives. You cannot mix INCLUDE and EXCLUDE directives for the same directory.

The files listed after INCLUDE must belong to the directory specified in the prior MOVE directive. If there is no prior MOVE directive, they must belong to the directory indicated in response to the input directory prompt. You can list any file types, including subdirectories. However, you cannot specify individual files within the subdirectories.

When you list a file, list only the last component of its pathname. You do not need to list aliases separately. You can use up to 50 INCLUDE directives for each directory, each listing as many files as you can fit on one record.

EXCLUDE filename...filename

The EXCLUDE directive allows you to specify which files are not to be copied from the directory being backed up. If you do not use EXCLUDE directives, the command assumes you want to include all files in the directory, unless you use INCLUDE directives to specifically include one or more files. All the restrictions on INCLUDE also apply to EXCLUDE.

END

The END directive marks the end of the control file.

Special Cases:

Copying a Single File

You can use BD to copy a single file by giving the file name in response to the DIRECTORY PATHNAME prompt. This method allows you to make a backup copy of the file as part of a complex operation using a control file. Unlike Copy/Concatenate (CC), BD can copy crash files as shown in the following example. The volume MAINTAIN, which receives the copy of the crash file, also contains the following control file as MAINTAIN.GETCRASH:

```
MOV .VCATALOG,MAINTAIN.CRASHLOG
INC S$CRASH
END

[] BD

BACKUP DIRECTORY
        DIRECTORY PATHNAME:
    SEQUENTIAL ACCESS NAME:
       CONTROL ACCESS NAME: MAINTAIN.GETCRASH
       LISTING ACCESS NAME: LP
                   OPTIONS: NOSYSFILE
       EXECUTION MODE(F,B): FOREGROUND
```

The operation returns the following listings:

```
  BACKUP DIRECTORY          12:56:04 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:
ORIGINAL DESTINATION:
ORIGINAL OPTIONS:       ALIAS,NODATE,NOBLOCK,NOREWIND,NOUNLOAD,NOSYSFILE,NOFAST
CONTROL FILE:           MAINTAIN.GETCRASH
LIST FILE:              LP

MOV .VCATALOG,MAINTAIN.CRASHLOG
INC S$CRASH
  **        DIRECTORY  .VCATALOG
  **  S$CRASH
  **        END OF DIRECTORY  .VCATALOG

END

ELAPSED TIME = 3 MINUTES   16 SECONDS
SIZE OF INPUT = 430 ADU'S

****** BACKUP DIRECTORY COMPLETED
```

Stacked Directories

If you include more than one MOVE directive in your control file, you can store the backup copies of more than one directory in a single sequential file. If you do this, you must specify the sequential file only once—in your response to the SEQUENTIAL ACCESS NAME prompt or in your first MOVE directive. If you stack directories in a sequential file, a subsequent VB (or RD) command must use a control file with corresponding MOVE directives to unstack the directories. However, if you stack the directories on magnetic tape, you can use multiple RD or VB commands to restore or verify each stacked directory in turn.

In the following example, the BD command stacks backup copies of the directory VOL2.DIR1, the file VOL2.DIR2.PROGRAMS, and all files in the directory VOL2.DIR3 except for VOL2.DIR3.HEADER into the backup file on MT01. To do this, the BD command requires the control file VOL2.STACKBD with the following directives:

```
MOVE VOL2.DIR1
MOVE VOL2.DIR2.PROGRAMS
MOVE VOL2.DIR3
EXCL HEADER
END

[] BD
```

BD (BACKUP DIRECTORY)

```
BACKUP DIRECTORY
        DIRECTORY PATHNAME:
   SEQUENTIAL ACCESS NAME: MT01
      CONTROL ACCESS NAME: VOL2.STACKBD
      LISTING ACCESS NAME: LP
                  OPTIONS: REWIND
      EXECUTION MODE(F,B): BACKGROUND
```

The backup operation returns the following listings:

```
  BACKUP DIRECTORY        13:33:35 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:
ORIGINAL DESTINATION: MT01
ORIGINAL OPTIONS:     ALIAS,NODATE,NOBLOCK,REWIND,NOUNLOAD,SYSFILES,NOFAST
CONTROL FILE:         VOL2.STACKBD
LIST FILE:            LP

MOVE VOL2.DIR1
  **       DIRECTORY  VOL2.DIR1
  **  FILE1
  **  FILE3
  **  FILE2
  **  PROGRAMS - PROGRAM FILE

  **  TASK        ID
  **  PROGA      >03
  **  PROGB      >0E
  **       END OF PROGRAM FILE

  **       END OF DIRECTORY  VOL2.DIR1

MOVE VOL2.DIR2.PROGRAMS
  **       DIRECTORY  VOL2.DIR2
  **  PROGRAMS - PROGRAM FILE
  **  TASK        ID
  **  PROGA      >03
  **  PROGB      >0E
  **       END OF PROGRAM FILE

  **       END OF DIRECTORY  VOL2.DIR2

MOVE VOL2.DIR3
EXCL HEADER
  **       DIRECTORY  VOL2.DIR3
  **  FILEB
  **  FILEA
  **  FILEC
  **       END OF DIRECTORY  VOL2.DIR3

END

ELAPSED TIME = 0 MINUTES   56 SECONDS
SIZE OF INPUT = 2617 ADU'S

****** BACKUP DIRECTORY COMPLETED
```

The following VB command makes sure the backup files match the original directories, using the control file VOL2.STACKVB with the following directives:

```
MOVE ,VOL2.DIR1
MOVE ,VOL2.DIR2.PROGRAMS
MOVE ,VOL2.DIR3
EXCL HEADER
END
```

```
[] VB

VERIFY BACKUP
   SEQUENTIAL ACCESS NAME: MT01
       DIRECTORY PATHNAME:
      CONTROL ACCESS NAME: VOL2.STACKVB
      LISTING ACCESS NAME: LP
                  OPTIONS: REWIND
      EXECUTION MODE(F,B): BACKGROUND
```

The VB command produces the following listings:

```
  VERIFY BACKUP            13:35:18 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       MT01
ORIGINAL DESTINATION:
ORIGINAL OPTIONS:      ALIAS,NODATE,REWIND,NOUNLOAD,SYSFILES,NOFAST
CONTROL FILE:          VOL2.STACKVB
LIST FILE:             LP

MOVE ,VOL2.DIR1

  **       DIRECTORY  VOL2.DIR1
  **  FILE1
  **  FILE3
  **  FILE2
  **  PROGRAMS - PROGRAM FILE

  **  TASK        ID
  **  PROGA      >03
  **  PROGB      >0E
  **       END OF PROGRAM FILE

  **       END OF DIRECTORY  VOL2.DIR1


MOVE ,VOL2.DIR2

  **       DIRECTORY  VOL2.DIR2
  **  PROGRAMS - PROGRAM FILE

  **  TASK        ID
  **  PROGA      >03
  **  PROGB      >0E
  **       END OF PROGRAM FILE

  **       END OF DIRECTORY  VOL2.DIR2
```

BD (BACKUP DIRECTORY)

```
MOVE ,VOL2.DIR3
EXCL HEADER

  **        DIRECTORY  VOL2.DIR3
  **   FILEB
  **   FILEA
  **   FILEC
  **        END OF DIRECTORY  VOL2.DIR3

  **   HEADER    INCLUDE/EXCLUDE FILE NOT FOUND
END

ELAPSED TIME = 0 MINUTES   55 SECONDS
SIZE OF INPUT = 2617 ADU'S
1 WARNINGS ISSUED

****** VERIFY BACKUP COMPLETED
```

The warning results from having an EXCLUDE directive in the control file. It does not indicate an error condition. If you delete the EXCLUDE directive from the control file, the warning message will not appear.

Multiple Volume Backups

A backup file can span more than one disk or magnetic tape. When you want to perform a multiple volume backup, you need to provide a device name or list of device names as your response to the SEQUENTIAL ACCESS NAME prompt. If you specify a disk unit or list of disk units, you receive an additional PATHNAME prompt. The pathname you enter must consist of a volume name (not a device name), followed by a period and a name for the backup file—VOL2.BACKUP for example. The disks you use in the backup operation must be given the same volume name, but not installed. The BD, RD, and VB commands install and unload the disks automatically. A multiple volume backup cannot be done from a batch stream.

The order in which you list the devices determines the order in which the command uses them in the backup, restoration, or verification. If you give a list of devices, the command uses each one in turn before it asks you to mount a new volume. If you specify only one device, the command prompts you for a new volume after it finishes the one mounted. The following message is displayed at your station (with a volume number in place of x):

END OF MEDIA MOUNT VOLUME x; TYPE $ TO QUIT, Y TO CONTINUE

To continue with the operation, you need to mount the new volume in the appropriate drive and press the Y key (but not Return). If you make a mistake, such as mounting the volumes in the wrong sequence, the command displays the message again to allow you to correct the problem and continue. You can quit without completing the operation by pressing the $ key.

When you use a control file, the following message appears each time a disk or tape drive is specified in the control file as the destination:

MOUNT A NEW VOLUME IF DESIRED—TYPE $ TO QUIT, Y TO CONTINUE

This message appears each time a disk or tape drive is specified in a control file as the destination, even the first time and even when the destination is the same as the previous destination. Therefore, it is not always necessary for you to mount a new volume. Enter Y to continue or enter $ if you are ready to terminate the command.

The following commands demonstrate how to perform a multiple volume backup and verify operation. BD backs up all files in VOL2.DIR1 except for VOL2.DIR1.PROGRAMS using a control file entered from the station. The backup file BACKUP82.OCTOBER fills three diskettes, but the only diskette drives available are DS03 and DS04. The operation begins with volume 1 of BACKUP82 mounted on DS03, volume 2 on DS04, and volume 3 held for later.

```
[] BD

BACKUP DIRECTORY
        DIRECTORY PATHNAME: VOL2.DIR1
   SEQUENTIAL ACCESS NAME: DS03,DS04
      CONTROL ACCESS NAME: ME
      LISTING ACCESS NAME: LP
                  OPTIONS:
      EXECUTION MODE(F,B): FOREGROUND

   PATHNAME OF BACKUP FILE
                 PATHNAME: BACKUP82.OCTOBER

  EXCLUDE PROGRAMS
  END
```

After it fills the first two diskettes, BD displays the following message at the station:

END OF MEDIA MOUNT VOLUME 3; TYPE $ TO QUIT, Y TO CONTINUE

After it receives a Y response, indicating that the third volume of BACKUP82 is mounted in DS03, the BD command continues executing, eventually producing the following listings:

```
 BACKUP DIRECTORY        13:40:53 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       VOL2.DIR1
ORIGINAL DESTINATION:  DS03,DS04
ORIGINAL OPTIONS:      ALIAS,NODATE,BLOCK,NOREWIND,NOUNLOAD,SYSFILES,NOFAST
CONTROL FILE:          ST03
LIST FILE:             LP
```

BD (BACKUP DIRECTORY)

```
EXCLUDE PROGRAMS
  **       DIRECTORY  VOL2.DIR1
  **  FILE1
  **  FILE3
  **  FILE2
  **       END OF DIRECTORY  VOL2.DIR1

END

ELAPSED TIME = 0 MINUTES   21 SECONDS
SIZE OF INPUT = 123 ADU'S

****** BACKUP DIRECTORY COMPLETED
```

The following VB command makes sure the contents of the multiple volume backup file match the original directory. Since it is not necessary to exclude the file VOL2.DIR1.PROGRAMS from the verification, the VB command requires no control file.

```
[] VB

VERIFY BACKUP
   SEQUENTIAL ACCESS NAME: DS03,DS04
       DIRECTORY PATHNAME: VOL2.DIR1
     CONTROL ACCESS NAME:
     LISTING ACCESS NAME: LP
                 OPTIONS:
     EXECUTION MODE(F,B): BACKGROUND

   PATHNAME OF BACKUP FILE
                PATHNAME: BACKUP82.OCTOBER
```

The VB command produces the following listings:

```
  VERIFY BACKUP           13:46:45 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       DS03,DS04
ORIGINAL DESTINATION:  VOL2.DIR1
ORIGINAL OPTIONS:      ALIAS,NODATE,NOREWIND,NOUNLOAD,SYSFILES,NOFAST
CONTROL FILE:
LIST FILE:             LP


  **       DIRECTORY  VOL2.DIR1
  **  FILE1
  **  FILE3
  **  FILE2
  **       END OF DIRECTORY  VOL2.DIR1


ELAPSED TIME = 0 MINUTES    8 SECONDS
SIZE OF INPUT = 123 ADU'S

****** VERIFY BACKUP COMPLETED
```

BD (BACKUP DIRECTORY)

Related Commands:

BDD	(Backup Directory to Device)
RD	(Restore Directory)
VB	(Verify Backup)

BDD (BACKUP DIRECTORY TO DEVICE)

4.19 BDD (BACKUP DIRECTORY TO DEVICE)

Prompts:

```
BACKUP DIRECTORY TO DEVICE
                SOURCE DISK UNIT:  devicename          (*)
                SOURCE PATHNAME:   pathname@
     DESTINATION DEVICE NAME:      devicename          (*)
          DESTINATION FILE NAME:   [pathname]@
                  LISTING DEVICE:  devicename@         (*)
                        VERIFY?:   YES/NO              (YES)
                  MORE BACKUPS?:   YES/NO              (*)
```

Purpose:

The BDD command allows you to make a sequential backup of a disk or a selected directory. Optional verification is available, and the backup may be made to either disk or magnetic tape. Once the backup is made, you can restore the backup to a disk by entering the Restore Directory (RD) command. When you restore the backup, you must restore to a disk with the same sector size as the original source disk. The backup file that is created is an image file. When you use the RD command, you must specify a device name, not a file name for the SEQUENTIAL ACCESS NAME prompt. Another prompt then asks for the file name.

BDD performs limited disk and file compression. Program files are partially compressed by releasing unused space from the end of the file, while space made available elsewhere by previously deleted tasks is not released.

The backup file that is created is an image file. When you use the Recover Directory (RD) command, you must specify a device name, not a file name for the SEQUENTIAL ACCESS NAME prompt. Another prompt then asks for the file name.

This utility does not require the system disk to remain mounted and, therefore, can back up a data disk in a two-disk system. Whenever the system disk drive is involved in the backup process, BDD requires that no other tasks, terminals, or jobs (including the spooler job) be active while it executes. The Backup Directory (BD) command does not require this. BD also offers options not available with BDD, such as the capability to back up files selected by the date of their creation or last modification. You may sometimes prefer to use BD, though BDD is generally faster.

BDD (BACKUP DIRECTORY TO DEVICE)

The BDD utility unloads the destination disk before it executes, and leaves it unloaded upon completion. Therefore, the disk cannot be accessed during execution of BDD, and afterward requires the Install Volume (IV) command to be performed before it can be accessed.

The proper procedure for using BDD when the system disk is involved (either as source or destination disk) is as follows:

1. Perform an initial program load (IPL) and log on to the system.
2. Use the Execute Operator Interface (XOI) command to become the system operator.
3. If you are using a printer as the listing device, use the Modify Spooler Device (MSD) command to make the printer shared or not available to the spooler.
4. Use the List Jobs (LJ) command to determine if any other jobs are running.
5. Use the Kill Jobs (KJ) command to terminate all other jobs excluding the system job and your job under which the BDD is to be performed.
6. Issue the Quit Operator Interface (QOI) command.
7. Execute BDD.
8. When BDD completes, perform an IPL and normal start-up procedures. BDD cannot be used from a batch stream.

BDD cannot be used from a batch stream.

Prompt Details:

SOURCE DISK UNIT:
The name of the source disk unit (where the disk you want to back up is mounted) in the form DSxx where xx is a disk unit number from 01 to 99.

SOURCE PATHNAME:
The pathname identifying the topmost directory of a set of files to be backed up. If an entire disk is to be backed up, enter the volume name. The pathname can identify a single file when only one file is to be backed up.

DESTINATION DEVICE NAME:
The name of the destination disk unit (DSxx) or tape unit (MTxx).

DESTINATION FILE NAME:
If you specified a tape device for the previous prompt, do not respond to this prompt. If you specified a disk device for the previous prompt, respond with a pathname for the backup file on the destination disk. The pathname must consist of only the volume name and a file name for the backup file. When you use multiple disks for backup, all disks must have the same volume name.

BDD (BACKUP DIRECTORY TO DEVICE)

LISTING DEVICE:
The name of the terminal (STxx) or printer (LPxx) which is to be the listing device. If a listing is not required, enter DUMY. It is recommended that a hard copy listing be made for record keeping purposes.

VERIFY?:
Although it is not recommended, a backup can be performed without verification if you enter NO in response to this prompt.

MORE BACKUPS?:
Enter YES if you want to make a series of backups. Your terminal will then display the full set of prompts for the next backup to be made. You can make as many as nine backups in a single execution of BDD.

Examples:

The following example shows a backup operation in which two backups are requested. For some of the prompts, the user has accepted the initial values.

In this example the disk being backed up in the first operation is PAE29, mounted in DS02. The entire disk is being backed up to a file with the pathname SMITH.JOB1 on a disk mounted in DS01. A printer (LP01) is specified as the listing device, so a hard copy will be made of the results of the backup operation. This permanent listing can be useful if any problems occur. The backup is to be verified.

Before execution begins on the first backup, the BDD command prompts are displayed for the second backup. The responses for the SOURCE DISK UNIT, DESTINATION DEVICE NAME, and LISTING DEVICE prompts are retained from the first backup operation. The user accepted them along with the other initial values, and specified new values for the remaining prompts.

```
BACKUP DIRECTORY TO DEVICE
        SOURCE DISK UNIT: DS02
         SOURCE PATHNAME: PAE29
 DESTINATION DEVICE NAME: DS01
   DESTINATION FILE NAME: SMITH.JOB1
           LISTING DEVICE: LP01
                  VERIFY?: YES
            MORE BACKUPS?: YES


BACKUP DIRECTORY TO DEVICE
        SOURCE DISK UNIT: DS02
         SOURCE PATHNAME: PAE29
 DESTINATION DEVICE NAME: DS01
   DESTINATION FILE NAME: JONES.JOB1
           LISTING DEVICE: LP01
                  VERIFY?: YES
            MORE BACKUPS?: NO
```

BDD (BACKUP DIRECTORY TO DEVICE)

To restore the second backup, you must use the following responses to the RD command prompts:

```
RESTORE DIRECTORY
SEQUENTIAL ACCESS NAME: DS01
    DIRECTORY PATHNAME: DS02.OLD
   CONTROL ACCESS NAME:
   LISTING ACCESS NAME:
               OPTIONS: REP
  EXECUTION MODE (F,G): FOREGROUND

 PATHNAME OF A BACKUP FILE
             PATHNAME: JONES.JOB 1
```

Figure 4-1 is an example of the VDT screen upon normal completion of the first backup. The numbered items in Figure 4-1 are described in Table 4-6.

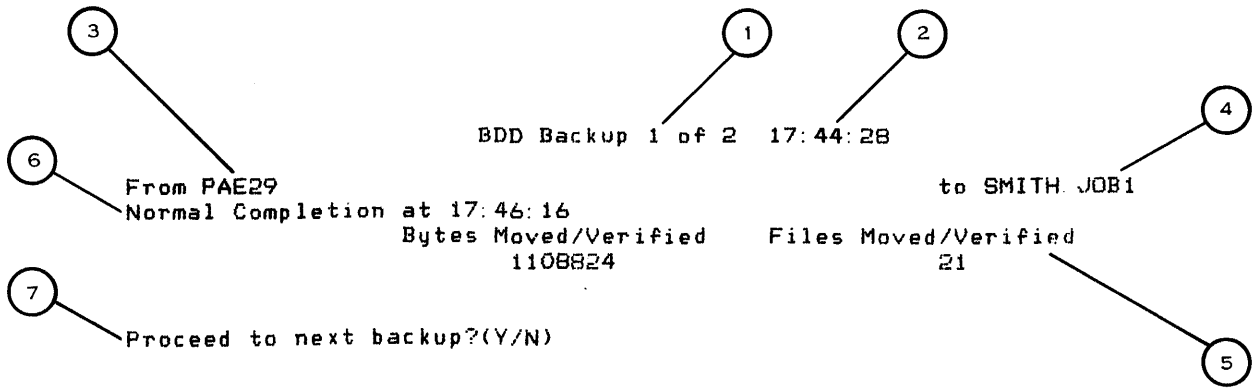


Figure 4-1. Screen Display Upon Normal BDD Completion

BDD (BACKUP DIRECTORY TO DEVICE)

The next example shows an operation that backs up a directory with the pathname SYS63B.MYDATA from a disk to magnetic tapes. Since the backup is to a tape device, no response is made to the DESTINATION FILE NAME prompt. A VDT (ST04) is specified as the listing device, so no permanent listing will be made of the results of the operation. The backup is being verified, and no additional backups are requested.

```
BACKUP DIRECTORY TO DEVICE
        SOURCE DISK UNIT: DS01
         SOURCE PATHNAME: SYS63B.MYDATA
    DESTINATION DEVICE: MT01
DESTINATION FILE NAME:
        LISTING DEVICE: ST04
               VERIFY?: YES
         MORE BACKUPS?: NO
```

Figure 4-2 shows an example of the VDT screen after the backup operation abnormally completes. In this example, the system disk drive is used. The numbered items in Figure 4-2 are described in Table 4-6.

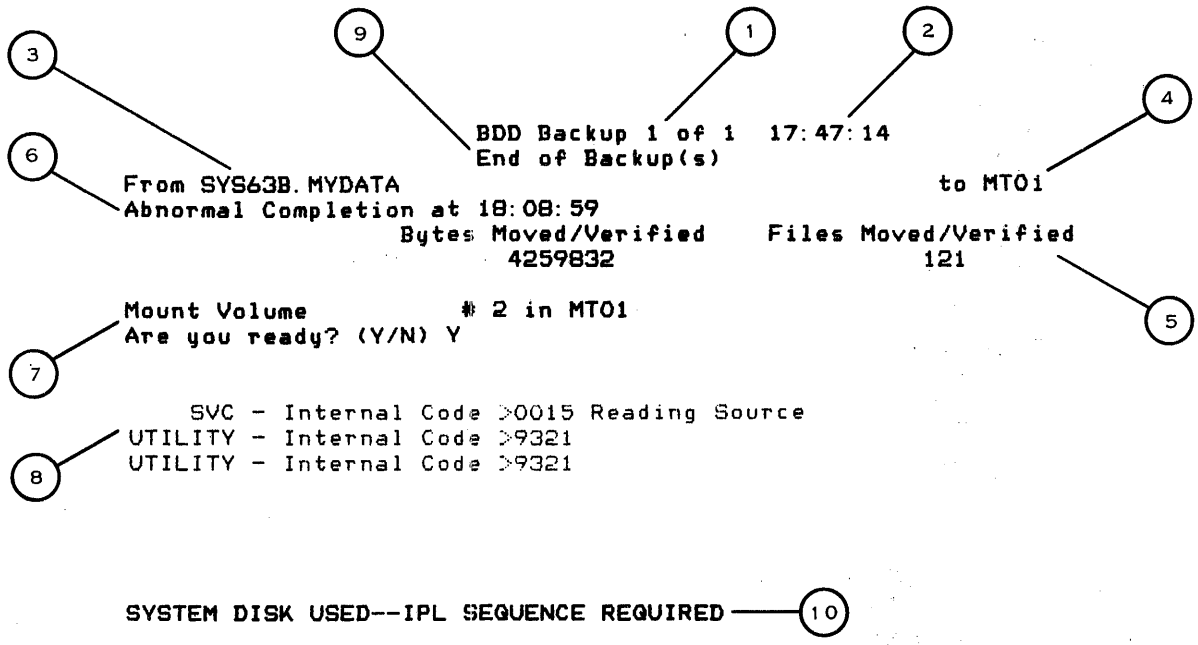


Figure 4-2. Screen Display Upon Abnormal BDD Completion

BDD (BACKUP DIRECTORY TO DEVICE)

Table 4-6. BDD Screen Display

Field	Description
1.	The number of the backup in progress.
2.	The time at which the backup began.
3.	Pathname of the source directory.
4.	Pathname of the destination file or the name of the destination device.
5.	Running tallies of the number of bytes moved or verified, and the number of files moved or verified.
6.	Completion message for the current backup with time of completion; indicates whether normal or abnormal completion.
7.	Various messages requiring user response.
8.	Error list and bad file list.
9.	Indicates termination of all requested backups.
10.	If the system disk drive was used, this message appears after termination of the final backup.

Messages:

The following informative messages can appear on your terminal while executing BDD.

Initializing Backup of < SOURCE PATHNAME> to < DESTINATION DEVICE>
: This message is displayed briefly after you respond to the final prompt of the BDD command. It informs you that BDD has started execution.

All other active Terminals, Jobs, and Tasks must be terminated
Are you ready?(Y/N)
: This message is displayed if either the source or destination disk is in the system disk drive and any other terminals, jobs, or tasks are active. Before continuing you must have all other terminals log off, and you must wait until any active jobs or tasks complete. If you respond Y, but other terminals or tasks are still active, the same message is displayed again. When all other tasks complete, all jobs terminate, and all other terminals log off, respond Y and the BDD command begins execution. If you are not ready to continue with BDD, enter N.

BDD (BACKUP DIRECTORY TO DEVICE)

Do you want to quit? (Y/N)

This message is displayed when you respond N to the *Are you ready? (Y/N)* prompt. If you respond Y, the BDD command terminates. If you respond N, and other terminals, jobs, or tasks are still active, the *Are you ready? (Y/N)* prompt is displayed again.

System drive available, type Y to continue

This message is displayed when the system disk drive is involved in the backup operation and is available for use. Enter Y, and BDD begins producing the first backup.

Summary of Progress Messages

During execution of BDD at a VDT, the upper half of your screen displays a summary of the progress of the backup operation, while the lower half of the screen is reserved for error messages. The summary display includes the date and time at which the backup began, the pathnames of the source directory and the destination file or device, a running tally of the backup operation, and a series of informative messages. Most messages require no action on your part unless attention to an error condition is required or a disk or tape needs to be replaced or removed.

Mount Volume < VOLUME NAME> #N in DSxx
Are you ready? (Y/N)

When a single destination disk cannot contain the entire backup file, BDD displays this message and pauses to allow you to change the disk. N indicates the number of the volume in the multiple disk backup sequence. Mount the next disk, then enter Y for BDD to continue.

Mount Volume #N in MTxx
Are you ready? (Y/N)

When a single destination tape cannot contain the entire backup file, BDD displays this message and pauses to allow you to change the tape. N indicates the number of the volume in the multiple tape backup sequence, but is not part of the volume name. All destination tapes used in multiple tape backups must have the same volume name. Mount the next tape, then enter Y for BDD to continue.

Start Tape Verification

When you select the verification option with a tape backup, this message is displayed when BDD starts tape verification. During tape verification, BDD advises you whenever it is necessary to change tapes.

Normal completion at HH:MM:SS

This message is displayed upon successful completion of each disk or tape backup, with the time at completion displayed in hours, minutes, and seconds.

Proceed to next backup? (Y/N)

When you request more than one backup, this message is displayed after each backup completion. When performing a series of backups, BDD prompts you each time (except the first) to mount both the source and destination volumes, even if the source volume is the same each time. If you wish to discontinue the series of backups, enter N. Otherwise enter Y to proceed to the next backup.

BDD (BACKUP DIRECTORY TO DEVICE)

End of Backup(s)
: This message indicates that the final backup operation is complete. Press the Return key to return to SCI.

SYSTEM DISK USED—IPL SEQUENCE REQUIRED
: If the system disk drive was involved in the backup operation, this message informs you that an IPL is required to bring up DNOS.

Error Messages
: During execution of BDD, any errors that are encountered are listed on the lower half of your screen. The errors appear in one of the two following forms:

SVC	—	Internal Code > 0018 on Source Disk
UTILITY	—	Internal Code > 9321

: The error code is a four-digit hexadecimal number representing the error returned from an SVC or from BDD. Use the Show Expanded Message (SEM) command to examine these internal error codes or refer to the specified error number in the *DNOS Messages and Codes Reference Manual* which includes a description of the error and suggested user actions.

Abnormal Completion at HH:MM:SS
: Under certain circumstances, BDD is unable to successfully complete the copy, and terminates the operation. The individual errors that led to the abnormal completion are listed on the lower half of your screen.

: A special case of abnormal completion occurs when the destination volume name is changed by BDD to $$$$$$$$. In this situation, the integrity of the destination disk has been destroyed and the Initialize New Volume (INV) command must be executed.

Notes:

If you use BDD from a terminal that is not a VDT or is a VDT not in VDT mode, the display of progress will not be shown.

All file security information for files on the source disk will be maintained on the destination disk.

BDD cannot backup files with a pathname (of any length) ending in any of the following: .VCATALOG, .S$DIAG, .S$PRINT, .S$ROLLA, .S$ROLLD, or .S$SDTQUE. To backup these files, you must first change the file names using the Modify File Name (MFN) command. There is normally no need to backup system files with these pathnames.

Related Commands:

BD	(Backup Directory)
CD	(Copy Directory)
CV	(Copy Volume)
RD	(Restore Directory)
VB	(Verify Backup)

BEMF (BUILD EXPANDED MESSAGE FILE)

4.20 BEMF (BUILD EXPANDED MESSAGE FILE)

Prompts:

```
BUILD EXPANDED MESSAGE FILE
                INPUT FILENAME:  [site:]filename@
               OUTPUT FILENAME:  [site:]filename@
           ERROR ACCESS NAME:    [site:]filename@      (DUMY)
CONVERT LOWER TO UPPER CASE?:    YES/NO                (NO)
  MAXIMUM MESSAGE ID LENGTH:     integer               (4)
```

Purpose:

The BEMF command creates an expanded message file for the S$EXPMSG directory from an edited source file of expanded message text. This allows you to create your own message files. BEMF reads the text file, reformats an entire explanation into one record, and writes that record to the key indexed file specified by the response to the OUTPUT FILENAME prompt. If you must modify system message files for internationalization or for some other reason, the files from the .MESSAGES.TEXT and .MESSAGES.EXPTEXT directories in the delivered system must be edited with great care by a systems programmer.

Prompt Details:

INPUT FILENAME:
The file name of the expanded explanation file from which to build the message. The format of this file is described in the *DNOS Systems Programmer's Guide*.

OUTPUT FILENAME:
The file name which contains the formatted messages. To be usable, the file must reside in (or be moved to) the .S$EXPMSG directory.

ERROR ACCESS NAME:
The file name where any errors encountered during formatting are listed. The messages in error are identified and the utility continues to the next message to be processed. In this way, all erroneous messages are identified after one execution of the utility. If errors do occur during execution of BEMF, delete the output file and recreate it after you correct the input text file.

CONVERT LOWER TO UPPER CASE?:
If you specify YES, all lowercase alphabetic characters are translated to uppercase so that English explanations can be written to terminals which do not support lowercase English. Conversion of local language lowercase characters to uppercase characters is not available. If NO, lowercase characters are not translated to uppercase English.

The conversion of lowercase to uppercase is not done for files of expanded messages produced by Texas Instruments. Since these explanations are accessed interactively at VDT terminals, they can be read in uppercase and lowercase format.

MAXIMUM MESSAGE ID LENGTH:
An integer value specifying the maximum length of the message ID. A key indexed file which is built by BEMF has the capability of handling message IDs of up to 8 characters.

BEMF (BUILD EXPANDED MESSAGE FILE)

Example:

In the following example, BEMF builds the message file .KC0017.MESSAGE from the text file .KC0017.TEXT. There is no translation of lowercase characters to uppercase characters. The maximum message ID is four characters in length. This file must later be moved to the .S$EXPMSG directory in order to be used by the system.

```
[]BEMF

BUILD EXPANDED MESSAGE FILE
                INPUT FILENAME: .KC0017.TEXT
               OUTPUT FILENAME: .KC0017.MESSAGE
            ERROR ACCESS NAME: .KC0017.LIST
CONVERT LOWER TO UPPER CASE?: NO
   MAXIMUM MESSAGE ID LENGTH: 4
```

Related Commands:

BMF (Build Message File)
SEM (Show Expanded Message)

BL (BACKSPACE LUNO)

4.21 BL (BACKSPACE LUNO)

Prompts:

```
BACKSPACE LUNO
                        LUNO:  integer
              RECORD COUNT:  integer
```

Purpose:

The BL command moves the current data-access position backward one or more records.

Prompt Details:

LUNO:
The logical unit assigned to the sequential device or file to be backspaced.

RECORD COUNT:
The number of records (from 1 through 65535) that the sequential file or device is to be backspaced. If an end-of-file is encountered before backspacing the number of records specified, the sequential device or file is positioned immediately preceding the end-of-file.

Example:

The following example backspaces a magnetic tape unit (assigned LUNO 32) four records using the BL command:

```
[] BL

BACKSPACE LUNO
                        LUNO: 32
              RECORD COUNT: 4
```

Notes:

The operations performed by the BL command include an open call and a close call to the device.

Related Commands:

FL (Forward Space LUNO)
RWL (Rewind Logical LUNO)

4.22 BMF (BUILD MESSAGE FILE)

Prompts:

```
BUILD MESSAGE FILE
                    INPUT FILENAME:  [site:]filename@
                   OUTPUT FILENAME:  [site:]filename@
      OUTPUT FILE TYPE (REL,SEQ):    {REL/SEQ}          (REL)
                ERROR ACCESS NAME:   [site:]filename@   (DUMY)
MAXIMUM MESSAGE TEXT LENGTH:         integer            (80)
```

Purpose:

The BMF command builds a file of brief messages from input contained in a text file that can be edited. To be usable, the BMF output file must reside in the .S$MSG directory. Messages from the S$MSG files must be printable on VDTs, printers, and any other device. Therefore, the .MESSAGES.TEXT files and the corresponding S$MSG files supplied by Texas Instruments are in all uppercase (for English), and no attempt is made to translate lowercase characters to uppercase.

Prompt Details:

INPUT FILENAME:
The file name of the text file from which to build the message file. The format of this file is described in the *DNOS Systems Programmer's Guide*.

OUTPUT FILENAME:
The file name that contains the formatted message. Any file name may be used, but the output must reside in the .S$MSG directory if it is to be used for reporting errors.

OUTPUT FILE TYPE:
If you specify REL, the output file is formatted as a relative record file. If you specify SEQ, the output file is formatted as a sequential file. Sequential files allow compact file sizes, while relative record files allow faster access.

ERROR ACCESS NAME:
The file name to which any errors encountered during formatting are listed.You can correct errors and execute the BMF command again to rebuild the S$MSG file.

MAXIMUM MESSAGE TEXT LENGTH:
An integer value specifying the logical record length of the input file. The legal range of values is 60 through 240.

BMF (BUILD MESSAGE FILE)

Example:

In the following example, the BMF command builds a message file from a text file and outputs the formatted message to the specified message file. This file must later be moved to the .S$MSG directory in order to be used by the system.

```
[]BMF

BUILD MESSAGE FILE
                INPUT FILENAME: .KC0017.TEXT
               OUTPUT FILENAME: .KC0017.MESSAGE
   OUTPUT FILE TYPE (REL,SEQ): SEQ
             ERROR ACCESS NAME: .KC0017.ERROR
  MAXIMUM MESSAGE TEXT LENGTH: 100
```

Notes:

There is no translation of lowercase characters to uppercase characters when the BMF command is executed; therefore, the text file used for input must be in all uppercase characters.

Related Commands:

BEMF	(Build Expanded Message File)
SEM	(Show Expanded Message)

CAG (CREATE ACCESS GROUP)

4.23 CAG (CREATE ACCESS GROUP)

Prompts:

```
CREATE ACCESS GROUP
        ACCESS GROUP NAME:   alphanumeric
              ADD USER ID(S):  [alphanumeric...alphanumeric]
```

Purpose:

The CAG command permits a user to create an access group if it does not currently exist on the system. The user issuing the command becomes the leader of the access group. Other users can also be added to the access group with this command.

The system determines to which access groups a user belongs when a job is created with the user's ID. Modifications to an access group's membership do not take effect until a new job is created with that user's ID. If currently logged on to SCI, the user must log off and log on again.

Prompt Details:

ACCESS GROUP NAME:
A string of one to eight alphanumeric characters with the first character being alphabetic.

ADD USER ID(S):
Enter the user ID(s) of the user(s) to be added as members of the access group. The user IDs must be valid, existing user IDs.

Example:

In the following example, a user who is logged on with a user ID of CHRIS executes the CAG command. The CAG command creates an access group named MANAGERS, which includes members with user IDs of MARY, FRED, and CHRIS. The user with the user ID CHRIS becomes the access group leader.

```
[] CAG

CREATE ACCESS GROUP
      ACCESS GROUP NAME: MANAGERS
         ADD USER ID(S): MARY,FRED
```

Notes:

If the user ID of the user issuing the command is a member of the access group SYSMGR, an error is returned. Members of the access group SYSMGR are not allowed to be leaders (or even members) of another group. For the same reason, you cannot add a SYSMGR group member to another group.

CAG (CREATE ACCESS GROUP)

Related Commands:

DAG	(Delete Access Group)
LAGM	(List Access Group Members)
MAG	(Modify Access Group)

4.24 CALL (CALL TERMINAL)

Prompts:

```
CALL TERMINAL
        TERMINAL ACCESS NAME:  stationname       (*)
            TELEPHONE NUMBER:  [character(s)]
                  ANSWERBACK:  [character(s)]    (*)
```

Purpose:

Use the CALL command to establish a connection with a remote terminal when the call is initiated by the host system.

Prompt Details:

TERMINAL ACCESS NAME:

A valid system terminal name. The terminal must be logically turned on and not currently in use. If auto-dialing is to be performed, the terminal must have an associated Auto Call Unit (ACU).

TELEPHONE NUMBER:

The telephone number which is to be dialed. Response to this prompt is optional and can be omitted. If omitted, manual dialing is indicated. The telephone number consists of the digits 0 through 9 and the following special characters:

= Used to signify the need to wait for a dial tone. Only applicable if the ACU supports this function.

* Used on some touch-tone telephones for special functions.

\# Used on some touch-tone telephones for special functions.

– Used only to improve readability and is ignored during the dialing sequence.

ANSWERBACK:

The ANSWERBACK prompt is a well-defined code used to identify a particular teleprinter device (TPD). This answerback code must be previously defined at the remote TPD. If you request an answerback, upon completion of the connection, the system delays for a specified time to allow the remote terminal to auto-answerback. A null response specifies that the terminal answerback is unknown or undefined.

CALL (CALL TERMINAL)

Example:

The following example auto-dials the TPD using the port established as ST21 during system generation. An answerback of BMT000 is expected from the TPD.

```
[] CALL

CALL TERMINAL
 TERMINAL ACCESS NAME: ST21
     TELEPHONE NUMBER: 837-1395
           ANSWERBACK: BMT000
```

Messages:

A completion message appears when a connection is established with a TPD. This message indicates whether answerback data is present and whether data is expected.

The received answerback consists of ASCII characters ranging from > 20 through > 7E. Any nongraphic characters are deleted from the answerback upon reception. The received answerback is compared with the user-supplied answerback and is saved as the value of the synonym ABM. If no answerback is received, this synonym is deleted.

Notes:

If the terminal answerback contains lowercase data, the user-supplied answerback must contain the same lowercase data for a successful comparison.

The user-supplied answerback is retained as a synonym and used as a default display on subsequent CALL commands.

Related Commands:

ANS	(Answer Incoming Call)
DISC	(Terminal Disconnection)
LHPC	(List Hardcopy Terminal Port Characteristics)
MHPC	(Modify Hardcopy Terminal Port Characteristics)

CB (CREATE BATCH STREAM)

4.25 CB (CREATE BATCH STREAM)

Prompts:

```
CREATE BATCH STREAM
        DIRECTORY TO PROCESS:   [site:]pathname
    BATCH TEMPLATE FILENAME:    [site:]filename
      OUTPUT BATCH FILENAME:    [site:]filename
        SUBSTITUTE FOR < D> :   [filename]
  PROCESS DIRECTORIES ONLY?:    YES/NO              (NO)
```

Purpose:

The CB command generates a batch stream according to a template file of commands and a directory to which they are applied.

The CB command performs a List Directory (LD) of the specified directory. For each file in the directory, the portion of the template file between exclamation points is placed into the output batch file. The file name is substituted for a < ?> in the template.

Prompt Details:

DIRECTORY TO PROCESS:
The directory that contains the files to be processed in the batch stream.

BATCH TEMPLATE FILENAME:
The file name which contains the set of procedures and SCI primitives that will be performed for each file in the directory specified in response to the DIRECTORY TO PROCESS prompt.

The file specified in response to this prompt has the following format and contents:

1. The BATCH command that indicates the beginning of the batch stream.

2. A prologue consisting of any number of comment lines and/or command to be executed (only once). These comment lines appear once in the batch stream being constructed.

3. A line with an exclamation mark (!) as its first character.

4. The set of primitives and command procedures to be performed for each file in the directory being processed. The character string < ?> will be replaced by the name of each file in the directory being processed. The number of entries in the directory is substituted for a < M> in the template.

5. A line with an exclamation mark (!) as its first character to mark the end of the set of commands.

6. Any comments and/or commands to be executed only once at the end of the batch stream.

7. The EBATCH command to indicate the end of the batch stream.

CB (CREATE BATCH STREAM)

OUTPUT BATCH FILENAME:
The name of the file that is to contain the batch stream built by the CB command.

SUBSTITUTE FOR <D>:
The set of primitives and command procedures in the input batch file can include pathnames with embedded characters of <D>. The response to this prompt is inserted in place of that character string when the batch stream is built. For example, it is common to include a directory name for a very general template file. The default value is no substitution.

PROCESS DIRECTORIES ONLY:
If you enter YES, only subdirectories of the directories specified in response to the DIRECTORIES TO PROCESS prompt are processed. That is, only directory names are used in the created batch stream. If you enter NO, both directory and file names are used in the created batch stream.

Example:

The following example shows a template file used and the batch stream created by the CB command. Assume that the template file ME.TEMPLATE has the following contents:

```
BATCH
* THIS BATCH STREAM WILL PRINT ALL MEMBERS OF THE ME.SOURCE
* DIRECTORY.  IT USES THE INITIAL VALUES FOR MOST OF THE
* PRINT FILE PROMPTS.
!
PF  FILE PATHNAME=ME.SOURCE.<?>,
    LISTING =LP01,
    BANNER  =NO
!
EBATCH
```

Also assume that the directory ME.SOURCE has three files, MARK1, MARK2, and MARK3. The following responses to the CB command prompts are entered to create a batch stream contained in the file ME.BATCH:

```
[] CB

CREATE BATCH STREAM
     DIRECTORY TO PROCESS: ME.SOURCE
  BATCH TEMPLATE FILENAME: ME.TEMPLATE
    OUTPUT BATCH FILENAME: ME.BATCH
       SUBSTITUTE FOR <D>:
PROCESS DIRECTORIES ONLY?: NO
```

CB (CREATE BATCH STREAM)

The file ME.BATCH contains the following batch stream and can be executed using the Execute Batch (XB) command to accomplish the printing:

```
BATCH
* THIS BATCH STREAM WILL PRINT THE MEMBERS OF THE ME.SOURCE
* DIRECTORY.  IT USES THE INITIAL VALUES FOR MOST OF THE
* PRINT FILE PROMPTS.
PF   FILE PATHNAME=ME.SOURCE.MARK1,
     LISTING =LP01,
     BANNER  =NO
PF   FILE PATHNAME=ME.SOURCE.MARK2,
     LISTING =LP01,
     BANNER  =NO
PF   FILE PATHNAME=ME.SOURCE.MARK3,
     LISTING =LP01,
     BANNER  =NO
EBATCH
```

Notes:

You must often edit the batch stream created by the CB command to remove operations on particular directory entries if the directory is not of uniform composition (for example, a directory that contains source and object files).

Also, the batch template file can contain CB commands for additional batch streams.

Related Commands:

XB (Execute Batch)
XBJ (Execute Batch Job)

CC (COPY/CONCATENATE)

4.26 CC (COPY/CONCATENATE)

Prompts:

```
COPY/CONCATENATE
        INPUT ACCESS NAME(S):  [site:]pathname@...[site:]pathname @  (*)
        OUTPUT ACCESS NAME:    [site:]pathname@
                REPLACE?:      YES/NO                                (NO)
    MAXIMUM RECORD LENGTH:     [integer]
```

Purpose:

The CC command copies one or more files of information from physical devices or disk storage into one file, concatenates them, and makes them into a single file accessible under one file name. The only limitation on the number of files that can be concatenated with one CC command is the space allowed for entry of pathnames or device names (50 characters total) that identify the files or the source of the files to be concatenated. You can save space by entering synonyms for pathnames or device names.

If the output file does not exist, the CC command creates it automatically according to the type of the first input file. When the input is a device, the output file is created as an expandable, blank-suppressed sequential file with a logical record length of 80 bytes. If you do not want these parameters, you must explicitly create the output file with the desired parameters. If the input is a file, the output file is created with exactly the same characteristics except that the current total number of records in the input file is specified as the primary allocation of the output file and the secondary allocation is defaulted. The maximum record length determines how many bytes to transfer per I/O operation while copying. It does not affect the record length of the output file. The write protect, delete protect, and forced write characteristics are not copied.

You can also use the CC command, like the Print File (PF) command, to copy one or more files to a line printer or to some other output device. When an output device needs carriage control, the CC command adds single space control to the output. If the file being copied contains carriage control characters, the CC command uses those characters for the output.

Prompt Details:

INPUT ACCESS NAME(S):
: One or more pathname(s), separated by commas, that specify a file or files that are to be concatenated into one output file. The CC command concatenates the files in the order that input pathnames or device names are entered. The number of files that can be concatenated is limited by the number of pathnames, file names, or synonyms that can fit in the space provided for a response (50 characters).

OUTPUT ACCESS NAME:
: The device or file name to which the concatenated files should be written. If the pathname does not specify an existing file, the CC command automatically creates it. The device name or pathname specified must be different from the input access name.

CC (COPY/CONCATENATE)

REPLACE?:
YES specifies that the file being copied is to replace the current contents of the output file specified if one exists. NO specifies that the command is to be aborted if the output file already exists.

MAXIMUM RECORD LENGTH:
The maximum length of the records to be copied. This prompt can be used to control the number of characters copied from each record. If the number supplied is smaller than the actual size of an input record then, only that many characters are copied to the output record. The default value is 512 bytes.

Example:

The following example concatenates three disk files to one disk file:

```
[] CC

COPY/CONCATENATE
     INPUT ACCESS NAME(S): .FILE1,.FILE2,.FILE3
       OUTPUT ACCESS NAME: .TEST
                  REPLACE?: NO
    MAXIMUM RECORD LENGTH: 80
```

Notes:

The CC command cannot copy directories, program files, or image files. It cannot copy sequential files that contain embedded end-of-file records. Special commands exist to copy these files, primarily the Copy Directory (CD) and Copy Sequential Media (CSM) commands. The CC command can copy KIFs, but two restrictions apply. The output file pathname must specify an existing file, and the output file must be a sequential file.

For blank-suppressed, nonexpandable sequential files, the destination file is created larger than the source file if the destination file does not exist. To copy such files, use the CD command, which is discussed later in this section.

Related Commands:

AF	(Append File)
CD	(Copy Directory)
CKS	(Copy KIF to Sequential File)
CSM	(Copy Sequential Media)
PF	(Print File)

CD (COPY DIRECTORY)

4.27 CD (COPY DIRECTORY)

Prompts:

```
COPY DIRECTORY
                INPUT PATHNAME:  [pathname]@                          (*)
               OUTPUT PATHNAME:  [pathname]@                          (*)
          CONTROL ACCESS NAME:   [site:][pathname]@
          LISTING ACCESS NAME:   [site:][pathname]@
                      OPTIONS:   [character(s)...character(s)]        (ADD)
        EXECUTION MODE(F,B):     {F/FOREGROUND/B/BACKGROUND}(FOREGROUND)
```

Depending on your response to the OPTIONS prompt, you might receive one or more additional prompts:

```
DATE OF OLDEST FILE
                      YEAR:  integer
                     MONTH:  integer
                       DAY:  integer
                      HOUR:  [integer]                                (0)
                    MINUTE:  [integer]                                (0)

DESTINATION PHYSICAL RECORD LENGTH
        PHYSICAL RECORD LENGTH:  integer                              (768)
```

Purpose:

The CD command copies files from one directory to another directory. You can check the accuracy of the copy by using the Verify Copy (VC) command.

The CD command provides numerous options that let you select the files you want to include in the copy. If the output directory does not have a file with the same name as one being copied, CD automatically creates the file. If the output directory already has a file with that name, CD replaces it with the file from the input directory only if you choose the REPLACE option. CD never changes the contents of the input directory.

Unless you tell it otherwise, CD copies all the files (and aliases) in the input directory, including any subdirectories and the files they contain. CD cannot copy the system files .S$ROLLD.S$ROLLA, .S$DIAG, and .S$CRASH or the spooler directory .S$SDTQUE when copying .VCATALOG unless they are specifically included using the INCLUDE directive in a control file.

CD allows you to use a *control file* containing directives that identify the input directory, output directory, and files involved in the operation. For information about the control file directives, see the notes for BD.

Do not have other file activity going to the directory while it is being copied. Activities such as creating files can cause inconsistent file structures to be created if a copy is in progress.

CD (COPY DIRECTORY)

Prompt Details:

INPUT PATHNAME:

The pathname of the directory that contains the files you want to copy. Unless you limit the command with options or control file directives, CD copies all files (and aliases) in this directory, including any subdirectories and files they contain. You can skip this prompt only if you specify the input pathname with a MOVE directive in a control file.

When you use a disk name as the first part of the pathname, CD inserts .VCATALOG after that name to generate the full pathname. This may cause the name to exceed 48 characters. To avoid this situation, use the volume name of the disk as the first component of the pathname.

OUTPUT PATHNAME:

The pathname for the directory that you want to receive the copied files. If this directory does not exist, CD creates it with the same number of entries as the input directory. You must give the pathname of a directory, even if you only want to copy one file. You can skip this prompt only if you specify the output pathname with a MOVE directive in a control file.

When you use a disk name as the first part of the pathname, CD inserts .VCATALOG after that name to generate the full pathname. This may cause the name to exceed 48 characters. To avoid this situation, use the volume name of the disk as the first component of the pathname.

CONTROL ACCESS NAME:

The access name of the control file. If you enter a pathname of a sequential file, CD uses the directives it finds there. If you enter ME, CD expects you to enter control file directives from your station. For details on control files, refer to the notes for BD. If you respond to both the INPUT PATHNAME and OUTPUT PATHNAME prompts, you do not need to specify a control file.

LISTING ACCESS NAME:

The device name or file name where you want to receive listings of the copy operation. If you skip this prompt, CD assumes by default that you want the listings sent to the terminal local file of your station. If the listing file reports an error in copying a file, the file is not included in the copy. If the listing file includes the following message next to the entry for a program file, it means that since the program file had no deletions, CD copied it all at once instead of a module at a time:

PROGRAM FILE COPIED USING FAST COPY

CD (COPY DIRECTORY)

OPTIONS:

List of options to control the copy operation. Each option is expressed as a pair of alternatives, for example ALIAS and NOALIAS. You can choose one or the other, but not both. If you do not choose either, CD uses a default. Options specified in the control file override any options chosen here. The options listed after ORIGINAL OPTIONS are the default options modified by the response to the OPTIONS prompt. Options specified by a control file are not included here.

Option	Purpose
ADD	(Default) If the last component of the file name of a file being copied matches the last component of the file name of a file in the output directory, do not replace the file in the output directory. In other words, leave the original contents of the output directory as is, but add any new files from the input directory.
REPLACE	If the file type and the last component of the file name of a file being copied match the file type and last component of the file name of a file in the output directory, replace the file in the output directory with a copy of the file in the input directory. If there is not a match, create a new entry in the output directory for a copy of the file in the input directory, giving it a file name that consists of the pathname of the directory followed by the last component of the file name of the file being copied. In other words, copy everything from the input directory into the output directory, deleting any files in the output directory whose file names conflict with the new additions.
ALIAS	(Default) Copy aliases.
NOALIAS	Do not copy aliases.
DATE	Copy only files updated later than a specified date. If you choose this option, CD displays additional prompts for the date and time. If the DATE option is used in a control file, additional date and time prompts must be answered. In batch or expert mode they can be answered directly. In interactive mode the DATE option must be included in the response to the OPTIONS prompt in order to answer the additional date and time prompts. If either the last update date or the creation date is after the date specified, the file is copied.
NODATE	(Default) Copy files regardless of their dates.
RPRL	Create relative record files with a specified physical record length. If you choose this option, CD displays an additional prompt for the physical record length. Using this option causes the last update date to be changed.
NORPRL	(Default) Create relative record files with the same physical record length as in the input directory.

CD (COPY DIRECTORY)

Option	Purpose
SPRL	Create sequential files with a specified physical record length. If you choose this option, CD displays an additional prompt for the physical record length. Using this option causes the last update date to be changed.
NOSPRL	(Default) Create sequential files with the same physical record length as in the input directory.
CMP	(Default) Create key indexed files without unused space at the end of the file.
NOCMP	Create key indexed files with the same allocation as in the input directory.
SYSFILES	(Default) Include these system files and directories in a copy operation on the system disk:

.SCI990	.S$SYSLIB
.S$CDT	.S$ISBTCH
.S$CMDS	.S$ISLIST
.S$EXPMSG	.S$LANG
.S$MSG	.S$PWCS
.SSGU	.S$SHARED
.S$SYSTEM	.S$SHIP
.S$IPL	.S$UTIL

If you are using a control file with INCLUDE directives, the SYSFILES option is ignored.

NOSYSFILE	Do not include the system files in the backup.

EXECUTION MODE(F,B):

Choice of whether to carry out the command in the foreground or background. If you enter F or FOREGROUND, the command executes in the foreground. If you enter B or BACKGROUND, the command executes in the background. Though the default is FOREGROUND, in a batch stream the CD command always executes in the background.

YEAR:

The two- or four-digit number for a year. This prompt appears only when you choose the DATE option. Your response becomes the year portion of the cutoff date used by the DATE option to determine which files to copy.

MONTH:

The one- or two-digit number of a month. This prompt appears only when you choose the DATE option. Your response becomes the month portion of the cutoff date used by the DATE option to determine which files to copy.

CD (COPY DIRECTORY)

DAY:
The one- or two-digit number of a day of the month. This prompt appears only when you choose the DATE option. Your response becomes the day portion of the cutoff date used by the DATE option to determine which files to copy.

HOUR:
The hour on the 24-hour clock. This prompt appears only when you choose the DATE option. Your response becomes the hour portion of the cutoff time used by the DATE option to determine which files to copy. If you skip this prompt, CD assumes 0 (the initial value).

MINUTE:
The minute of the hour. This prompt appears only when you choose the DATE option. Your response becomes the minute portion of the cutoff time used by the DATE option to determine which files to copy. If you skip this prompt, CD assumes 0 (the initial value).

PHYSICAL RECORD LENGTH:
The physical record length in bytes. This prompt appears only when you choose the RPRL or SPRL options.

Examples:

Copying a Directory Without a Control File

The following example copies the input directory VOL2.DIR1 to the output directory VOL2.DIR2. The input directory consists of the files FILE1, FILE2, FILE3 and PROGRAMS (a program file that contains the modules PROGA and PROGB). Since the prompt response calls for the ADD option and the output directory already has a file with FILE1 as the last component of its pathname, that file is not replaced.

```
[] CD

COPY DIRECTORY
          INPUT PATHNAME: VOL2.DIR1
         OUTPUT PATHNAME: VOL2.DIR2
    CONTROL ACCESS NAME:
    LISTING ACCESS NAME: LP
                OPTIONS: ADD
    EXECUTION MODE(F,B): FOREGROUND
```

CD produces the following listings:

```
 COPY DIRECTORY          14:19:14 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       VOL2.DIR1
ORIGINAL DESTINATION: VOL2.DIR2
ORIGINAL OPTIONS:      ADD,ALIAS,NODATE,NOSPRL,NORPRL,CMP,SYSFILES
CONTROL FILE:
LIST FILE:             LP


 **       DIRECTORY  VOL2.DIR2
 **   FILE2
 **   FILE3
 **   PROGRAMS - PROGRAM FILE
 **   PROGRAM FILE COPIED USING FAST COPY
 **   FILE1      NOT REPLACED

ELAPSED TIME = 0 MINUTES   14 SECONDS
SIZE OF INPUT = 349 ADU'S
1 WARNINGS ISSUED

****** COPY DIRECTORY COMPLETED
```

Copying a Directory Using a Control File

This example uses the control file VOL2.CONTROL to copy files from the input directories VOL2.DIR1 and VOL2.DIR3 to the output directory VOL2.DIR2. The control file specifies the REPLACE option for the copy operation on VOL2.DIR1 and excludes the files FILE1 and FILE2. The copy operation on VOL2.DIR3 includes the files FILEA and FILEB and uses the ADD option to override the prior REPLACE option in the control file.

```
OPTION REP
MOVE VOL2.DIR1
EX FILE1,FILE2
OPTION ADD
MOVE VOL2.DIR3
IN FILEA
IN FILEB
END

[] CD

COPY DIRECTORY
          INPUT PATHNAME:
         OUTPUT PATHNAME: VOL2.DIR2
    CONTROL ACCESS NAME: VOL2.CONTROL
    LISTING ACCESS NAME: LP
                OPTIONS: ADD
    EXECUTION MODE(F,B): FOREGROUND
```

CD (COPY DIRECTORY)

CD returns the following listings:

```
 COPY DIRECTORY          14:24:45 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:
ORIGINAL DESTINATION: VOL2.DIR2
ORIGINAL OPTIONS:     ADD,ALIAS,NODATE,NOSPRL,NORPRL,CMP,SYSFILES
CONTROL FILE:         VOL2.CONTROL
LIST FILE:            LP

OPTION REP
MOVE VOL2.DIR1
EX FILE1,FILE2

 **       DIRECTORY  VOL2.DIR2
 **  FILE3
 **  PROGRAMS - PROGRAM FILE
 **  PROGRAM FILE COPIED USING FAST COPY
OPTION ADD
MOVE VOL2.DIR3
IN FILEA
IN FILEB

 **       DIRECTORY  VOL2.DIR2
 **  FILEB
 **  FILEA
END

ELAPSED TIME = 0 MINUTES   19 SECONDS
SIZE OF INPUT = 457 ADU'S

****** COPY DIRECTORY COMPLETED
```

Copying a Single File

You can use CD to copy a single file by giving the file name in response to the INPUT PATHNAME prompt. This method allows you to copy files that Copy/Concatenate (CC) cannot handle, such as crash files. The following example copies the crash file .S$CRASH to the directory MAINTAIN.CRASHLOG:

```
[] CD

COPY DIRECTORY
            INPUT PATHNAME: .S$CRASH
           OUTPUT PATHNAME: MAINTAIN.CRASHLOG
      CONTROL ACCESS NAME:
      LISTING ACCESS NAME: LP
                  OPTIONS: NOSYSFILE
      EXECUTION MODE(F,B): FOREGROUND
```

The operation returns the following listings:

```
 COPY DIRECTORY          14:31:54 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       .VCATALOG
ORIGINAL DESTINATION: MAINTAIN.CRASHLOG
ORIGINAL OPTIONS:      ADD,ALIAS,NODATE,NOSPRL,NORPRL,CMP,NOSYSFILE
CONTROL FILE:
LIST FILE:             LP


 **       DIRECTORY  MAINTAIN.CRASHLOG
 **  S$CRASH

ELAPSED TIME = 0 MINUTES   41 SECONDS
SIZE OF INPUT = 430 ADU'S

****** COPY DIRECTORY COMPLETED
```

Messages:

TYPE/USAGE CONFLICT
: The CD command found files of different types but with the same name in the input and output directories. For example, the input directory has a relative record file with the same name as a sequential file in the output directory. Since the files have different structures, the copy process fails for those files. This error can also occur if the components of the directories are a file and a subdirectory with the same names. For a successful copy process, change the name of the input file or the output file. The REPLACE option does not correct this error.

Notes:

When CD is used in a file security system, you must have read access rights to all files which are to be copied. If you are replacing any existing files in the directory specified for OUTPUT PATHNAME, you must also have delete access rights to the files being replaced.

CD cannot access the spooler directory (.S$SDTQUE) on a running system disk until you halt output on each printer device and kill the spooler job. Then, you can copy this directory by using an INCLUDE directive from a control file.

Related Commands:

BD (Backup Directory)
RD (Restore Directory)
VC (Verify Copy)

CF (CREATE FILE)

4.28 CF (CREATE FILE)

Prompts:

```
CREATE FILE - SEQ, REL, KEY, DIR, PRO, IMG
  FILE TYPE: {SEQ/REL/KEY/DIR/PRO/IMG}          (SEQ)
```

Subsequent system prompts and user responses depend on the type of file specified. Entering the CF command specifying a file type has the same effect as entering the specialized create file command for the type of file specified. The following file type abbreviations correspond to the specialized create file commands:

CFDIR — Create Directory File
CFIMG — Create Image File
CFKEY — Create Key Indexed File
CFPRO — Create Program File
CFREL — Create Relative Record File
CFSEQ — Create Sequential File

Purpose:

The CF command creates any of the available file types.

Prompt Details:

FILE TYPE:
One of the following abbreviations:

Abbreviation	File Type
DIR	Directory file
IMG	Image file
KEY	Key indexed file
PRO	Program file
REL	Relative record file
SEQ	Sequential file

Example:

In the following example, the CF command creates a relative record file (REL):

```
[] CF

CREATE FILE - SEQ, REL, KEY, DIR, PRO, IMG
          FILE TYPE: REL

CREATE RELATIVE RECORD FILE
                    PATHNAME: VOL1.FILE2
     LOGICAL RECORD LENGTH: 80
   PHYSICAL RECORD LENGTH: >230
       INITIAL ALLOCATION: 500
     SECONDARY ALLOCATION: 1000
               EXPANDABLE?: YES
             FORCED WRITE?: YES
```

Special Cases:

You cannot use this command in batch mode. Use the create command for the specific file type.

Related Commands:

CFDIR (Create Directory File)
CFIMG (Create Image File)
CFKEY (Create Key Indexed File)
CFPRO (Create Program File)
CFREL (Create Relative Record File)
CFSEQ (Create Sequential File)

CFDIR (CREATE DIRECTORY FILE)

4.29 CFDIR (CREATE DIRECTORY FILE)

Prompts:

```
CREATE DIRECTORY FILE
                    PATHNAME:  [site:]pathname@        (*)
                 MAX ENTRIES:  integer
DEFAULT PHYSICAL RECORD SIZE:  [integer]
```

Purpose:

The CFDIR command creates special files, called directories, that contain pointers to other files and to subdirectories cataloged under the directory. You must create each directory and subdirectory in a hierarchy by a separate CFDIR command. For example, a file can be identified by the pathname VOL1.DIR.SUBDIR.FILE1. Create the directory and the subdirectory in that hierarchy separately by entering CFDIR commands that create VOL1.DIR and VOL1.DIR.SUBDIR.

Separate file creation commands create files under directories or subdirectories.

Prompt Details:

PATHNAME:
The pathname by which the directory is to be identified. You must include in the pathname all directories in the hierarchy under which the directory is cataloged (except VCATALOG). You can specify the pathname by a synonym, synonym followed by a pathname (synonym.pathname), logical name, or logical name followed by a pathname (logical name.pathname).

MAX ENTRIES:
An integer number specifying the maximum number of files and file support structures (channels, aliases, key descriptor records) to be kept in this directory. The number you enter is rounded up to the nearest prime number. The maximum number of entries that can be specified is 65,521. It is advisable to create the directory about 10 percent larger than actually needed.

DEFAULT PHYSICAL RECORD SIZE:
An even integer that specifies the physical record size to be used in all subsequent creation operations under this directory. If not specified, the value used is the same as that specified when the parent directory was created.

CFDIR (CREATE DIRECTORY FILE)

Example:

In the following example, the CFDIR command creates a directory with the pathname VOL1.SOURCE.MYFILE.TEST, which contains a maximum of 101 entries (supplied integer value of 100 rounded to the prime number of 101). The directories VOL1.SOURCE and VOL1.SOURCE.MYFILE were previously created.

```
[] CFDIR

CREATE DIRECTORY FILE
                      PATHNAME: VOL1.SOURCE.MYFILE.TEST
                   MAX ENTRIES: 100
DEFAULT PHYSICAL RECORD SIZE:
```

Assumptions:

Upper level directories specified in the pathname have previously been created.

Notes:

If you do not specify a volume in the pathname, the system disk is assumed.

The logical record length of a directory is always 256 bytes.

Related Commands:

CF	(Create File)
CFIMG	(Create Image File)
CFKEY	(Create Key Indexed File)
CFPRO	(Create Program File)
CFREL	(Create Relative Record File)
CFSEQ	(Create Sequential File)

CFIMG (CREATE IMAGE FILE)

4.30 CFIMG (CREATE IMAGE FILE)

Prompts:

```
CREATE IMAGE FILE
                    PATHNAME:  [site:]filename@
                  ALLOCATION:  integer
```

Purpose:

The CFIMG command creates an image file used for memory swapping, system loader files, and diagnostic dumps. No other use is suited for this file type; therefore, the use of image files is not recommended.

Prompt Details:

PATHNAME:
The pathname by which the image file is to be identified. Specify the file name by a pathname, a synonym, a synonym followed by a pathname (synonym.pathname), a logical name, or a logical name followed by a pathname (logical name.pathname).

ALLOCATION:
An even integer that specifies the size of the file in logical records. The logical record size is the sector size of the disk. An estimate of the number of records needed is as follows:

$$\frac{\text{length of object} + \text{sector size of disk}}{\text{sector size of disk}}$$

Example:

The following example creates an image file.

```
[] CFIMG

CREATE IMAGE FILE
            PATHNAME: VOL1.FILE3
          ALLOCATION: 1000
```

Related Commands:

CF	(Create File)
CFDIR	(Create Directory File)
CFKEY	(Create Key Indexed File)
CFPRO	(Create Program File)
CFREL	(Create Relative Record File)
CFSEQ	(Create Sequential File)

4.31 CFKEY (CREATE KEY INDEXED FILE)

Prompts:

```
CREATE KEY INDEXED FILE
                    PATHNAME:  [site:]filename@
       LOGICAL RECORD LENGTH:  integer
    PHYSICAL RECORD LENGTH:    [integer]
          INITIAL ALLOCATION:  [integer]
       SECONDARY ALLOCATION:   [integer]
               MAXIMUM SIZE:   integer
```

The following prompts are repeated for each key defined:

```
KEY DESCRIPTION FOR KEY NUMBER
          START POSITION:  integer          (*)
             KEY LENGTH:   integer
             DUPLICATES?:  YES/NO           (NO)
             MODIFIABLE?:  YES/NO           (NO)
       ANY MORE KEYS?:     YES/NO           (YES)
```

Purpose:

The CFKEY command creates a key indexed file. Each key indexed file uses two entries in its directory.

Prompt Details:

PATHNAME:
The pathname of the key indexed file to be created. Specify the filename by a pathname, a synonym, a synonym followed by a pathname (synonym.pathname), a logical name, or a logical name followed by a pathname (logical name.pathname).

LOGICAL RECORD LENGTH:
The length, in bytes, of the data records to be entered into the file.

PHYSICAL RECORD LENGTH:
The size of a block of data in the file. A block of data can contain multiple logical records. The block size should be an even multiple or factor of the ADU size of the disk on which the file is being created. The suggested block size is three times the sector size, in bytes, or an ADU, whichever is larger. However, if the file is going to contain either a large number of data records (100,000 or more), one or more very large keys, or data records that are close to or more than three times the sector size, it is advisable to use nine times the sector size, in bytes, as the physical record size. The physical record length must be at least 22 bytes larger than the logical record length. The default size is the directory size in which this file was created.

INITIAL ALLOCATION:
The amount of disk space, in logical records, that should be reserved for the file when it is created. The default value is system defined and is one ADU or the specified physical record length. The default value for this prompt can be specified by entering a null value.

CFKEY (CREATE KEY INDEXED FILE)

SECONDARY ALLOCATION:
The amount of disk space, in logical records, that should be allocated if the file grows larger than the initial allocation. The default value is system defined and is one ADU or the specified physical record length.

MAXIMUM SIZE:
The estimated adjusted number of logical records to be placed in the file. That is, the number of logical records times the percent of blanks in each logical record. For example, if there are to be 10,000 logical records in the file and about 50 percent of each record is made up of blanks, the adjusted number of logical records is 5000.

START POSITION:
The location within the logical record where the key begins. The first character of the record is position 1.

KEY LENGTH:
The number of characters in the key. The range of legal values is 1 through 100.

DUPLICATES?:
The value of a key in a record can be unique throughout the file, or there can be many records containing the same key value. These same values are called duplicates. If the key does not have to be unique, the response to this prompt should be YES. If the key value must be unique, such as a key for a social security number, the response to this prompt should be NO. This feature can aid in preventing erroneous data from being entered into the file and in guaranteeing unique key values for every record in the file.

MODIFIABLE?:
The value of a key can be changed if it is defined to be modifiable. All keys, except the first key defined, can be modifiable. Keys such as an employee's salary and marital status are examples of modifiable keys. A YES response makes the key modifiable, and a NO response makes the key nonmodifiable.

ANY MORE KEYS?:
If there are more keys to be defined, the response to this prompt must be YES; otherwise, it must be NO. After you enter a NO response, the key indexed file is created.

CFKEY (CREATE KEY INDEXED FILE)

Example:

In the following example, the CFKEY command creates a key indexed file with three keys:

```
[] CFKEY

CREATE KEY INDEXED FILE
                    PATHNAME: VOL1.FILE1
     LOGICAL RECORD LENGTH: 80
    PHYSICAL RECORD LENGTH: 560
        INITIAL ALLOCATION: 1200
      SECONDARY ALLOCATION: 500
               MAXIMUM SIZE: 100

KEY DESCRIPTION FOR KEY NUMBER 1
             START POSITION: 1
                 KEY LENGTH: 2
               DUPLICATES?: NO
               MODIFIABLE?: NO
             ANY MORE KEYS?: YES

KEY DESCRIPTION FOR KEY NUMBER 2
             START POSITION: 3
                 KEY LENGTH: 4
               DUPLICATES?: NO
               MODIFIABLE?: YES
             ANY MORE KEYS?: YES

KEY DESCRIPTION FOR KEY NUMBER 3
             START POSITION: 7
                 KEY LENGTH: 8
               DUPLICATES?: YES
               MODIFIABLE?: NO
             ANY MORE KEYS?: NO
```

CFKEY (CREATE KEY INDEXED FILE)

To create a key indexed file in batch mode, the batch stream must include a CFKEY command followed by 1 or more (up to a maximum of 14) KEY commands and an ENDKEY command. The following example creates a key indexed file in a batch stream.

Command	Comment
CFKEY parameters	Create key indexed file
KEY parameters	Specify from 1 to 14 keys
*	*
*	*
*	*
KEY parameters	
ENDKEY	Terminate batch mode CFKEY.

The batch parameters for the CFKEY command in batch mode include:

PATHNAME
LOGICAL RECORD LENGTH
PHYSICAL RECORD LENGTH
INITIAL ALLOCATION
SECONDARY ALLOCATION
MAXIMUM SIZE

The batch parameters for each KEY command are:

START POSITION
KEY LENGTH
DUPLICATES?
MODIFIABLE?

The ENDKEY command does not have any parameters. The indicated order of these commands cannot be varied without producing errors. The inputs for parameters must correspond to the definitions given earlier for CFKEY parameters in interactive mode.

To include CFKEY in another command procedure, you should specify the CFKEY, KEY. . . . , ENDKEY command sequence as in batch mode. However, on the CFKEY command, the following field prompt assignment should be included:

INTERACTIVE? = NO

This field prompt assignment prevents the CFKEY parameters from being prompted interactively.

CFKEY (CREATE KEY INDEXED FILE)

Notes:

The value entered for the initial allocation prompt is compared against the results of a calculation involving the logical record length, physical record length, and maximum size. The larger of the two is used by the CFKEY processor as the initial allocation. If the default is taken for this prompt, the results of the calculation are used. These estimates are not crucial; however, if the creation allocation is too small, the file may become fragmented if it is on a disk containing many other expanding files.

If you abort the CFKEY command while in the process of defining a key indexed file, the next attempt to use the CFKEY command will show the following error message: ERROR: INVALID KEY SEQUENCE; CFKEY BEFORE ENDKEY. Enter the ENDKEY command to clear the error condition caused by the abort before entering your first CFKEY command following an abort.

Related Commands:

CF	(Create File)
CFDIR	(Create Directory File)
CFIMG	(Create Image File)
CFPRO	(Create Program File)
CFREL	(Create Relative Record File)
CFSEQ	(Create Sequential File)
CKS	(Copy KIF to Sequential File)
CSK	(Copy Sequential File to KIF)
MKF	(Map Key Indexed File)
MKL	(Modify KIF Logging)

CFPRO (CREATE PROGRAM FILE)

4.32 CFPRO (CREATE PROGRAM FILE)

Prompts:

```
CREATE PROGRAM FILE
                    PATHNAME: [site:]filename@
          MAX NUMBER OF TASKS: integer              (255)
MAX NUMBER OF PROCEDURES: integer                   (255)
   MAX NUMBER OF OVERLAYS: integer                  (255)
          INITIAL ALLOCATION: integer               (85)
        SECONDARY ALLOCATION: [integer]
                  EXPANDABLE?: YES/NO               (YES)
```

Purpose:

The CFPRO command creates a program file. A program file is a special type of relative record file used to contain executable program components in memory image form.

Prompt Details:

PATHNAME:
The pathname by which the program file is to be identified. Specify the pathname by a synonym, synonym followed by a pathname (synonym.pathname), logical name, or logical name followed by a pathname (logical name.pathname).

MAX NUMBER OF TASKS:
The maximum number of tasks allowed in this program file. A response of zero allows for 255 tasks.

MAX NUMBER OF PROCEDURES:
The maximum number of procedures and program segments allowed in this program file. A response of zero allows for 255 procedures and program segments.

MAX NUMBER OF OVERLAYS:
The maximum number of overlays allowed in this program file. A response of zero allows for 255 overlays.

INITIAL ALLOCATION:
An integer that specifies the initial file size in sectors (1 physical record = 1 sector). The initial value is 85 sectors, which is the minimum size for overhead information when the defaults are taken for the MAX NUMBER OF TASKS, MAX NUMBER OF PROCEDURES, and MAX NUMBER OF OVERLAYS prompts.

SECONDARY ALLOCATION:
An integer that specifies the amount of disk space (in sectors) to be allocated when the file must grow. If you specify a null response, the system provides a default value of one.

CFPRO (CREATE PROGRAM FILE)

EXPANDABLE?:
If you enter YES, the file can grow. If NO, it cannot grow. Program files are usually created expandable.

Example:

In the following example, the CFPRO creates a program file:

```
[] CFPRO

CREATE PROGRAM FILE
                    PATHNAME: VOL1.DIR2.FILE2
         MAX NUMBER OF TASKS: 255
   MAX NUMBER OF PROCEDURES: 255
     MAX NUMBER OF OVERLAYS: 255
          INITIAL ALLOCATION: 85
        SECONDARY ALLOCATION:
                  EXPANDABLE?: YES
```

Related Commands:

CF	(Create File)
CFDIR	(Create Directory File)
CFIMG	(Create Image File)
CFKEY	(Create Key Indexed File)
CFREL	(Create Relative Record File)
CFSEQ	(Create Sequential File)

CFREL (CREATE RELATIVE RECORD FILE)

4.33 CFREL (CREATE RELATIVE RECORD FILE)

Prompts:

```
CREATE RELATIVE RECORD FILE
                    PATHNAME:  [site:]filename@
          LOGICAL RECORD LENGTH:  [integer]
        PHYSICAL RECORD LENGTH:  [integer]
              INITIAL ALLOCATION:  [integer]
          SECONDARY ALLOCATION:  [integer]
                   EXPANDABLE?:  YES/NO          (YES)
                  FORCED WRITE?:  YES/NO         (NO)
```

Purpose:

The CFREL command creates a relative record file.

Prompt Details:

PATHNAME:
The pathname by which the relative record file is to be identified. Specify the pathname by a synonym, synonym followed by a pathname (synonym.pathname), logical name, or logical name followed by a pathname (logical name.pathname).

LOGICAL RECORD LENGTH:
The length of a data record for this file. The specified length must be an even integer. The default value is 80, which is specified by entering a null value or a zero.

PHYSICAL RECORD LENGTH:
The size of a block of data in the file. A block can contain more than one logical record. The default value is the defined record size of the directory in which this file is created. The specified length must be an even integer.

INITIAL ALLOCATION:
The initial size, in logical records, of this file. The system provides a default value of one ADU or the specified physical record length rounded to the nearest ADU.

SECONDARY ALLOCATION:
If the file grows beyond the initial size, this number of logical records specifies the growth segment size. The system provides a default value of one ADU or the specified physical record length rounded up to the nearest ADU.

EXPANDABLE?:
If you enter YES, the file can grow beyond its initial size. If NO, the file cannot grow.

FORCED WRITE?:
Disk writes to relative record files are ordinarily deferred until memory is needed. YES specifies that logical records are to be written immediately to disk. NO specifies that the CFREL command is to defer writes.

CFREL (CREATE RELATIVE RECORD FILE)

Example:

In the following example, the CFREL creates a relative record file:

```
[] CFREL

CREATE RELATIVE RECORD FILE
                    PATHNAME: VOL1.FILE2
     LOGICAL RECORD LENGTH: 288
   PHYSICAL RECORD LENGTH: 864
       INITIAL ALLOCATION: 500
     SECONDARY ALLOCATION:
               EXPANDABLE?: YES
             FORCED WRITE?: YES
```

Related Commands:

CF	(Create File)
CFDIR	(Create Directory File)
CFIMG	(Create Image File)
CFKEY	(Create Key Indexed File)
CFPRO	(Create Program File)
CFSEQ	(Create Sequential File)

CFSEQ (CREATE SEQUENTIAL FILE)

4.34 CFSEQ (CREATE SEQUENTIAL FILE)

Prompts:

```
CREATE SEQUENTIAL FILE
                    PATHNAME:  [site:]filename@
         LOGICAL RECORD LENGTH:  [integer]
        PHYSICAL RECORD LENGTH:  [integer]
            INITIAL ALLOCATION:  [integer]
         SECONDARY ALLOCATION:  [integer]
                  EXPANDABLE?:  YES/NO          (YES)
              BLANK SUPPRESS?:  YES/NO          (NO)
                 FORCED WRITE?:  YES/NO         (NO)
```

Purpose:

The CFSEQ command creates a sequential file.

Prompt Details:

PATHNAME:
The pathname by which the sequential file is to be identified. Specify the pathname by a synonym, synonym followed by a pathname (synonym.pathname), logical name, or logical name followed by a pathname (logical name.pathname).

LOGICAL RECORD LENGTH:
An even integer that specifies the average record size, in bytes, for this file. If records are to be blank-suppressed, do not include blank strings in the size estimate. The default value is 80, which is specified by entering a null value or zero.

PHYSICAL RECORD LENGTH:
An even integer that specifies the size of a block of data in the file. A block usually contains more than one logical record. The default value is the defined physical record size of the directory in which the file is created.

INITIAL ALLOCATION:
An integer that specifies the initial file size in logical records. The system provides a default value of one ADU or the specified physical record length rounded up to the nearest ADU.

SECONDARY ALLOCATION:
An integer that specifies the increment of additional disk space in logical records, to be allocated when the existing space is full. The system provides a default value of one ADU or the specified physical record length rounded up to the nearest ADU.

EXPANDABLE?:
If you enter YES, the file can grow beyond its initial size. If NO, the file cannot grow.

CFSEQ (CREATE SEQUENTIAL FILE)

BLANK SUPPRESS?:
If you enter YES, the file is to be stored with strings of blanks encoded in compact form. If you enter NO, blank strings are not to be encoded.

FORCED WRITE?:
Disk writes to sequential files are normally deferred until memory is needed. YES specifies that the CFSEQ command is to force immediate writes. NO specifies that CFSEQ is to do deferred disk writes.

Example:

In the following example, the CFSEQ command creates a sequential file of 80-character logical records:

```
[] CFSEQ

CREATE SEQUENTIAL FILE
                    PATHNAME: VOL1.FILE1
     LOGICAL RECORD LENGTH:
    PHYSICAL RECORD LENGTH:
        INITIAL ALLOCATION: 100
      SECONDARY ALLOCATION:
               EXPANDABLE?: YES
          BLANK SUPPRESS?: NO
            FORCED WRITE?: NO
```

Related Commands:

CF	(Create File)
CFDIR	(Create Directory File)
CFIMG	(Create Image File)
CFKEY	(Create Key Indexed File)
CFPRO	(Create Program File)
CFREL	(Create Relative Record File)

CIC (CREATE IPC CHANNEL)

4.35 CIC (CREATE IPC CHANNEL)

Prompts:

```
CREATE IPC CHANNEL
          CHANNEL PATHNAME: [site:]pathname@
OWNER TASK PROGRAM FILE: [site:]pathname@
  OWNER TASK NAME OR ID: alphanumeric/integer
           CHANNEL TYPE: {SYM/MS}              (SYMMETRIC)
          CHANNEL SCOPE: {GLOBAL/JOB/TASK}     (GLOBAL)
 MAXIMUM MESSAGE LENGTH: integer               (100)
  SHARED CHANNEL ACCESS?: YES/NO               (YES)
```

The following prompts are displayed if MS (master/slave) is specified as channel type:

```
MASTER/SLAVE CHANNEL ATTRIBUTES
          RESOURCE TYPE: alphanumeric          (CHAN)
       PROCESS ASSIGNS?: YES/NO                (NO)
     PROCESS ABORT I/O?: YES/NO                (NO)
PROCESS I/O UTILITY SVCS?: YES/NO              (NO)
```

Purpose:

The CIC command creates a global, job-local, or task-local channel accessible by any tasks that use the channel name.

A global channel is nonreplicatable and is accessible by any task in the system. A job-local channel is replicatable once for each job and is accessible by any task in the job. A task-local channel is replicated once for each task in any job that uses the channel.

Two types of channels can be created: master/slave and symmetric. A master/slave channel is one in which the owner (master) interprets and/or executes messages and/or commands transmitted by requesters (slaves) on the channel. A symmetric channel is one in which the owner and requester(s) issue read and write commands that match each other. The read command of one task is processed as soon as the other task issues a write command and vice versa.

CIC (CREATE IPC CHANNEL)

Prompt Details:

CHANNEL PATHNAME:
A valid pathname by which the channel is to be known. The pathname must be identical to the owner task program file pathname except for the last component.

OWNER TASK PROGRAM FILE:
The pathname of the program file on which the owner task is installed.

OWNER TASK NAME OR ID:
The installed name or the ID of the owner task on the program file. If you specify a task name, you must install a task with that name on the program file.

CHANNEL TYPE:
The type of channel to create:

SYM — Symmetric
MS — Master/Slave

CHANNEL SCOPE:
The scope of the channel:

GLOBAL — Global
JOB — Job-local
TASK — Task-local

MAXIMUM MESSAGE LENGTH:
The maximum message length, in bytes, of a message passed on this channel. The maximum message length is decimal 12,288 (hexadecimal 3000) bytes.

SHARED CHANNEL ACCESS?:
A YES response permits two or more tasks to share the channel. A NO response permits only two tasks to use the channel.

RESOURCE TYPE:
The device or file type for which the owner task makes requests. A master/slave channel owner task can process any type of request from the requester, as long as that request is transmitted correctly. To ensure the channel is efficiently used, you must create it with a device type appropriate for the types of requests to be made. For example, if write operations with the reply option are to be used, create the channel with a device type of VDT. If channel requesters use only simple read and write requests, create the channel as device type of CHAN.

CIC (CREATE IPC CHANNEL)

The following device and file types may be specified:

Devices:

ASR	—	Automatic send/receive terminal
CHAN	—	Channel
CR	—	Card reader
CS	—	Cassette tape
DK	—	Diskette drive
DS	—	Disk drive
IN	—	Industrial device
KSR	—	Keyboard send/receive terminal
LP	—	Line printer
MT	—	Magnetic tape
SD	—	Special device
VDT	—	Video display terminal

Files:

DIR	—	Directory file
IMG	—	Image file
KIF	—	Key indexed file
PRO	—	Program file
REL	—	Relative record file
SEQ	—	Sequential file

PROCESS ASSIGNS?:
A YES response permits Assign LUNO and Release LUNO requests for this channel to be processed by the owner task. A NO response prohibits these requests from being processed by the owner task.

PROCESS ABORT I/O?:
A YES response allows Abort I/O SVCs (operation code >0F) for this channel to be processed by the owner task. A NO response prohibits these requests from being processed by the owner task.

PROCESS I/O UTILITY SVCS?:
A YES response permits all I/O utility SVCs (I/O suboperation codes >90 and greater) for this channel to be processed by the owner task. A NO response prohibits these requests from being processed by the owner task.

CIC (CREATE IPC CHANNEL)

Example:

In the following example, assume a user's job contains several tasks where separate computations are to be output as a final report. The user can create a master/slave, task-local channel to be used by each of these tasks, using the CIC command as follows:

```
[] CIC

CREATE IPC CHANNEL
          CHANNEL PATHNAME: .KC0017.REPORT
  OWNER TASK PROGRAM FILE: .KC0017.COMPILE
    OWNER TASK NAME OR ID: TAXES
              CHANNEL TYPE: MS
             CHANNEL SCOPE: TASK
   MAXIMUM MESSAGE LENGTH: 100
   SHARED CHANNEL ACCESS?: YES

 MASTER/SLAVE CHANNEL ATTRIBUTES
              RESOURCE TYPE: CHAN
          PROCESS ASSIGNS?: YES
        PROCESS ABORT I/O?: NO
 PROCESS I/O UTILITY SVCS?: NO
```

Assumptions:

The owner task exists on the program file specified before the channel is used. The task can be installed after the CIC command is processed, since the CIC command does not verify existence of the owner task.

Related Commands:

DIC	(Delete IPC Channel)
SCS	(Show Channel Status)

CKD (CHECK DISK FOR CONSISTENCY)

4.36 CKD (CHECK DISK FOR CONSISTENCY)

Prompts:

```
CHECK DISK FOR CONSISTENCY
                    DISK UNIT:  devicename
        LISTING ACCESS NAME:  [site:][pathname]@
```

Purpose:

The CKD command displays the following:

1. The date, time, and disk type
2. All directories on the specified disk, with an analysis of the entry counts for each directory
3. The number of ADUs dually allocated
4. A dump of the bit maps for the disk, if any dual allocations have occurred

Additional information concerning the CKD command is discussed following the CKD examples and sample output.

Prompt Details:

DISK UNIT:
The device name of the disk unit to be checked (for example, DS01, DS02, and so on).

LISTING ACCESS NAME:
The access name to which output is to be printed; this must not be a file on the disk being checked. Specify the pathname as a synonym, synonym followed by a pathname (synonym.pathname), logical name, or logical name followed by a pathname (logical name.pathname).

Examples:

The following examples show the output produced by the CKD command. In the first example, the disk being checked is contained in disk DS03 and has no multiply-allocated ADUs. In the second example, the disk being checked is contained in disk device DS03 and does have multiply-allocated ADUs.

CKD (CHECK DISK FOR CONSISTENCY)

In both examples, the response to the LISTING ACCESS NAME prompt is a file.

The disks will be checked for consistency by responding to the CKD command prompts as follows:

```
[] CKD

CHECK FOR CONSISTENCY
              DISK UNIT: DS03
    LISTING ACCESS NAME: VOL1.CKDLIST

CKD DNOS    DISK TYPE:DS31       10:32:22 TUESDAY, JUNE 14, 1983.
PASS # 1
SIZE  NFL   NAR   FILES AVAIL DIRECTORY
0035  0001  0034  0001  0034  LINK5.TEMPLATE.DECLARE
0017  0001  0016  0001  0016  LINK5.TEMPLATE
0065  0003  0062  0003  0062  LINK5.BDD.OBJECT
0065  0001  0064  0001  0064  LINK5.BDD
0065  0003  0062  0003  0062  LINK5.CV.OBJECT
0065  0001  0064  0001  0064  LINK5.CV
000B  0005  0006  0005  0006  LINK5
0000 MULTIPLY ALLOCATED ADUS

[] CKD

CHECK DISK FOR CONSISTENCY
              DISK UNIT: DS03
     LISTING ACCESS NAME: VOL1.CKDLIST

CKD DNOS    DISK TYPE:DS31       10:39:32 TUESDAY, JUNE 14, 1983.
PASS # 1
SIZE  NFL   NAR   FILES AVAIL DIRECTORY
0035  0001  0034  0001  0034  LINK5.TEMPLATE.DECLARE
0017  0001  0016  0001  0016  LINK5.TEMPLATE
0065  0003  0062  0003  0062  LINK5.BDD.OBJECT
0065  0001  0064  0001  0064  LINK5.BDD
0065  0003  0062  0003  0062  LINK5.CV.OBJECT
0065  0001  0064  0001  0064  LINK5.CV
000B  0005  0006  0005  0006  LINK5
PASS # 2
DUAL ALLOCATION FILE 0006 LINK5.BDD.OBJECT.BDD
0001 0009 4244 4420 2020 2020 0000 0000
1A00 0100 0050 000B 1121 0003 0000 0000
0000 0021 0000 000A 0092 0000 0000 0000
0000 0000 0000 07BF 00AA 9B5D 07BF 00AE
C358 0101 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
```

CKD (CHECK DISK FOR CONSISTENCY)

```
DUAL AND UNALLOCATED 0007 LINK5.BDD.OBJECT.BDMESG
0001 0040 4244 4D45 5347 2020 0000 0000
1A00 0100 0050 000B 1127 0003 0000 0000
0000 0020 0000 000A 006A 0000 0000 0000
0000 0000 0000 07BF 00AA AAD6 07BF 00AE
C3A3 0101 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0005 MULTIPLY ALLOCATED ADUS
ADUS 1127 THRU 112B DUALLY ALLOCATED
0007 (1127 - 112B)  0006 (1127 - 112B)
PARTIAL BIT MAP 0000
01A4 FFFF FFFF F003 FFFF FFFF FFFF FFFF
FF00 0000 0000 0000 0000 FFFF FFFF FFFF
FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
FFFF FFFF FFFF F000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0003 FFFF FFFF FFFF F000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
PARTIAL BIT MAP 0001
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 003F FFF0 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
```

CKD (CHECK DISK FOR CONSISTENCY)

```
PARTIAL BIT MAP 0002
05B5 0000 0000 FF00 0000 0000 0000 0000
0000 0000 0000 0000 1FFF FFFF FFFF FFFF
FFFF FFFF FF80 0000 0000 7FF0 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0003 FF80 0000 0000 0380
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 1FFF FFFF FFFF FFFF FFFF FFFF
FF80 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0003 F800 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 003F 0000 0000 3800 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
ADUS 1000 THRU 1007  ALLOCATED BUT NOT USED
PARTIAL BIT MAP 0003
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
PARTIAL BIT MAP 0004
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 FFFF FFFF
FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
ADUS 112C THRU 1131  USED BUT NOT ALLOCATED
ADUS 120E THRU 1218  ALLOCATED BUT NOT USED
```

CKD (CHECK DISK FOR CONSISTENCY)

To execute the CKD command, the procedures are as follows:

1. Mount the volume to be checked, leaving it write protected. Write protection is a caution against disk allocation or deallocation on the disk during checking.

2. Install the volume, if not installed, using the Install Volume (IV) command.

3. Ensure that no disk allocation activity is occurring. This means that no compiler, assembler, or application activity can run that places files on the disk being checked. If the disk is the system disk, this prevents logging on or off and using any utility that produces output to a file (including the terminal local file), such as the Map Disk (MD), List Directory (LD), and Show Terminal Status (STS) commands. However, you can use these commands if their output goes to a file not on the disk being checked or to a teleprinter device. Violating the above precautions will cause inaccurate error messages to appear.

4. Execute the CKD command. Reply to the DISK UNIT prompt with the unit name where the disk to be checked is mounted. The CKD command will not execute if you enter the volume name. For the LISTING ACCESS NAME prompt, specify a device or file where the output is to be printed. Do not specify a file on the disk being checked since this causes inaccurate error messages to appear.

5. If you encounter problems, create a hard copy and contact your customer representative for assistance.

The first section of the CKD output is created during the first pass made on the disk. This section contains a list of all directories on the disk, with an analysis of the entry counts printed in five columns of numbers to the left of the directory pathname.

The columns are defined as follows:

Column	Definition
SIZE	The total number of entries the directory can hold.
NFL	The number of files currently in the directory as recorded in the directory overhead record. This value should match the value in the FILES column.
NAR	The number of available entries as recorded in the directory overhead record. This value should match the value in the AVAIL column.

CKD (CHECK DISK FOR CONSISTENCY)

FILES	The actual number of files currently in the directory determined by count. If this does not match the NFL column, the message ***** BAD DIRECTORY is printed above the line describing the bad directory.
AVAIL	The actual number of available entries in the directory determined by count. If this does not match the NAR column, the bad directory message is printed as for FILES.

NOTE

All numbers printed by this utility are hexadecimal.

The contents of these five fields do not directly affect any dual allocation problems that might exist. Errors indicate system crashes or power failures during critical sequences that update these counts on disk. However, bad directories can exist without causing problems to other parts of the disk that is being used.

The remaining sections of the output are produced by the second pass on the disk. The first of these is a dump of all File Descriptor Records (FDRs) claiming storage that is either not allocated in the bit maps, dually allocated, or both. The full pathnames of the files are printed, along with an index unique to the file on that disk. The index is used later for reference.

The next part of the output for the second pass is a single line, which specifies how many ADUs are dually allocated. If the number specified is zero, this concludes the information printed in the output listing file.

If the number of dually-allocated ADUs is greater than zero, the output file contains a list of all ranges of dually-allocated storage. Under each range is a list in the format, FFFF (AAAA-BBBB). FFFF is the file index, and AAAA-BBBB is the subrange of ADUs claimed by that particular file. Since a range of dually-allocated storage can be larger than the range of any particular file, it may not be true that all files listed under a given range are dually allocated over each other. Only those files in which individual ranges overlap are dually allocated.

Finally, a dump of the bit maps for the disk is produced. This is used to determine what problems are occurring with the bit maps. Interspersed in the dump can be lines of output stating the ranges of ADUs that are either used but not allocated or allocated but not used. Allocated but not used ADUs produce no problems; however, used but not allocated ADUs indicate that the disk structures have been damaged and need to be corrected.

Use the Copy Directory (CD) command to move data from damaged directories if possible; otherwise, contact your customer representative.

CKD (CHECK DISK FOR CONSISTENCY)

Notes:

When using CKD on a disk with secured files, the CKD should be issued by a member of the SYSMGR access group. Other secured files may not be accessible by CKD.

Related Commands:

RVI (Recover Volume Information)

4.37 CKR (COPY KIF RANDOMLY)

Prompts:

```
COPY KIF RANDOMLY
  KEY INDEXED FILE PATHNAME:   [site:]filename@
        OUTPUT FILE PATHNAME:  [site:]filename@
          LISTING ACCESS NAME: [site:][pathname]@
```

Purpose:

The CKR command copies the contents of a key indexed file to an output file. These files are not copied in a specific order. This command enables you to recover a key indexed file after system failure during partial logging. Use the CKR command also to recover the maximum amount of data possible from a key indexed file that has been partially destroyed or is unusable.

The CKR command reads the physical records of the key indexed file, checking the validity of the records in the data block. The logical records extracted from the file are written to an output file. If a data block has an inconsistent structure, an informative error message is written to the listing file. This message indicates the physical record number of the bad block and the offset into the block of the error.

The CKR command can be used to copy a key indexed file from a disk with one physical record length to a disk of another physical record length. Other command procedures, such as Copy Directory (CD), are unable to convert key indexed files of different physical record lengths.

Prompt Details:

KEY INDEX FILE PATHNAME:
: The pathname of the key indexed file to prepare for initial loading. Specify the pathname by a synonym followed by a pathname (synonym.pathname), logical name, or logical name followed by a pathname (logical name.pathname).

OUTPUT FILE PATHNAME:
: This pathname may be an existing key indexed file pathname or a sequential file pathname.

LISTING ACCESS NAME:
: The pathname of the device or file to which a description of the bad areas of the key indexed file are written. A null response specifies the terminal local file.

CKR (COPY KIF RANDOMLY)

Example:

The following example shows the execution of the CKR command for a key indexed file containing five physical records. The records could cause errors or system crashes if accessed by CKS or a user program.

```
[] CKR

COPY KIF RANDOMLY
  KEY INDEXED FILE PATHNAME: VOL1.KIF
      OUTPUT FILE PATHNAME: VOL1.SEQ
       LISTING ACCESS NAME: VOL1.LIST

1809              LOGICAL RECORDS COPIED
[]

[] SF

SHOW FILE
  FILE PATHNAME: VOL1.LIST

RECORD    OFFSET        MESSAGE
 00062     00D2     LOGICAL RECORD SIZE TOO LARGE
 00063     00F0     KEY ID TOO LARGE
 00064     0000     INCORRECT RECORD NUMBER
 00080     00F2     UNSUPPRESSION ERROR
 0022B     00EE     LOGICAL RECORD SIZE IS ODD

1809              LOGICAL RECORDS COPIED
[]
```

Messages:

INCORRECT RECORD NUMBER
: The first two words of the physical record are not the same as the physical record number.

LOGICAL RECORD SIZE IS ODD
: The blank-suppressed logical record size must be an even number.

LOGICAL RECORD SIZE TOO LARGE
: The logical record size is considered too large if it is more than 50 percent larger than the defined logical record size.

KEY ID TOO LARGE
: The key ID cannot be larger than the maximum key ID of the physical record.

UNSUPPRESSION ERROR
: While the record was being unsuppressed, it became too large.

CKR (COPY KIF RANDOMLY)

00XX READ ERROR
The specified error code (XX) was returned when the physical record was real.

XXXXXX LOGICAL RECORDS COPIED
The specified number (XXXXXX) of logical records was recovered from the key indexed file. The number is a decimal number.

Related Commands:

CKS	(Copy KIF to Sequential File)
CSK	(Copy Sequential File to KIF)

CKS (COPY KIF TO SEQUENTIAL FILE)

4.38 CKS (COPY KIF TO SEQUENTIAL FILE)

Prompts:

```
COPY KIF TO SEQUENTIAL FILE
                INPUT PATHNAME:  [site:]filename@
                           KEY:  integer
         BEGINNING KEY VALUE:  [character(s)]
            ENDING KEY VALUE:  [character(s)]
              ASCII OR BINARY:  {A/B}                (ASCII)
        OUTPUT ACCESS NAME:  [site:]pathname@
                    REPLACE?:  YES/NO                (NO)
```

Purpose:

The CKS command copies the contents of a key indexed file to a sequential file in a specified order. You specify the order by naming a key by which the file is read while being placed into the sequential output file. For example, if you specify the number 2 for the KEY prompt and that key contains zip codes, the key indexed file is read sequentially by zip code as it is copied to the output file.

Prompt Details:

INPUT PATHNAME:
A pathname that identifies the key indexed file to be copied.

KEY:
An integer number in the range of 1 through 14 that specifies the key number of the key by which the key indexed file is to be read as it is being copied to the sequential file.

BEGINNING KEY VALUE:
CKS starts copying records from the key indexed file record that has a key value greater than or equal to this value. The number of characters entered need not be the same as the size of the specified key. If you specify fewer characters, the rest of the key is filled with binary zeros. If you specify a null response, CKS begins copying with the lowest key value.

ENDING KEY VALUE:
CKS stops copying logical records from the key indexed file when the key value of the logical record is larger than this value. The number of characters entered need not be the same as the size of the specified key. If you specify fewer characters, the rest of the key is filled with binary zeros. If you specify a null response, CKS stops copying when there are no more records.

ASCII OR BINARY:
If you specify A, the given key values are processed as ASCII characters. If you specify B, the specified key values are processed as binary values. Therefore, the values must consist of only the characters zero through > F. If one of the specified keys contains both ASCII and binary characters, a B must be entered, and any ASCII characters must be specified as binary. For example, if the first byte of a key is binary > 2C and the second byte is a letter, such as C, the key value must be specified as > 2C43.

CKS (COPY KIF TO SEQUENTIAL FILE)

OUTPUT ACCESS NAME:

The sequential file name to which the CKS command should write the logical records of the key indexed file. CKS creates a sequential file if the file named does not exist before entry of the command. Specify the pathname by a synonym, synonym followed by a pathname (synonym.pathname), logical name, or logical name followed by a pathname (logical name.pathname).

REPLACE?:

If you enter YES in response to this prompt, the file being copied is to replace the current contents of the output file if it already exists. An entry of NO specifies that the command is to be aborted if the output file already exists.

Example:

In the following example, the records of a key indexed file are read by key number 2 and copied to a sequential file through the CKS command, as shown below:

```
[] CKS

COPY KIF TO SEQUENTIAL FILE
          INPUT PATHNAME: VOL1.KIF
                     KEY: 2
    BEGINNING KEY VALUE: JOHNSON
       ENDING KEY VALUE: SMITH
        ASCII OR BINARY: A
     OUTPUT ACCESS NAME: VOL1.SEQKEY2
                REPLACE: YES
```

Special Cases:

There is a nondisplayed prompt called DISPLAY. If the response to DISPLAY is N (no) within a batch stream or a procedure that executes the CKS command, the message indicating the number of logical records copied by CKS is not displayed in the batch listing file.

Messages:

The number of records copied by CKS is displayed at the terminal when CKS terminates.

Notes:

The CKS command is not intended to be used to back up key indexed files. To back up a key indexed file, use the Copy Directory (CD) or the Backup Directory (BD) command, specifying the pathname of the file to be backed up as the INPUT PATHNAME parameter.

Related Commands:

CSK (Copy Sequential File to KIF)

CL (COPY LINES)

4.39 CL (COPY LINES)

After you activate the Text Editor, the CL command specifies which lines are to be copied in the current file to another position in the file. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the CL command and the following prompts:

```
COPY LINES
                    START LINE:  [BEGINNING/END/line number]
                      END LINE:  [BEGINNING/END/line number]
           INSERT AFTER LINE:    [BEGINNING/END/line number]
```

4.40 CM (CREATE MESSAGE)

Prompts:

```
CREATE MESSAGE
    RECEIVING TERMINAL NAME:   [site:][stationname]@     (*)
                      MESSAGE:  [character(s)]
```

Purpose:

The CM command sends a one-line message from a terminal or a batch stream to a terminal. A receiving terminal can only get a message created by the CM command if an interactive SCI is active at the destination terminal.

If no user is logged on at the terminal or if the current state of the terminal prevents receipt of the message, the CM command places the message in a queue until it can be transmitted. Use CM also in batch mode to send status messages to the initiating terminal.

Prompt Details:

RECEIVING TERMINAL NAME:
The station name of the terminal to which the system should send the message. A null response specifies your own terminal.

MESSAGE:
If in interactive mode, a one-line message from 1 to 50 characters long. If you use the CM command in a batch stream, the message can be a maximum of 200 characters in length.

Example:

In the following example, the CM command sends a message to the terminal named ST02:

```
[ ] CM

CREATE MESSAGE
  RECEIVING TERMINAL NAME: ST02
                  MESSAGE: LOG-OFF
```

The message displayed is of the following format:

I MAIL – 0001 RECEIVED HH:MM:SS [day],[month],[date], [year]. LOG – OFF

CM (CREATE MESSAGE)

Special Cases:

If no message text is supplied, the command is effectively ignored.

If you use the CM command in a batch stream executed by the Execute Batch Job (XBJ) command and specify the terminal default, the message is sent to the nonexistent station, ST00.

Notes:

You should avoid sending messages to nonexistent terminals since each message requires memory space until it can be sent.

Related Commands:

MSG (Display Message)

4.41 COM (CREATE OPERATOR MESSAGE)

Prompts:

```
CREATE OPERATOR MESSAGE
                    SITE NAME:  [site]
                     MESSAGE:  [character(s)]
```

Purpose:

The COM command sends a one-line message from a terminal to the system operator. The operator can receive a message created by the COM command only if an Execute Operator Interface (XOI) or a Receive Operator Messages (ROM) command is active at the system operator's terminal.

If the current state of the operator's terminal prevents receipt of the message, the Operator Interface places the message in a queue until it can be transmitted. Use the COM command in batch mode to send status messages to the operator.

Prompt Details:

SITE NAME:
A valid site name. Specify the site name of the computer site to which you want to send a message. A null response causes the message to be sent to the system operator at your site. Respond to this prompt only if your computer is part of a network.

MESSAGE:
A one-line message from 1 to 50 characters long.

Example:

In the following example, the COM command sends a message to the system operator:

```
[] COM

CREATE OPERATOR MESSAGE
         SITE NAME:
           MESSAGE: LOG-OFF FOR BACKUP
```

The message is displayed on the operator console when an XOI or ROM command has been issued by the operator.

In the following example, the computer in use is a part of a network. The COM command sends a message to the system operator of the computer whose site name is SOAPLAKE.

```
[] COM

CREATE OPERATOR MESSAGE
                SITE NAME: SOAPLAKE
                  MESSAGE: LOAD VOLUME VOL1
```

COM (CREATE OPERATOR MESSAGE)

Special Cases:

If you do not supply a message text, the command is effectively ignored.

Notes:

The site name of a computer is specified during system generation using the Execute System Generation Utility (XSGU) command, and can be modified by using the Modify Scheduler/ Swap Parameters (MSP) command on a disk image.

Related Commands:

CM	(Create Message)
KOM	(Kill Operator Messages)
KOR	(Kill Operator Interface Request)
LOM	(List Operator Messages)
QOI	(Quit Operator Interface)
ROM	(Receive Operator Messages)
ROR	(Respond to Operator Interface Request)
XOI	(Execute Operator Interface)

4.42 CP (CREATE PATCH)

Prompts:

CREATE PATCH

LINK MAP:	[site:]filename@	(*)
ASSEMBLER INPUT ACCESS NAME:	[site:]pathname@	(*)
PROGRAM FILE:	[site:]filename@	(*)
MODULE TO PATCH:	character(s)	(*)
OFFSET IN MODULE:	[integer]	
PATCH TYPE (IB/IA/RE):	{IB/IA/RE}	
VERIFICATION DATA:	[integer...integer]	
PATCH AREA NAME:	character(s)	(NFPAT)
PATCH NUMBER:	[integer]	
# WORDS TO PATCH:	[integer]	(2)

Purpose:

The CP command modifies a particular area of a program directly from a terminal without relinking the program. The specified PATCH AREA NAME must be an area in the program with the first word containing the byte length of the patch area and each remaining word containing the value > DEAD. The system patch area used for this purpose is the module named NFPAT. In order to use CP, you should also create a patch area module to be linked with application programs.

Prompt Details:

LINK MAP:
The pathname of the link map produced by the Execute Link Editor (XLE) command when the program was linked. The link map must have records of 80 characters or less.

ASSEMBLER INPUT ACCESS NAME:
The pathname where assembly language instructions, which are to be applied as patches, reside. Specify a file name or the terminal where the CP command is executed by entering ME in response to this prompt. Use the Enter key to terminate input from ME.

PROGRAM FILE:
The pathname of the program file on which the program to be patched resides.

MODULE TO PATCH:
The name of the module, linked into the program, to be patched.

OFFSET IN MODULE:
An integer value specifying the byte within the module from which to begin applying the patches. If no value is supplied, zero is used as the default.

CP (CREATE PATCH)

PATCH TYPE (IB/IA/RE):
The valid responses to this prompt have the following meaning:

IB Insert the patches in the module before the word specified in the OFFSET IN MODULE prompt.

IA Insert the patches in the module after the word specified in the OFFSET IN MODULE prompt.

RE Begin replacing the words in the module with the patches, beginning with the word specified in the OFFSET IN MODULE prompt.

VERIFICATION DATA:
The integer value(s) contained in the specified starting address of the OFFSET IN MODULE prompt. If you specify more than one integer, separate them by commas; it is assumed these values are contained in consecutive memory addresses, beginning with the specified starting address.

PATCH AREA NAME:
The name of the module linked with the program, which is used to contain the patches. The memory area of the module must contain a >DEAD value in each word. The first word of the patch area contains the number of bytes in the patch area. The system patch area used for patching purposes is named NFPAT.

PATCH NUMBER:
An optional integer value that is user-defined. This value allows you to keep track of patches applied.

WORDS TO PATCH:
An integer value which specifies the number of words to be patched. You must modify at least two words, but not more than five, for IB and IA patches. No limit is set for RE patches. The default value is two.

CP (CREATE PATCH)

Example:

In the following example, the CP command patches four words of a specified module on a user program file. The assembly language instructions are entered from the terminal where the CP command was executed, and the modifications begin at the specified word address entered in response to the OFFSET IN MODULE prompt.

```
[] CP

CREATE PATCH
                    LINK MAP: DS02.USER.MAP1
ASSEMBLER INPUT ACCESS NAME: ME
                PROGRAM FILE: .USER.PROGFIL
             MODULE TO PATCH: USERMOD
           OFFSET IN MODULE: >0A
       PATCH TYPE (IB,IA,RE): RE
           VERIFICATION DATA: >0000
             PATCH AREA NAME: USPAT
                PATCH NUMBER: >A14
            # WORDS TO PATCH: 4
```

The following instructions could be entered at the user's terminal, one line at a time, when the cursor is positioned at the bottom left-hand corner of the terminal screen:

```
            REF   USERMOD
            MOV   R10,R0
            JNE   X
            B     @USERMOD + >116
      X     EQU   $
            END
```

The entry of each instruction is terminated by pressing the Return key, and the next instruction can be entered when the cursor is positioned at the bottom left-hand corner of the terminal screen. When the END instruction is keyed, press the Enter key to signal CP that all instructions have been entered and the modifications can be performed. When the CP command completes execution, pressing the Command key reactivates SCI at the terminal.

CP (CREATE PATCH)

Notes:

There are two types of RE patches, those which replace fewer or an equal number of words in a program and those which replace a greater number of words. For instance, if a three-word instruction is to be replaced by a one-word instruction, the one-word instruction replaces the first word of the three-word instruction, with the remaining two effectively becoming NOPs. In this case, and in the case of an equal number of words being replaced, the patch area is not needed. However, when you use a greater number of words as patches than the amount of words being replaced, the first two words cause a branch to the specified patch area. The instructions being patched into the module are moved to the patch area. A branch then occurs back to the program, with execution resuming at the next instruction after the patches.

When a patch is to be inserted before (IB) or after (IA) the specified word in the program, the instructions deleted to allow room for the branch to the specified patch area are moved to the patch area.

An additional prompt, OPTIONAL, is not displayed in interactive mode, but you can specify it in interactive expert mode or in a batch stream. If you specify YES for this prompt, the CP command is prevented from producing an error message if the MODULE TO PATCH is not found on the specified link map. This capability allows batch streams to contain patches for modules that are optional in the link of the program being patched.

4.43 CPI (COPY PROGRAM IMAGE)

Prompts:

```
COPY PROGRAM IMAGE
                INPUT PROGRAM FILE:   [site:]filename@
               OUTPUT PROGRAM FILE:   [site:][filename]@
            INPUT IMAGE NAME OR ID:   {character(s)/integer}
OUTPUT IMAGE NAME AND/OR ID:          [character(s)/integer/character(s),integer]
              IMAGE TYPE (T/P/S/O):   {T/P/S/O}
                           OPTIONS:   {ADD/REPLACE}          (ADD)
```

If you enter T for IMAGE TYPE, the following prompts appear:

```
COPY PROGRAM IMAGE TASK EXTENSION
 COPY ASSOCIATED OVERLAYS?:   YES/NO                    (YES)
COPY ASSOCIATED PROCEDURES?: YES/NO                     (YES)
```

If you enter S for IMAGE TYPE, the following prompt appears:

```
COPY PROGRAM IMAGE SEGMENT EXTENSION
 COPY ASSOCIATED OVERLAYS?:   YES/NO                    (YES)
```

If you enter O for IMAGE TYPE, the following prompts appear:

```
COPY PROGRAM IMAGE OVERLAY EXTENSION
ASSOCIATED SEGMENT NAME/ID:   [character(s)/integer]
   ASSOCIATED SEGMENT TYPE:   {T/P}                     (TASK)
```

Purpose:

The CPI command copies the image of a task segment, procedure segment, program segment, or overlay from one program file to another. It also can copy an image to a different location within the same program file. If a task image is being copied, all procedure segments and/or overlays associated with the task can also be copied unless it is being copied to the same program file. You can define the name and/or ID of the image in the output program file to be different from those in the input program file. In the case when the image is being copied to a different location within the same program file, you must specify a unique name and ID. The names and IDs of the procedure segments and overlays being copied concurrently with their associated task must be the same. If it is desirable to change the names and/or IDs, copy the procedure segments and overlays separately.

If overlays of a task are copied at the same time as the task, the execution of the Map Program File (MPF) command to map the output file shows the overlays linked in reverse order to that shown if the input file is mapped through the MPF command. However, this should cause no problems. The MPF command is discussed later in this section.

CPI (COPY PROGRAM IMAGE)

Prompt Details:

INPUT PROGRAM FILE:
A pathname that identifies the program file containing the image to be copied.

OUTPUT PROGRAM FILE:
A pathname that identifies the program file into which the image is copied. If you wish to copy an image into the same program file as the input program file, you must omit a response to this prompt.

INPUT IMAGE NAME OR ID:
The name or the ID of the image to be copied.

OUTPUT IMAGE NAME AND/OR ID:
The name and/or ID under which the image is installed in the output program file. If left blank, the input name and ID are used. If you specify only one, the other is the same as on the input. If you specify both, separate them by a comma. If the copy is being done with the same file as the input and output program file, specify both an image name and an ID.

IMAGE TYPE (T/P/S/O):
If the image is a task segment, enter T; if it is a procedure segment, enter P; if it is a program segment, enter S; and if it is an overlay, enter O.

OPTIONS:
If the image already exists in the output program file and it should be replaced, enter REPLACE; otherwise, enter ADD. If you specify REPLACE, you must be using the same image name and ID for the input image as the one being replaced.

COPY ASSOCIATED OVERLAYS?:
If all overlays linked to the segment being copied should also be copied, enter YES; otherwise, enter NO.

COPY ASSOCIATED PROCEDURES?:
If all procedure segments associated with the task being copied should also be copied, enter YES; otherwise, enter NO. If you enter NO, the task is copied without attached procedures, but the flags and IDs are set to indicate the same set of attached procedures.

ASSOCIATED SEGMENT NAME/ID:
Enter the name or ID of the segment in the output program file to which the overlay should be linked. If linking is not desired, leave this field blank.

ASSOCIATED SEGMENT TYPE:
If the overlay is to be linked to a task segment, enter T. If the overlay is to be linked to a program segment, enter P.

CPI (COPY PROGRAM IMAGE)

Example:

The first CPI example copies a task segment and all its associated overlays and procedure segments from one program file to another and installs the task, by ID, on the output program file. The second example copies a specified overlay from one program file to another, installs the overlay by ID and name on the output program file, and links the overlay to a specified task. The third example makes available a second copy of a task in the same program file using the ID specified in the INPUT IMAGE NAME OR ID prompt. The CPI prompts and responses are shown below:

```
[] CPI

COPY PROGRAM IMAGE
          INPUT PROGRAM FILE: .S$PROGA
         OUTPUT PROGRAM FILE: .EXAMPLE
      INPUT IMAGE NAME OR ID: >33
OUTPUT IMAGE NAME AND/OR ID: >01
        IMAGE TYPE (T/P/S/O): T
                     OPTIONS: ADD

COPY PROGRAM IMAGE TASK EXTENSION
  COPY ASSOCIATED OVERLAYS?: YES
COPY ASSOCIATED PROCEDURES?: YES

[] CPI

COPY PROGRAM IMAGE
          INPUT PROGRAM FILE: .S$PROGA
         OUTPUT PROGRAM FILE: .EXAMPLE
      INPUT IMAGE NAME OR ID: >3
OUTPUT IMAGE NAME AND/OR ID: >7,NEW
        IMAGE TYPE (T/P/S/O): O
                     OPTIONS: ADD

COPY PROGRAM IMAGE OVERLAY EXTENSION
 ASSOCIATED SEGMENT NAME/ID: >1
    ASSOCIATED SEGMENT TYPE: TASK

[] CPI

COPY PROGRAM IMAGE
          INPUT PROGRAM FILE: .S$PROGA
         OUTPUT PROGRAM FILE:
      INPUT IMAGE NAME OR ID: >2
OUTPUT IMAGE NAME AND/OR ID: NEWTASK,>6
        IMAGE TYPE (T/P/S/O): T
                     OPTIONS: ADD

COPY PROGRAM IMAGE TASK EXTENSION
  COPY ASSOCIATED OVERLAYS?: YES
COPY ASSOCIATED PROCEDURES?: YES
```

CPI (COPY PROGRAM IMAGE)

Related Commands:

MPI	(Modify Program Image)
SPI	(Show Program Image)

CRV (CHECK AND RESET VOLUME)

4.44 CRV (CHECK AND RESET VOLUME)

Prompts:

```
CHECK AND RESET VOLUME
                    UNIT NAME:  devicename
```

Purpose:

The CRV command resets a media change condition on the disk drive. The system registers a media change condition if you turn off and on the disk drive power or if you replace the disk without a UV/IV command sequence execution. If you do not perform the CRV command, the disk can be read but any attempt to write to the disk results in an error. This error prevents you from creating dual allocation problems on a disk.

This command compares the volume information of the disk currently mounted in a specified drive with the volume information of the disk previously mounted in that drive during the last Install Volume (IV). If the volume information matches, the CRV command resets the software write-protect bit of the disk status word, thus resetting the media change condition. If the volume information does not match, the CRV command performs an Unload Volume (UV) on the installed disk name and resets the disk status word bit. A message informs you of the results of the CRV command if a reset occurs. Otherwise, the main menu appears. If necessary, you must then perform an IV on the currently mounted disk.

The media change condition is detected only for the DS80, DS300, CD1400, WD500, and WD800 disks.

Prompt Details:

UNIT NAME:
: The device name of the disk drive containing the volume to be checked for media change conditions.

Example:

The following example performs the CRV command on DS04.

```
[] CRV

CHECK AND RESET VOLUME
                UNIT NAME: DS04
```

CRV (CHECK AND RESET VOLUME)

Notes:

When executing the CRV command in expert mode or from a batch stream, you have the capability of specifying UNCONDITIONAL RESET. UNCONDITIONAL RESET indicates that the volume information has changed but the mounted disk is the same disk for which the previous IV was performed. This forces a reset of the media change status, disregarding the result of the volume information comparison. This feature is designed for use after a power failure on a disk for which a Modify Volume Information (MVI) has been performed. However, if the currently mounted disk is not the previously installed disk, the use of this feature can corrupt the data on the mounted disk.

Related Commands:

IV	(Install Volume)
UV	(Unload Volume)

4.45 CSF (CREATE SYSTEM FILES)

Prompts:

```
CREATE SYSTEM FILES
                VOLUME NAME:  [site:]alphanumeric     (*)
          MEMORY SIZE IN BYTES:  integer              (256K)
    DISK SECTOR SIZE IN BYTES:  {256/288}
```

Purpose:

The CSF command deletes and/or creates a crash file and roll file. CSF first attempts to delete (clear) the crash file and then the roll file; it then creates new files. CSF calls Delete File (DF) twice (once for each file to be deleted) and Create Image File (CFIMG) twice (once for each file created). The DF command is discussed later in this section.

Prompt Details:

VOLUME NAME:
: The disk or volume name of the disk where the system crash and roll files are to be created.

MEMORY SIZE IN BYTES:
: The memory size, specified in K bytes (K = 1024), available in the system. For example, enter 64 or 64K to specify a memory size of 65536 bytes. The K is optional.

DISK SECTOR SIZE IN BYTES:
: An integer value that is the number of bytes per sector. The response to this prompt must be either decimal 256 or 288.

Example:

In the following example, the CSF command creates the system roll and crash files:

```
[] CSF

CREATE SYSTEM FILES
                VOLUME NAME: REL30
          MEMORY SIZE IN BYTES: 256K
    DISK SECTOR SIZE IN BYTES: 256
```

CSF (CREATE SYSTEM FILES)

Notes:

If the amount of memory you enter is less than the amount of physical memory available, the crash file cannot contain all tasks that were executing at the time the crash dump was taken. This greatly reduces the usefulness of the crash dump. Execute the Show Memory Map (SMM) command to determine the amount of physical memory available.

The CSF command cannot be used to create system files on the system disk.

Related Commands:

CF (Create File)

CSK (COPY SEQUENTIAL FILE TO KIF)

4.46 CSK (COPY SEQUENTIAL FILE TO KIF)

Prompts:

```
COPY SEQUENTIAL FILE TO KIF
          INPUT ACCESS NAME:   [site:]pathname@
           OUTPUT PATHNAME:    [site:]filename@
```

Purpose:

The CSK command copies the contents of a sequential file to a previously defined key indexed file.

Prompt Details:

INPUT ACCESS NAME:
The pathname from which the CSK command is to copy sequential information to the specified key indexed file.

OUTPUT PATHNAME:
A file name identifying the key indexed file to which the sequential information is to be copied. You must define the key indexed file before entering the CSK command.

CSK (COPY SEQUENTIAL FILE TO KIF)

Example:

In the following example, a key indexed file is created and the contents of a sequential file are copied into the key indexed file through the CSK command as shown below:

```
[] CFKEY

CREATE KEY INDEXED FILE
                  PATHNAME: VOL1.KIF
     LOGICAL RECORD LENGTH: 80
   PHYSICAL RECORD LENGTH: 864
       INITIAL ALLOCATION:
     SECONDARY ALLOCATION:
             MAXIMUM SIZE: 125

     KEY DESCRIPTION FOR KEY NUMBER 1
           START POSITION: 1
               KEY LENGTH: 5
              DUPLICATES?: NO
              MODIFIABLE?: NO
           ANY MORE KEYS?: YES

     KEY DESCRIPTION FOR KEY NUMBER 2
           START POSITION: 8
               KEY LENGTH: 4
              DUPLICATES?: YES
              MODIFIABLE?: NO
           ANY MORE KEYS?: YES

     KEY DESCRIPTION FOR KEY NUMBER 3
           START POSITION: 14
               KEY LENGTH: 2
              DUPLICATES?: NO
              MODIFIABLE?: NO
           ANY MORE KEYS?: NO

[] CSK

COPY SEQUENTIAL FILE TO KIF
     INPUT ACCESS NAME: VOL1.KIFSRC
       OUTPUT PATHNAME: VOL1.KIF
```

Related Commands:

CKS (Copy KIF to Sequential File)

CSM (COPY/VERIFY SEQUENTIAL MEDIA UTILITY)

4.47 CSM (COPY/VERIFY SEQUENTIAL MEDIA UTILITY)

Prompts:

```
COPY/VERIFY SEQUENTIAL MEDIA UTILITY
          MASTER ACCESS NAME:  [site:]pathname@
            COPY ACCESS NAME:  [site:]pathname@
  MAXIMUM RECORD LENGTH:       integer                (8512)
COPY + VERIFY? (NO: = VERIFY): YES/NO
TERMINATE ON(EOM,2EOF,COUNT):  {integer/EOM/2EOF}
          LISTING ACCESS NAME: [site:]pathname@
```

Purpose:

The CSM command copies the contents of a sequential medium to another sequential medium. Allowable media for either source or destination are cassette, magnetic tape, and sequential disk file. Use this command also to verify that a copy operation has been successful.

Prompt Details:

MASTER ACCESS NAME:
The pathname of the sequential media from which the CSM command is to copy sequential information. This name must specify a sequential device (magnetic tape or cassette) or a sequential disk file.

COPY ACCESS NAME:
The pathname of the device or file to which the CSM command is to copy sequential information. If the destination is a file that does not yet exist, it is automatically created as a blank suppressed sequential file.

MAXIMUM RECORD LENGTH:
An integer representing the size in bytes of the largest record to be copied. The default is 8512 bytes. Due to memory limitations, the largest record that can be copied using this utility is 12,288 bytes.

COPY + VERIFY? (NO: = VERIFY):
If you enter YES in response to this prompt, both the copy and verify operations are performed. If you enter NO, only the verify operation is performed.

TERMINATE ON(EOM,2EOF,COUNT):
If you enter EOM, the sequential medium is copied to the end of medium. If you enter 2EOF, the copy continues until a double end of file is found. An integer number (count) can also be entered to copy a specific number of files. EOM is not supported for cassette tapes.

LISTING ACCESS NAME:
The pathname of the device or file to which the report on the copy or verify operation is written. A response to this prompt is required.

CSM (COPY/VERIFY SEQUENTIAL MEDIA UTILITY)

Example:

In the following example, the CSM command copies 20 files of information from a sequential file named .TEST1 to cassette unit CS02. The report of the copy operation is written to the spool device LP01.

```
[] CSM

COPY/VERIFY SEQUENTIAL MEDIA UTILITY
          MASTER ACCESS NAME: .TEST1
            COPY ACCESS NAME: CS02
       MAXIMUM RECORD LENGTH: 8512
  COPY+VERIFY? (NO :=VERIFY): YES
TERMINATE ON(EOM,2EOF,COUNT): 20
          LISTING ACCESS NAME: LP01
```

Related Commands:

CC (Copy/Concatenate)

4.48 CV (COPY VOLUME)

Prompts:

```
COPY VOLUME
                SOURCE DISK UNIT:  devicename        (*)
              SOURCE VOLUME NAME:  devicename
           DESTINATION DISK UNIT:  devicename        (*)
         DESTINATION VOLUME NAME:  devicename
                  LISTING DEVICE:  devicename@       (*)
                        VERIFY?:   YES/NO            (YES)
      CONVERT SEQUENTIAL FILES?:   YES/NO            (NO)
        CONVERT REL-REC FILES?:    YES/NO            (NO)
                   MORE COPIES?:   YES/NO            (NO)
```

Purpose:

The CV command copies and optionally verifies disk volumes used with DNOS. The copy is made from disk to disk regardless of disk type as long as the destination disk is large enough to hold all of the data being copied.

CV can tolerate imperfect media. That is, if the output disk has a bad track where the input disk has data, CV will copy the data onto a different track on the output disk rather than terminate the copy process. It does not support magnetic tape as a copy medium.

CV performs disk compression and limited file compression on all files except KIFs. Unused space within a nonexpandable file of any type is never compressed or recovered. Expandable files (except program files in certain cases) are fully compressed and the unused space is recovered to the disk. Expandable program files are compressed and unused space (made available by previously deleted tasks, procedures, or overlays) is recovered to the disk when copying between disks with different sector sizes. When copying between disks with identical sector sizes, only the unused space at the end of an expandable program file is recovered, unless you respond YES to the CONVERT REL-REC FILES? prompt. In that case an expandable program file is fully compressed and the unused space (made available by previously deleted tasks, procedures, or overlays) is recovered to the disk.

The CV utility unloads both the source and destination disks before it executes, and leaves them unloaded upon completion. Therefore, the disks cannot be accessed during execution of CV, and afterwards require the IV (Install Volume) command to be performed before they can be accessed.

This utility does not require the system disk to remain mounted, and therefore can copy a data disk in a two-disk system. If the system disk drive is involved in the copy process, CV requires that no other tasks or terminals be active when it executes.

CV (COPY VOLUME)

The proper procedure for using CV when the system disk is involved (either as master or copy disk) is as follows:

1. Perform an initial program load (IPL) and log on to the system.
2. Use the Execute Operator Interface (XOI) command to become the system operator.
3. If you are using a printer as the listing device, use the Modify Spooler Device (MSD) command to make the printer shared or not available to the spooler.
4. Use the List Jobs (LJ) command to determine jobs which are running.
5. Use the Kill Jobs (KJ) command to terminate all other jobs excluding the system job and your job under which the CV is to be performed.
6. Issue the Quit Operator Interface (QOI) command.
7. Execute CV.
8. When CV completes, perform an IPL and normal start-up procedures.

Prompt Details:

SOURCE DISK UNIT:
The name of the source disk unit (where the disk you want to copy is mounted) in the form DSxx, where xx is a disk unit number from 01 to 99.

SOURCE VOLUME NAME:
The volume name of the disk to be copied.

DESTINATION DISK UNIT:
The name of the destination disk unit in the form DSxx, where xx is a disk unit number from 01 to 99.

DESTINATION VOLUME NAME:
The volume name of the disk to which CV copies the disk identified by the source volume name.

LISTING DEVICE:
The device name of the terminal (STxx) or printer (LPxx) which is to be the listing device. If a listing is not required, enter DUMY. It is highly recommended that a hard copy listing be made for record keeping purposes.

VERIFY?:
Although it is not recommended, a copy can be made without verification if you enter NO in response to this prompt.

CONVERT SEQUENTIAL FILES?:
If the two disks involved in the copy process do not have the same default physical record lengths, physical record length conversion allows more efficient use of the destination disk. You can request physical record length conversion for sequential files by responding YES.

CONVERT REL — REC FILES?:
If the two disks involved in the copy process do not have the same default physical record lengths, physical record length conversion allows more efficient use of the destination disk. You can request physical record length conversion for relative record files by responding YES.

MORE COPIES?:
Enter YES if you wish to make a series of copies. Your terminal will then display the full set of prompts for the next copy to be made. The maximum number of copies you can produce with one CV command is nine.

Examples:

The following example describes a volume copy operation in which two copies are requested. For some of the prompts, the user has accepted the initial values.

In this example the disk named SYS63A, mounted in DS02, is being copied to a disk named SYS63B, mounted in DS01. A printer (LP01) is specified as the listing device, so a hard copy will be made of the results of the copy operation. This permanent listing can be useful if any problems occur. The copy is to be verified, and no record length conversion is to be performed.

Before execution begins on the first copy, the CV command prompts are displayed for the second copy. The responses for the SOURCE DISK UNIT, DESTINATION DISK UNIT, and LISTING DEVICE prompts are retained from the first copy. The user accepted them along with the other initial values, and specified new values for the remaining prompts.

```
COPY VOLUME
          SOURCE DISK UNIT: DS02
        SOURCE VOLUME NAME: SYS63A
     DESTINATION DISK UNIT: DS01
  DESTINATION VOLUME NAME: SYS63B
            LISTING DEVICE: LP01
                   VERIFY?: YES
CONVERT SEQUENTIAL FILES?: NO
   CONVERT REL-REC FILES?: NO
              MORE COPIES?: YES

COPY VOLUME
          SOURCE DISK UNIT: DS02
        SOURCE VOLUME NAME: SYS63A
     DESTINATION DISK UNIT: DS01
  DESTINATION VOLUME NAME: SYS63C
            LISTING DEVICE: LP01
                   VERIFY?: YES
CONVERT SEQUENTIAL FILES?: NO
   CONVERT REL-REC FILES?: NO
              MORE COPIES?: NO
```

CV (COPY VOLUME)

Figure 4-3 is an example of the VDT screen display upon normal termination of the first copy. Figure 4-4 is an example of the VDT screen display upon abnormal termination of the second copy. The numbered items in Figure 4-3 and Figure 4-4 are described in Table 4-7.

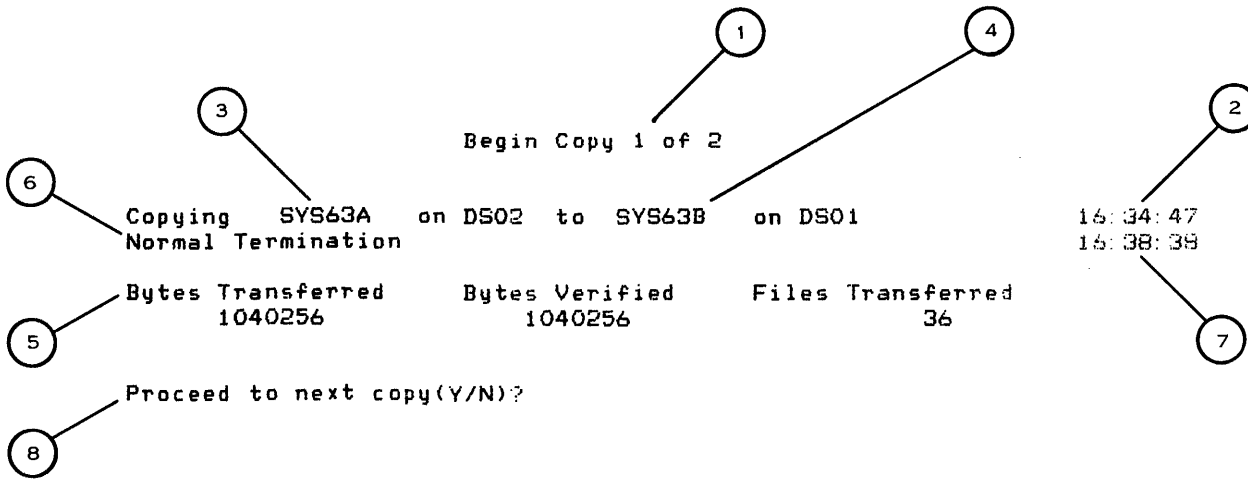


Figure 4-3. Screen Display Upon Normal CV Termination

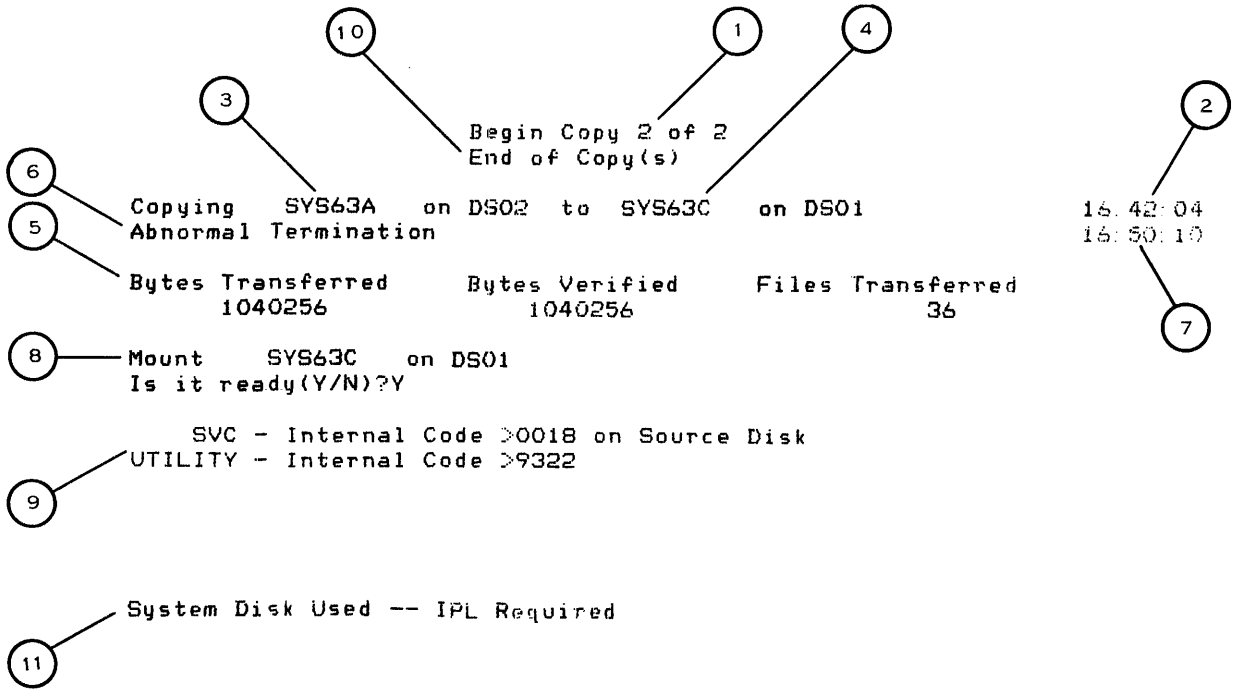


Figure 4-4. Screen Display Upon Abnormal CV Termination

Table 4-7. CV Screen Display

Field	Description
1.	The number of the copy in progress
2.	The time when the current copy began
3.	Volume name of the source disk, and the drive where it is mounted
4.	Volume name of the destination disk, and the drive where it is mounted
5.	Running tallies of the bytes copied, bytes verified, and files copied
6.	Completion message for the current copy; indicates whether termination was normal or abnormal
7.	The time when the current copy terminated
8.	Various messages requiring user response
9.	Error list (and bad file list if requested)
10.	Indicates termination of all requested copies
11.	If the system disk drive was used, this message appears after termination of the final copy.

Messages:

The following informative messages can appear on your terminal while executing CV.

Initializing Copy of < SOURCE VOLUME NAME> on < SOURCE DEVICE>
to < DESTINATION VOLUME NAME> on < DESTINATION DEVICE>

This message is displayed briefly after you respond to the final prompt of the CV command. It informs you that CV has started execution.

All other active Terminals, Jobs, and Tasks must be terminated.
Are you ready(Y/N)?

This message is displayed if either the source or destination disk is in the system disk drive, and any other terminals, jobs or tasks are active. Before continuing you must have all other terminals log off, and you must wait until any active jobs or tasks complete. If you respond Y, but other terminals or tasks are still active, the same message is displayed again. When all other tasks complete, all jobs terminate, and all other terminals log off, respond Y and the Copy Volume command begins execution. If you are not ready to continue with CV, enter N.

CV (COPY VOLUME)

Do you want to quit(Y/N)?

This message is displayed when you respond N to the *Are you ready(Y/N)?* prompt. If you respond Y, the Copy Volume command terminates. If you respond N, and other terminals, jobs, or tasks are still active, the *Are you ready(Y/N)?* prompt is displayed again.

Mount < VOLUME NAME> on DSxx
Is it ready(Y/N)?

This message is shown any time the system disk drive is involved in the copy operation. If the system drive is not involved, it is shown before the first copy operation if disks of the expected volume names are not mounted in the proper drives. It is also shown before every subsequent copy operation, even if all disks are already mounted in the proper drives. When the correct disks are mounted in the correct drives, enter Y, and CV begins producing the copy.

Summary of Progress Messages

During execution of CV from a VDT, your screen displays a summary of the progress of the copy operation. This includes the date and time at which the copy began, the volume names of the source and destination disks, and a series of messages. These messages keep you informed of the progress of the copy operation. Most require no action on your part unless attention to an error condition is required or the system disk needs to be replaced or removed.

Normal Termination HH:MM:SS

This message is displayed upon successful completion of each copy.

Proceed to next copy(Y/N)?

This message is displayed after each copy completion when you request a series of copies. If you wish to discontinue the series of copies, enter N. Otherwise enter Y, and CV continues with the next copy.

End of Copy(s)

This message indicates that the final copy operation is complete. Press the Return key to return to SCI.

System Disk Used — IPL Required

If the system disk drive was involved in the copy operation, this message informs you that an IPL is required to bring up DNOS.

Error Messages

During execution of CV, any errors that are encountered are listed on the lower half of your screen. The errors appear in one of the two following forms:

SVC – Internal Code > 0018 on Source Disk
UTILITY – Internal Code > 9321

The error code is a four-digit hexadecimal number representing the error returned from an SVC or from CV. Use the Show Expanded Message (SEM) command to find more information on these internal error codes or refer to the specified error number in the *DNOS Messages and Codes Reference Manual* which includes a description of the error and suggested user actions.

Do you want a bad file summary(Y/N)?
: After the copy operation is complete, this message is displayed if any errors were encountered while copying data. Enter Y, and CV produces a list of files which may have been damaged by the errors. Otherwise, enter N.

BAD File(s)
: This message is displayed before each of the files in a bad file summary list. The files listed with this message should not be considered good, and should be examined to determine the extent of file damage.

Abnormal Termination HH:MM:SS
: Under certain circumstances, CV is unable to successfully complete the copy, and terminates the operation. The individual errors that led to the abnormal termination are listed on the lower half of your screen. After an abnormal termination, the data on the destination disk may be left in a questionable state. In this situation, the volume name of the destination disk is changed by CV to $$$$$$$$. Before that particular disk can be used again, you must execute the INV command to initialize the volume.

Abnormal termination - run time error HH:MM:SS
: In rare instances, CV may terminate due to fatal system errors. This message indicates a serious situation, and your customer representative should be consulted. In this situation, the volume name of the destination disk is changed by CV to $$$$$$$$. Before that particular disk can be used again, you must execute the INV command to initialize the volume.

Notes:

If you use CV from a terminal that is not a VDT or is a VDT not in VDT mode, the display of progress will not be shown.

All file security information for files on the source disk will be maintained on the destination disk.

Related Commands:

BD (Backup Directory)
BDD (Backup Directory to Device)
CD (Copy Directory)
CVD (Copy and Verify Disk)
DCOPY (Disk Copy/Restore)

CVD (COPY AND VERIFY DISK)

4.49 CVD (COPY AND VERIFY DISK)

Prompts:

```
COPY AND VERIFY DISK

ANSWER (Y/N) QUESTIONS WITH Y FOR YES OR
ANY OTHER CHARACTER EXCEPT $ FOR NO

RESPOND ANYTIME WITH $ TO RESTART
```

After beginning execution with this announcement, the following prompts are displayed sequentially. Each prompt is displayed after entering a response to the previous prompt.

```
LISTING DEVICE NAME— devicename
COPY WITHOUT VERIFY? (Y/N) Y/N
MASTER DEVICE – devicename
      VOLUME – volumename
COPY DEVICE – devicename
      VOLUME – volumename
MOUNT DESIRED VOLUMES--TYPE CR WHEN READY
```

Mount the disks in the devices previously specified and when ready to begin the copy procedure, press the Return key. When the copy and verify operations are complete, the following message and prompt are displayed:

```
COPY AND VERIFY COMPLETE
dd mmm yyyy—hh:mm:ss
QUIT? (Y/N) Y/N
```

This message signifies that a usable copy has been produced and verified without errors. The *dd mmm yyyy* sequence represents the day, month, and year, respectively, on which the copy is made. Likewise, the hh:mm:ss sequence represents the hour, minute, and second on which the copy and verify are completed. The QUIT? (Y/N) prompt follows the message.

Purpose:

The CVD command copies from disk to disk with an optional verify operation. The copy disk and the master disk must be the same type of disk. The copy procedure is performed file by file with disk compression on all files except key-indexed files. Program files are partially compressed by releasing the unused space at the end of the file. Space available in program files as a result of deleting tasks cannot be released with the CVD command. A Copy Directory (CD) command is required to release the additional space in program files.

CVD (COPY AND VERIFY DISK)

Prompt Details:

LISTING DEVICE NAME—
The spooler logical name of a print device. Press the Return key or enter the device name DUMY if a hardcopy list is not necessary. Once you enter the device name, it cannot be changed.

COPY WITHOUT VERIFY? (Y/N)
If you enter Y, a copy is produced without verification. If you enter N, the copy operation is followed by the verification process. The default is N.

MASTER DEVICE –
VOLUME –
Enter the device name and volume name of the original disk from which the copy is to be made.

COPY DEVICE –
VOLUME –
Enter the device name and volume name of the destination disk.

QUIT? (Y/N)
If the response is Y, the CVD command terminates. If you enter a negative response, CVD returns to the COPY WITHOUT VERIFY? (Y/N) prompt, allowing another copy/verify process. The default is N.

Example:

In the following example, the CVD command performs copy and verify disk operations from the original disk to another disk of the same type (for example, from DS50 to DS50).

```
[] CVD

       System Prompts                          User Responses


COPY AND VERIFY DISK

ANSWER (Y/N) QUESTIONS WITH Y FOR YES OR
ANY OTHER CHARACTER EXCEPT $ FOR NO
RESPOND ANYTIME WITH $ TO RESTART
LISTING DEVICE NAME--PRINTER                 Enter the name of a
                                             spooler device or a
                                             spooler logical name.

COPY WITHOUT VERIFY? (Y/N) N                 A negative response
                                             indicates the copy
                                             process is followed by
                                             verification.
```

CVD (COPY AND VERIFY DISK)

```
MASTER DEVICE-DS02                          The disk drive and
   VOLUME-MYDISK                            volume name of the
                                            source disk.

COPY DEVICE-DS03                            The disk drive and
   VOLUME-HISDISK                           volume name of the
                                            copy disk.

MOUNT DESIRED VOLUMES, TYPE CR WHEN READY   Press the Return key.
01 MAR 1982--13:21:22
COPY VOLUME NAME ERROR. SPECIFIED = HISDISK
ACTUAL = SCRATCH.
USE NAMES AS TYPED? (Y/N) Y                 Y response changes
                                            the copy volume name
                                            SCRATCH to the volume
                                            name HISDISK.

COPY AND VERIFY COMPLETE
01 MAR 1982--13:45:51
QUIT? (Y/N) Y                               Enter Y to complete
                                            copy/verify process.
```

You can perform the CVD command using the system disk as a master disk or a copy disk. When you use the system disk for the CVD operation, all other terminals must be logged off. After the copy process begins, logging on the system is prohibited. The proper procedure for using the CVD command when the system disk is involved includes the following steps:

1. Perform the initial program load (IPL).
2. Set the date and time using the Initialize Date and Time (IDT) command.
3. Use the Execute Operator Interface (XOI) command to become the system operator. Press the Command key.
4. If you are using a printer as the listing device, use the Modify Spooler Device (MSD) command to make the printer shared or not available to the spooler.
5. Use the List Jobs (LJ) command to determine jobs which are running.
6. Ask other users to terminate their jobs and log off.
7. Use the Kill Jobs (KJ) command to terminate all other jobs excluding the system job and the user job under which the CVD is to be performed.
8. Issue the Quit Operator Interface (QOI) command.
9. Execute the CVD command.
10. Count the appropriate volume in the system disk drive.
11. Perform another IPL.

CVD (COPY AND VERIFY DISK)

In the next example, DS01 represents the system disk. The CVD command is executed as follows:

```
[] CVD
```

System Prompts	User Responses
COPY AND VERIFY DISK	
ANSWER (Y/N) QUESTIONS WITH Y FOR YES OR ANY OTHER CHARACTER EXCEPT $ FOR NO RESPOND ANYTIME WITH $ TO RESTART	
LISTING DEVICE NAME--LP01	Enter the name of a print device.
COPY WITHOUT VERIFY? (Y/N) N	A negative response indicates the copy process is followed by verification.
MASTER DEVICE-DS01 VOLUME-SYSDISK	The disk drive and volume name of the system disk.
COPY DEVICE-DS03 VOLUME-MYDISK	The disk drive and volume name of the copy disk.
MOUNT DESIRED VOLUMES, TYPE CR WHEN READY 01 MAR 1982--13:21:22	Press the Return key.
COPY VOLUME NAME ERROR. SPECIFIED = MYDISK ACTUAL = SCRATCH.	
USE NAMES AS TYPED? (Y/N) Y	Y response changes the copy volume name SCRATCH to the volume name MYDISK.
COPY AND VERIFY COMPLETE 01 MAR 1982--13:45:51	
QUIT? (Y/N) Y	Enter Y to complete copy/verify process.

CVD (COPY AND VERIFY DISK)

Messages:

< MASTER or COPY> VOLUME NAME ERROR. SPECIFIED = < volume name> .
ACTUAL = < volume name> .
You have supplied a volume name different from the actual volume name of either the master disk or the copy disk. This error prompts the message USE NAMES AS TYPED? (Y/N) which allows you to continue with the specified name, mount a different volume, or respecify the device and volume names.

ALL OTHER ACTIVE TERMINALS AND TASKS MUST BE TERMINATED
TYPE CR WHEN READY
When the system disk is being used in the copy process, this message is displayed until all terminals are logged off or until you enter a $ character and press the Return key.

PROCESS USED SYSTEM DISK — DO IPL
This message is displayed upon successful completion of the CVD involving the system disk. This causes the system to halt with > 177 displayed on the front panel, turning on the fault light for the operator and programmer panels. To continue, perform an IPL.

Errors occurring while copying a specific file can indicate a failure by the hardware, software, or user. The following errors can occur while copying a file:

UNABLE TO ALLOCATE DISK SPACE FOR < filename>
The system has been unable to find enough contiguous space on the copy disk for the file indicated. This may be caused by imperfections on the copy disk. You should copy the file to another disk.

VERIFICATION ERROR
A hardware error was encountered. The file was not verified due to a data check failure between the source and copy disks. Have a local system programmer examine the system. If no problem can be detected, call a customer representative.

INTERNAL STACK OVERFLOW
A system task has overflowed its allocated stack. Have a local system programmer examine the system. If no problem can be detected, call a customer representative.

SVC ERROR < error number>
An SVC error was encountered. Refer to the specified error number listed in the *DNOS Messages and Codes Reference Manual*. A description of the error and suggested user actions are included.

I/O ERROR < error number>
An I/O error was encountered. Refer to the specified error number listed in the *DNOS Messages and Codes Reference Manual*. A description of the error and suggested user actions are included.

INTERNAL ERROR
An unknown error has occurred. This can be a user, hardware, or system error. Have a local system programmer examine the system. If no problem can be detected, call a customer representative.

CVD (COPY AND VERIFY DISK)

CANNOT TERMINATE SYSLOG TASK
CVD was unable to terminate the system log task. Perform an initial program load and try the operation again.

DISK MUST BE ANALYZED BEFORE USE—ANALYZE? (Y/N)
An IDS has not been performed on this disk. You should answer '$' to terminate CVD and perform an IDS. If this is not possible, answer 'Y', and CVD will analyze the disk.

DISK FATAL ERROR
CVD cannot continue due to a serious error. If the system disk is not involved in the copy, examine the system log file for errors on the disk. Take the action appropriate for the errors that occurred.

PROCESS TERMINATED DUE TO I/O ERROR.
THE COPY BEING MADE HAS BEEN DESTROYED.
A serious I/O error has occurred. The copy disk contains none of the data that was copied to it. The copy disk will require an INV.

DISK NOT INITIALIZED.
INITIALIZE THIS DISK? (Y/N)
The disk needs to be initialized. You should answer '$', then execute the INV command.

Notes:

The announcement banner specifically states that any character other than Y (excluding the $ sign) is interpreted as a negative response. Furthermore, when responding to prompts requesting a Y/N response, it is important to remember that CVD uses the *last* character entered for the actual response. For example, *YES* is interpreted as a negative response, while *NOY* is interpreted as a positive response.

Entering the $ character as response to a prompt causes CVD to terminate current operation and display the QUIT? (Y/N) prompt. The $ character is only valid when CVD has prompted for input.

If the CVD command recognizes an unrecoverable error on the copy disk, the process is aborted. The copy disk is rendered unusable without a subsequent INV command to guarantee the integrity of the data on the disk.

It is not recommended that you perform the copy operation without verification. However, this option is available and saves one-third of the time involved when you eliminate the verification process.

The CVD command does not support a verify only function; therefore, you must perform verification of the copy during the copy process.

CVD (COPY AND VERIFY DISK)

All file security information for files on the source disk will be maintained on the destination disk.

Related Commands:

BD	(Backup Directory)
CD	(Copy Directory)
CV	(Copy Volume)
DCOPY	(Disk Copy/Restore)
RD	(Restore Directory)
VB	(Verify Backup)
VC	(Verify Copy)

DA (DELETE ALIAS FROM PATHNAME)

4.50 DA (DELETE ALIAS FROM PATHNAME)

Prompts:

```
DELETE ALIAS FROM PATHNAME
                PATHNAME:  [site:] pathname@
```

Purpose:

Use the DA command to delete an alias assignment. You can only delete one alias with a single DA command.

Prompt Details:

PATHNAME:
A pathname that identifies the alias to be deleted. The alias is the last element of the pathname.

Example:

In the following example, the alias FILEB is assigned to a file named FILE1 and then deleted from that file name. The Add Alias (AA) and DA commands assign and delete the alias as shown below:

```
[] AA

ADD ALIAS TO PATHNAME
            PATHNAME: VOL1.PROGDIR.DIRA.FILE1
      ALIAS PATHNAME: VOL1.PROGDIR.DIRA.FILEB

[] DA

DELETE ALIAS FROM PATHNAME
         PATHNAME: VOL1.PROGDIR.DIRA.FILEB
```

Related Commands:

AA (Add Alias)

DAG (DELETE ACCESS GROUP)

4.51 DAG (DELETE ACCESS GROUP)

Prompts:

```
DELETE ACCESS GROUP
                    PASSCODE:   – character(s)
          ACCESS GROUP NAME:    alphanumeric
```

Purpose:

The DAG command deletes the specified access group. The user of this command must be the leader of the specified access group or must be a member of the SYSMGR access group. In either case, the group leader must be the only member of the group. DAG does not remove the access group from the access list associated with any files. It is assumed that the leader or the users in the group have already used the Modify Security Access Rights (MSAR) command to remove the group from the list of associated access groups for each file with which the group was associated.

The system determines to which access groups a user belongs when a job is created with the user's ID. Modifications to an access group's membership do not take effect until a new job is created with that user's ID. If currently logged on to SCI, the user must log off and log on again.

Prompt Details:

PASSCODE:
Enter your assigned log-on passcode.

ACCESS GROUP NAME:
Specify the name of the access group to be deleted. The access group being deleted can contain only one member, the group leader. If access group members other than the access group leader need to be removed from the access group, use the Modify Access Group (MAG) command.

Example:

In the following example, a user with the passcode P89J55 is the leader and only remaining member of the access group named CLERKS. The access group named CLERKS is deleted. (The passcode is not echoed to the user's terminal.)

```
[] DAG

DELETE ACCESS GROUP
                PASSCODE:
      ACCESS GROUP NAME: CLERKS
```

DAG (DELETE ACCESS GROUP)

Notes:

It is the responsibility of the user issuing the DAG command to ensure that no files exist which have the specified access group in the list of associated access groups. The List Access Group File Rights (LAGFR) command is available to assist in this process.

If you delete your creation access group, the PUBLIC access group becomes your creation access group after you log off. If you create any files before you log off, your current creation access group is set into the files even though it has been deleted.

Related Commands:

CAG	(Create Access Group)
LAGFR	(List Access Group File Rights)
LAGM	(List Access Group Members)
MAG	(Modify Access Group)

DB (DELETE BREAKPOINTS)

4.52 DB (DELETE BREAKPOINTS)

Prompts:

```
DELETE BREAKPOINTS
                    RUN ID:  {integer/ALL}          (*)
              ADDRESS(ES):   [full exp list/ALL]
```

If you enter ALL in response to the RUN ID prompt, the following prompt is displayed:

```
DELETE ALL BREAKPOINTS FOR ALL TASKS IN THIS JOB
                ARE YOU SURE?:  YES/NO
```

Purpose:

The DB command deletes a breakpoint(s) from a specified address(es) in a task and restores the original value that was replaced with an XOP 15,15 instruction at that address(es). The task is temporarily suspended while the breakpoint(s) is deleted; then, the original task state is restored. Deleting a breakpoint where a task is stopped does not cause the task to resume execution; you must use the Resume Task (RT) command to cause the task to resume execution. Alternatively, you can use the Delete and Proceed from Breakpoint (DPB) command to delete a breakpoint and resume execution.

The parameters of the DB command are interpreted as in the AB command, with the following exceptions:

- If you specify ALL in response to the RUN ID prompt and YES to the ARE YOU SURE? prompt, all breakpoints for all tasks in the job are deleted, including breakpoints which were set in tasks that have terminated.
- If you do not specify an address in response to the ADDRESS(ES) breakpoint at which the task is currently stopped is deleted.
- If you specify ALL in response to the ADDRESS(ES) prompts, breakpoints for that task are deleted.
- If the breakpoint does not exist at the specified or implied address, or a breakpoint within a list of breakpoints does not exist, you are warned with an error message.

DB (DELETE BREAKPOINTS)

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

You can also specify ALL to delete all breakpoints for all tasks in the job. If you specify ALL, an additional prompt is displayed and any response to the ADDRESS(ES) prompt is ignored.

ADDRESS(ES):
The integer value(s) of the breakpoint address(es) within the task set by the AB command. Separate the addresses by commas. If you enter ALL, all breakpoints are deleted. The default breakpoint is the one at which the task is currently stopped.

ARE YOU SURE?:
This prompt is displayed only when you specify ALL in response to the RUN ID prompt. Respond YES to delete all breakpoints for all tasks (both active and terminated) in the job. Respond NO to abort the command.

Example:

In the following example, the DB command deletes specified breakpoints within a task with a run ID of >A0:

```
[] DB

DELETE BREAKPOINTS
          RUN ID: >A0
    ADDRESS(ES): >200,>41A,>AC,>506
```

Notes:

If your terminal is in VDT mode, the debug panel is displayed as explained in the description of the Show Panel (SP) command before SCI is reactivated.

Related Commands:

AB	(Assign Breakpoint)
DPB	(Delete and Proceed from Breakpoint)
LB	(List Breakpoints)
PB	(Proceed from Breakpoint)
SP	(Show Panel)

DBP (DELETE BREAKPOINT — PASCAL)

4.53 DBP (DELETE BREAKPOINT — PASCAL)

The DBP command deletes a breakpoint at entry to, return from, or both entry to and return from a Pascal routine. Refer to the *DNOS TI Pascal Programmer's Guide* for further information on the DPB command and the following prompts:

```
DELETE BREAKPOINT - PASCAL
                    RUN ID:  integer exp                (*)
              ROUTINE NAME:  character(s)
WHERE (ENTRY/RETURN/BOTH):  [{ENTRY/RETURN/BOTH}]
```

4.54 DCOPY (DISK COPY/RESTORE)

Prompts:

```
DISK COPY/RESTORE

ANSWER (Y/N) QUESTIONS WITH Y FOR YES OR
ANY OTHER CHARACTER EXCEPT $ FOR NO

RESPOND ANYTIME WITH $ TO RESTART
```

After beginning execution with this announcement, DCOPY displays the following prompts sequentially. Each prompt is displayed after entering a response to the previous prompt.

```
LISTING DEVICE NAME— devicename
VERIFY ONLY? (Y/N) Y/N
COPY WITHOUT VERIFY? (Y/N) Y/N
MASTER DEVICE – devicename
      VOLUME – volumename
COPY DEVICE – devicename
      VOLUME – volumename
MOUNT DESIRED VOLUMES—TYPE CR WHEN READY
```

Mount the disks in the devices that you previously specified and press the Return key when you are ready to begin the copy procedure. When the copy and verify operations are complete, the following message and prompt are displayed:

```
COPY AND VERIFY COMPLETE
dd mmm yyyy—hh:mm:ss
QUIT?    (Y/N)   Y/N              (N)
```

This message signifies that a usable copy has been produced and verified without errors. The *dd mmm yyyy* sequence represents the day, month, and year, respectively, which the copy is made. Likewise, the hh:mm:ss sequence represents the hour, minute, and second which the copy and verify are completed. The QUIT? (Y/N) prompt follows the message.

Purpose:

The DCOPY command copies and optionally verifies disks used with the DNOS operating system. You can perform the copy process from disk to magnetic tape, magnetic tape to disk, or disk to disk. The copy operation is processed track by track without disk compression. A disk-to-disk copy requires that the copy disk be the same type as the master disk, for example, two DS50s or two DS10s.

When you use a magnetic tape unit as an output device, the first and last records are identification records of ASCII characters. Each identification record contains a volume name, the date of creation, a reel number, and data describing the type of disk from which the tape was created. The reel number contained in the identification records at the beginning and end of the tape provides the means for copying disks to or from multiple tapes.

DCOPY (DISK COPY/RESTORE)

Prompt Details:

LISTING DEVICE NAME—
The spooler logical name of a print device. Press the Return key or enter the device name DUMY if a hardcopy list is not necessary. Once you enter the device name, it cannot be changed. You cannot use a device that is currently available to the spooler as a listing device for DCOPY.

VERIFY ONLY? (Y/N)
If you enter Y as a response, the copy disk is compared to the original disk to verify the disks are identical. If you enter N as a response, the COPY WITHOUT VERIFY? (Y/N) prompt is displayed. The default is N.

COPY WITHOUT VERIFY? (Y/N)
This prompt is invoked after you enter a negative response for the VERIFY ONLY? (Y/N) prompt. If you enter Y in response to the COPY WITHOUT VERIFY? (Y/N) prompt, a copy is produced without verification. If you enter N, the copy operation is followed by the verification process. The default is N.

MASTER DEVICE –
VOLUME –
Enter the device name and volume name of the original disk from which the copy is to be made.

COPY DEVICE –
VOLUME –
Enter the device name and volume name of the destination disk.

QUIT? (Y/N)
If the response is Y, the DCOPY command terminates. If you enter a negative response, DCOPY returns to the VERIFY ONLY?(Y/N) prompt, allowing another verification process with the same disk volumes.

DCOPY (DISK COPY/RESTORE)

Example:

In the following example, the DCOPY command performs copy and verify disk operations from the original disk to another disk of the same type (for example, from DS50 to DS50).

```
[] DCOPY
```

System Prompts	User Responses
DISK COPY/RESTORE	
ANSWER (Y/N) QUESTIONS WITH Y FOR YES OR ANY OTHER CHARACTER EXCEPT $ FOR NO RESPOND ANYTIME WITH $ TO RESTART	
LISTING DEVICE NAME--PRINTER	Enter the spooler logical name of a print device.
VERIFY ONLY? (Y/N) N	A negative response invokes COPY WITHOUT VERIFY? prompt.
COPY WITHOUT VERIFY? (Y/N) Y	A positive response indicates the copy process is performed without verification.
MASTER DEVICE-DS02 VOLUME-MYDISK	The disk drive and volume name of the original disk.
COPY DEVICE-DS03 VOLUME-HISDISK	The disk drive and volume name of the copy disk.
MOUNT DESIRED VOLUMES, TYPE CR WHEN READY 01 MAR 1982--13:21:22	Press the Return key.
COPY VOLUME NAME ERROR. SPECIFIED = HISDISK ACTUAL = SCRATCH.	
USE NAMES AS TYPED? (Y/N) Y	Y response changes copy volume name SCRATCH to volume name HISDISK.
COPY AND VERIFY COMPLETE 01 MAR 1982--13:45:51	
QUIT? (Y/N) Y	Enter Y to complete copy/verify process.

DCOPY (DISK COPY/RESTORE)

You can perform the DCOPY command using the system disk as a master disk or a copy disk. When you use the system disk for the DCOPY operation, you must terminate jobs and tasks; all other terminals must be logged off. After the copy process begins, logging on the system is prohibited. The proper procedure for using the DCOPY command when the system disk is involved includes the following steps:

1. Perform the initial program load (IPL).
2. Set the date and time using the Initialize Date and Time (IDT) command.
3. Use the Execute Operator Interface (XOI) command to become the system operator.
4. If you are using a printer as the listing device, use the Modify Spooler Device (MSD) command to make the printer shared or not available to the spooler.
5. Use the List Jobs (LJ) command to determine jobs which are running.
6. Use the Kill Jobs (KJ) command to terminate all other jobs excluding the system job and the job under which the DCOPY is to be performed.
7. Issue the Quit Operator Interface (QOI) command.
8. Execute the DCOPY command.

Performing these steps ensures that memory-resident or time-delay tasks do not attempt to modify the system disk during the course of the copy. In the next example, DS01 represents the system disk. The DCOPY command is executed as follows:

```
[] DCOPY

       System Prompts                        User Responses

DISK COPY/RESTORE

ANSWER (Y/N) QUESTIONS WITH Y FOR YES OR
ANY OTHER CHARACTER EXCEPT $ FOR NO
RESPOND ANYTIME WITH $ TO RESTART
LISTING DEVICE NAME--LP01                    Enter listing  device
                                             name.

VERIFY ONLY? (Y/N) N                         A  negative  response
                                             invokes  COPY WITHOUT
                                             VERIFY?  prompt.
```

DCOPY (DISK COPY/RESTORE)

```
COPY WITHOUT VERIFY? (Y/N) N                    A negative response
                                                indicates the copy
                                                operation is followed
                                                by the verification
                                                process.

MASTER DEVICE-DS01                              The disk drive and
   VOLUME-SYSDISK                               volume name of the
                                                system disk.

COPY DEVICE-DS03                                The disk drive and
   VOLUME-MYDISK                                volume name of the
                                                copy disk.

MOUNT DESIRED VOLUMES, TYPE CR WHEN READY       Press the Return key.
01 MAR 1982--13:21:22

COPY VOLUME NAME ERROR. SPECIFIED = MYDISK
ACTUAL = SCRATCH.

USE NAMES AS TYPED? (Y/N) Y                     Y response changes
                                                copy volume name
                                                SCRATCH to volume
                                                name MYDISK.

COPY AND VERIFY COMPLETE
01 MAR 1982--13:45:51

QUIT? (Y/N) Y                                   Enter Y to complete
                                                copy/verify process.
```

Messages:

< MASTER or COPY> VOLUME NAME ERROR. SPECIFIED = < volume name> .
ACTUAL = < volume name> .

You have supplied a volume name different from the actual volume name of either the master disk or the copy disk. This error prompts the message USE NAMES AS TYPED? (Y/N) which allows you to continue with the specified name, mount a different volume, or respecify the device and volume names.

ALL OTHER ACTIVE TERMINALS, JOBS, AND TASKS MUST BE TERMINATED

When you use the system disk in the copy process, this message is displayed until all terminals are logged off or until you enter a $ character and press the Return key.

PROCESS USED SYSTEM DISK — DO IPL

This message is displayed upon successful completion of the DCOPY where the system disk is used in the copy process. This causes the system to halt with > 177 displayed on the front panel, turning on the fault light for the operator and programmer panels. To continue, you must perform an IPL.

SVC ERROR < error number>

An SVC error was encountered. Refer to the specified error number listed in the *DNOS Messages and Codes Reference Manual* which includes a description of the error and suggested user actions.

DCOPY (DISK COPY/RESTORE)

I/O ERROR < error number>
An I/O error was encountered. Refer to the specified error number listed in the *DNOS Messages and Codes Reference Manual* which includes a description of the error and suggested user actions.

CANNOT USE INSTALLED DISK
You cannot use DCOPY on an installed disk. Use the Unload Volume (UV) command to unload the disk.

INCOMPATIBLE DISK TYPES: RESPECIFY
In a disk-to-disk copy, both disks must be of the same type, two DS50s for example.

CANNOT ASSIGN DEVICE
The specified device is either offline or not configured.

MASTER DISK OFFLINE/NOT READY — TYPE < CR> WHEN READY
Before you can use DCOPY, you must bring the drive of the master disk online and make the master disk ready.

CANNOT TERMINATE SYSLOG TASK
DCOPY was unable to terminate the system log task. Perform an initial program load and try the operation again.

DISK MUST BE ANALYZED BEFORE USE—ANALYZE? (Y/N)
An IDS has not been performed on this disk. You should answer '$' to terminate DCOPY and perform an IDS. If this is not possible, answer 'Y', and DCOPY will analyze the disk.

DISK FATAL ERROR
DCOPY cannot continue due to a serious error. If the system disk is not involved in the copy, examine the system log file for errors on the disk. Take the action appropriate for the errors that occurred.

PROCESS TERMINATED DUE TO I/O ERROR.
THE COPY BEING MADE HAS BEEN DESTROYED.
A serious I/O error has occurred. The copy disk contains none of the data that was copied to it. The copy disk will require an INV.

DISK NOT INITIALIZED.
INITIALIZE THIS DISK? (Y/N)
The disk needs to be initialized. You should answer '$', then execute the INV command.

TRACK ALLOCATION CONFLICT. TRACKS MARKED BAD ARE NEEDED FOR IMAGE RESTORE
The bad track list on the master disk does not match that on the copy disk. You should use CVD or CV to perform this copy.

READ ERROR. TRACK = XXXX. CHARACTER COUNT = YYYY.

DCOPY has encountered an error trying to read from the master disk and was unsuccessful after several retries. XXXX is the track number that has the error. YYYY is the number of characters successfully read before the error. The copy disk will require an INV. An alternative is to use the CV command to perform the copy.

WRITE ERROR. TRACK = XXXX. CHARACTER COUNT = YYYY.

DCOPY has encountered an error trying to write to the copy disk and was unsuccessful after several retries. XXXX is the track number that has the error. YYYY is the number of characters successfully written before the error. The copy disk will require an INV. An alternative is to use the CV command to perform the copy.

DON'T HAVE XXXX BLOCKS OF MEMORY

DCOPY does not have enough memory to perform its job. It needs to run on a system with more physical memory or you need to generate a smaller operating system. Refer to the *DNOS System Generation Reference Manual* for information.

VERIFY ERROR—TRACK XXXX, BYTE YYYY

DCOPY could not verify the data just written. XXXX and YYYY define the physical location of the error. You should perform an IDS on the copy disk, then attempt the copy using the CVD or CV command.

Notes:

The announcement banner specifically states that any character other than Y (excluding the $ sign) is interpreted as a negative response. Furthermore, when responding to prompts requesting a Y/N response, it is important to remember that DCOPY uses the *last* character entered for the actual response. For example, *YES* is interpreted as a negative response, while *NOY* is interpreted as a positive response.

Entering the $ character as response to a prompt causes DCOPY to terminate the current operation and display the QUIT? (Y/N) prompt. The $ character is only valid when DCOPY has prompted for input.

If the DCOPY command recognizes an unrecoverable error on the copy disk, the process is aborted. The copy disk is rendered unusable without a subsequent INV command to guarantee the integrity of the data on the disk.

DCOPY (DISK COPY/RESTORE)

One or more system files on the system disk are updated when you terminate DCOPY. Therefore, when you make a copy of the system disk, you *must* verify the copy before you terminate DCOPY.

A character count of 1 through 100 inclusive indicates an error during a disk format I/O call (read or write operation).

All file security information for files on the source disk will be maintained on the destination disk.

Related Commands:

BD	(Backup Directory)
CD	(Copy Directory)
CV	(Copy Volume)
CVD	(Copy and Verify Disk)
RD	(Restore Directory)
VB	(Verify Backup)
VC	(Verify Copy)

4.55 DD (DELETE DIRECTORY)

Prompts:

```
DELETE DIRECTORY
                    PATHNAME:  [site:]pathname@
          LISTING ACCESS NAME:  [site:][pathname]@
                ARE YOU SURE?:  YES/NO
```

Purpose:

The DD command deletes a specified directory plus all subdirectories and files in which the directory is a pathname component. However, the DD command does not delete any directory, subdirectory, or file that is delete protected. If a directory, subdirectory, or file is not deleted because it is delete protected, the DD command specifies the pathname in the listing file and indicates that it is delete protected. The DD command then continues execution.

Prompt Details:

PATHNAME:
The pathname identifying the directory to be deleted. If the pathname identifies a file, that file is deleted.

LISTING ACCESS NAME:
The device or file name to which the DD command should write the results of the delete operation. The default is the terminal local file where the DD command is executed.

ARE YOU SURE?:
If you specify YES, you are sure the directory is to be destroyed. NO cancels the command. This prompt helps ensure that the correct directory is deleted.

Example:

In the following example, the DD command deletes a directory and all the subdirectories and files in which the directory is a pathname component.

```
[] DD

 DELETE DIRECTORY
                  PATHNAME: .EXAMPLE
      LISTING ACCESS NAME:
            ARE YOU SURE?: YES
   DELETED   .EXAMPLE.DIR1.FILE1
   DELETED   .EXAMPLE.DIR1.FILE2
   DELETED   .EXAMPLE.DIR1
   DELETED   .EXAMPLE.FILE1
   DELETED   .EXAMPLE
```

DD (DELETE DIRECTORY)

Messages:

The DD command lists each subdirectory and file that is deleted to the device or file specified for the LISTING ACCESS NAME prompt. If you delete a file or directory, the listing file contains a message indicating the reason for the failure.

Notes:

Even though a directory is delete protected, the files in that directory are deleted unless they are also delete protected.

No error condition is set if the directory does not exist. This circumstance allows batch streams to delete directories without having to deal with error messages about the directory not being there.

Related Commands:

CFDIR	(Create Directory File)
LD	(List Directory)

4.56 DF (DELETE FILE)

Prompts:

DELETE FILE
PATHNAME(S): [site:]filename@...[site:]filename@

Purpose:

The DF command deletes a file (or files) of any file type available on the system. If the file is a directory, it must be empty or the command is aborted. You can delete nonempty directories only by the DD command. If the file to be deleted is delete protected, the delete file command fails and reports an error.

Prompt Details:

PATHNAME(S):
Enter the access name(s) of the file(s) to be deleted.

Example:

In the following example, the DF command deletes a file named .EXAMPLE:

```
[] DF

DELETE FILE
         PATHNAME(S): .EXAMPLE
```

Related Commands:

CF (Create File)

DIC (DELETE IPC CHANNEL)

4.57 DIC (DELETE IPC CHANNEL)

Prompts:

```
DELETE IPC CHANNEL
        CHANNEL PATHNAME:   [site:]pathname@
```

Purpose:

The DIC command allows you to delete a channel. On an unsuccessful completion of this command, a message describing the error is displayed on your terminal.

Prompt Details:

CHANNEL PATHNAME:
The pathname of the channel to be deleted.

Example:

The following example deletes the channel .KC0017.REPORT:

```
[ ] DIC

DELETE IPC CHANNEL
       CHANNEL PATHNAME: .KC0017.REPORT
```

Notes:

If you delete the program file on which a channel owner task resides, the associated channel is also automatically deleted.

Related Commands:

CIC	(Create IPC Channel)
SCS	(Show Channel Status)

4.58 DISC (TERMINAL DISCONNECTION)

Prompts:

```
TERMINAL DISCONNECTION
  TERMINAL ACCESS NAME:  stationname                (*)
```

Purpose:

The DISC command is utilized to terminate a connection for a remote teleprinter device (TPD).

Prompt Details:

TERMINAL ACCESS NAME:
A valid system terminal name. The device must be logically turned on and connected.

Example:

The following example terminates the connection with the TPD that was established as ST21 during system generation.

```
[]DISC

TERMINAL DISCONNECTION
 TERMINAL ACCESS NAME: ST21
```

Messages:

When terminating a connection at another station, a completion message is displayed at the controlling station. If the terminal being disconnected is ME, no message is displayed for a successful disconnection. If an edit is active (that is, the synonym $$EA = Y), the following message is displayed:

NO DISCONNECT – EDIT IN PROGRESS

If the SCI debugger is active (that is, the synonym $$DA = Y), the following message is displayed:

NO DISCONNECT – DEBUG IN PROGRESS

DISC (TERMINAL DISCONNECTION)

Notes:

Terminal disconnection is allowed, provided no task is currently active at the terminal. If other users have LUNOs assigned to the terminal, disconnection is not allowed.

If the terminal is ME (that is, DISC is run on the remote terminal), disconnection is allowed if SCI and DISC are the only active tasks; however, the DISC task cannot be running in background mode. Additionally, if the terminal is ME, DISC/PROC forces Quit SCI (Q) processing.

Related Commands:

ANS	(Answer Incoming Call)
CALL	(Call Terminal)
LHPC	(List Hardcopy Terminal Port Characteristics)
MHPC	(Modify Hardcopy Terminal Port Characteristics)

4.59 DL (DELETE LINES)

After you activate the Text Editor, the DL command deletes the designated set of lines from the current file. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the DL command and the following prompts:

```
DELETE LINES
        START LINE:  [BEGINNING/END/line number]
          END LINE:  [BEGINNING/END/line number]
```

DO (DELETE OVERLAY)

4.60 DO (DELETE OVERLAY)

Prompts:

```
DELETE OVERLAY
        PROGRAM FILE OR LUNO: {[site:]filename/integer}@          (*)
          OVERLAY NAME OR ID: {alphanumeric/integer}
```

Purpose:

Use the DO command to remove a previously installed overlay from a program file. You can specify the overlay by name or by ID.

Prompt Details:

PROGRAM FILE OR LUNO:
The file name of, or the LUNO assigned to, the program file on which the overlay has been installed. If you specify a LUNO in response to this prompt, you must assign it to the program file prior to execution of the DO command. If you specify zero, the .S$SHARED program file is assumed. If you specify >FF, the utility program file is assumed.

OVERLAY NAME OR ID:
The name or installed ID of the overlay that is to be deleted from the specified program file.

Example:

In the following example, the DO command deletes the overlay named OLAY1 from the program file specified by LUNO >80.

```
[] DO

DELETE OVERLAY
     PROGRAM FILE OR LUNO: >80
       OVERLAY NAME OR ID: OLAY1
```

Notes:

Delete protected overlays cannot be deleted unless you remove the delete protection by executing the Modify Overlay Entry (MOE) command.

Related Commands:

DP	(Delete Procedure Segment)
DPS	(Delete Program Segment)
DT	(Delete Task Segment)
IO	(Install Overlay)
MOE	(Modify Overlay Entry)

DP (DELETE PROCEDURE SEGMENT)

4.61 DP (DELETE PROCEDURE SEGMENT)

Prompts:

```
DELETE PROCEDURE SEGMENT
      PROGRAM FILE OR LUNO:   {[site:]filename/integer}@          (*)
      PROCEDURE NAME OR ID:   {alphanumeric/integer}
```

Purpose:

The DP command removes a previously installed procedure segment from a program file. You can specify the procedure segment by name or ID.

Prompt Details:

PROGRAM FILE OR LUNO:
The file name of, or the LUNO assigned to, the program file in which the procedure segment has been installed. If you specify a LUNO in response to this prompt, you must assign it to the program file prior to execution of the DP command. If you specify zero, the .S$SHARED program file is assumed. If you specify > FF, the utility program file is assumed.

PROCEDURE NAME OR ID:
The name or ID of the procedure segment to be deleted from the specified program file.

Example:

In the following example, the DP command deletes a procedure segment named MYPROC from the program file specified by LUNO > 7B.

```
[] DP

DELETE PROCEDURE SEGMENT
     PROGRAM FILE OR LUNO: >7B
     PROCEDURE NAME OR ID: MYPROC
```

Assumptions:

The specified procedure segment is not delete protected. Delete protected procedures cannot be deleted until you remove the delete protection by executing the Modify Procedure Segment Entry (MPE) command. If the attached procedure of a task is deleted, the task will not execute. You must reinstall the procedure to execute the task.

DP (DELETE PROCEDURE SEGMENT)

Notes:

The DP command also deletes a program segment from a program file.

Related Commands:

DO	(Delete Overlay)
DPS	(Delete Program Segment)
DT	(Delete Task Segment)
IP	(Install Procedure)
MPE	(Modify Procedure Segment Entry)

DPB (DELETE AND PROCEED FROM BREAKPOINT)

4.62 DPB (DELETE AND PROCEED FROM BREAKPOINT)

Prompts:

```
DELETE AND PROCEED FROM BREAKPOINT
                        RUN ID:  integer                (*)
    DESTINATION ADDRESS(ES):  [full exp list]
```

Purpose:

The DPB command proceeds from a breakpoint at which a task is currently stopped and deletes that breakpoint. If you have already deleted the breakpoint, the command functions as if it were a Proceed from Breakpoint (PB) command. Optionally, it sets one or more breakpoints in the task.

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

DESTINATION ADDRESS(ES):
The address(es) of an additional breakpoint(s) which is to be set within the task. A null response specifies that no new breakpoints are to be set.

Example:

In the following example, the DPB command deletes the breakpoint where the task identified by run ID > 4E has stopped and assigns new breakpoints at addresses > 1A, > 2FE, and > 340. The task then resumes execution.

```
[] DPB

DELETE AND PROCEED FROM BREAKPOINT
                        RUN ID: >4E
  DESTINATION ADDRESS(ES): >1A,>2FE,>340
```

Related Commands:

AB	(Assign Breakpoint)
DB	(Delete Breakpoints)
LB	(List Breakpoints)
PB	(Proceed from Breakpoint)
SP	(Show Panel)

DPBP (DELETE AND PROCEED FROM BREAKPOINT — PASCAL)

4.63 DPBP (DELETE AND PROCEED FROM BREAKPOINT — PASCAL)

The DPBP command deletes a breakpoint that suspends a Pascal task, optionally assigns another breakpoint, and resumes execution of the task. Refer to the *DNOS TI Pascal Programmer's Guide* for further information on the DPBP command and the following prompts:

```
DELETE AND PROCEED FROM BREAKPOINT - PASCAL
                    RUN ID:  integer exp                (*)
       DESTINATION ROUTINE:  [character(s)]
WHERE(ENTRY/RETURN/BOTH):  [{ENTRY/RETURN/BOTH}]
```

4.64 DPS (DELETE PROGRAM SEGMENT)

Prompts:

```
DELETE PROGRAM SEGMENT
        PROGRAM FILE OR LUNO:   {[site:]filename/integer}@          (*)
          SEGMENT NAME OR ID:   {alphanumeric/integer}
```

Purpose:

The DPS command deletes a program segment from a specified program file.

Prompt Details:

PROGRAM FILE OR LUNO:
The filename of, or the LUNO assigned to, the program file on which the program segment has been installed. If you specify a LUNO in response to this prompt, you must assign it to the program file prior to execution of the DPS command. If you specify zero, the .S$SHARED program file is assumed. If you specify > FF, the utility program file is assumed.

SEGMENT NAME OR ID:
The name or ID by which the program segment is known on the specified program file.

Example:

The following example deletes the program segment PRINT from the program file VOL1.USER:

```
[] DPS

DELETE PROGRAM SEGMENT
     PROGRAM FILE OR LUNO: VOL1.USER
       SEGMENT NAME OR ID: PRINT
```

Assumptions:

The specified program segment is not delete protected. Delete protected program segments cannot be deleted until you remove the delete protection by executing the Modify Program Segment Entry (MSE) command.

DPS (DELETE PROGRAM SEGMENT)

Notes:

The DPS command also deletes a procedure segment from a program file.

Related Commands:

DO	(Delete Overlay)
DP	(Delete Procedure Segment)
DT	(Delete Task Segment)
IPS	(Install Program Segment)
MSE	(Modify Program Segment Entry)

4.65 DS (DELETE STRING)

After you activate the Text Editor, the DS command deletes a designated number of occurrences of a specified string from the current file being edited. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the DS command and the following prompts:

```
DELETE STRING
    NUMBER OF OCCURRENCES:  integer          (1)
             START COLUMN:  integer          (*)
               END COLUMN:  integer          (*)
                   STRING:  [character(s)]   (*)
```

DSB (DELETE SIMULATED BREAKPOINTS)

4.66 DSB (DELETE SIMULATED BREAKPOINTS)

Prompts:

```
DELETE SIMULATED BREAKPOINTS
        BREAKPOINT NUMBERS:   [full exp list/ALL]
```

Purpose:

The DSB command deletes a list of simulated breakpoints that were assigned using the Assign Simulated Breakpoint (ASB) command.

Prompt Details:

BREAKPOINT NUMBERS:
Integer value returned by the ASB command specifies the number of the breakpoint to delete. If you enter the characters ALL, all the simulated breakpoints are deleted. The default is the breakpoint where the task is stopped. Obtain current simulated breakpoints by executing the List Simulated Breakpoints (LSB) command.

Example:

In the following example, the DSB command deletes breakpoint number three.

```
[] DSB

DELETE SIMULATED BREAKPOINT
     BREAKPOINT NUMBERS:  3
```

Related Commands:

ASB	(Assign Simulated Breakpoint)
LSB	(List Simulated Breakpoints)
ST	(Simulate Task)
XD	(Execute in Debug Mode)

4.67 DT (DELETE TASK SEGMENT)

Prompts:

```
DELETE TASK SEGMENT
        PROGRAM FILE OR LUNO:  {[site:]filename/integer}@          (*)
             TASK NAME OR ID:  {alphanumeric/integer}
```

Purpose:

Use the DT command to remove a previously installed task segment from a program file. You can delete the task segment either by name or by installed ID. If associated overlays exist, they too are deleted.

Prompt Details:

PROGRAM FILE OR LUNO:
The file name of, or LUNO assigned to, the program file on which the task segment has been installed. If you specify a LUNO in response to this prompt, you must assign it to the program file prior to execution of the DT command. If you specify zero, the .S$SHARED program file is assumed. If you specify >FF, the utility program file is assumed.

TASK NAME OR ID:
The name or ID of the task segment on the specified program file.

Example:

In the following example, the DT command deletes a task with an ID of >83 from a program file specified by LUNO >4A.

```
[] DT

DELETE TASK SEGMENT
    PROGRAM FILE OR LUNO: >4A
         TASK NAME OR ID: >83
```

Assumptions:

The task specified is not delete protected. Delete protected task segments cannot be deleted until you remove the delete protection by executing the Modify Task Segment Entry (MTE) command.

Related Commands:

DO	(Delete Overlay)
DP	(Delete Procedure Segment)
DPS	(Delete Program Segment)
IT	(Install Task)
MTE	(Modify Task Segment Entry)

DUI (DELETE USER ID)

4.68 DUI (DELETE USER ID)

Prompts:

DELETE USER ID
USER ID: alphanumeric

Purpose:

The DUI command allows authorized users to delete a user's identification information to prevent him from accessing the system.

Prompt Details:

USER ID:
The user ID to be removed from the system.

Example:

In the following example, the List User IDs (LUI) command lists the existing user IDs and their associated descriptions and privilege levels to the terminal local file. The DUI command deletes the specified user ID, and another LUI command shows that the user ID has been deleted.

```
[] LUI

LIST USER IDS
     OUTPUT ACCESS NAME:

  USER ID         DESCRIPTION        PRIVILEGE CODE
  GEORGIA     EXAMPLE USER                4
  SUPERMAN    HERB                        7
  CLARK1      CLARK JONES                 0
  CLARK2      CLARK JONES                 2
  STAR        KIRBY                       6
  VISITOR     ANY LOCAL ANALYST           7
  SYSTEM      ANY SYSTEM USER             3

16:00:21 MONDAY, APRIL 11, 1983.

[] DUI

DELETE USER ID
          USER ID: CLARK2
```

DUI (DELETE USER ID)

```
[] LUI

LIST USER IDS
     OUTPUT ACCESS NAME:

  USER ID          DESCRIPTION          PRIVILEGE CODE
  GEORGIA     EXAMPLE USER                    4
  SUPERMAN    HERB                            7
  CLARK1      CLARK JONES                     0
  STAR        KIRBY                           6
  VISITOR     ANY LOCAL ANALYST               7
  SYSTEM      ANY SYSTEM USER                 3

16:00:50 MONDAY, APRIL 11, 1983.
```

Special Cases:

You should not delete the user ID when it is in use. Deleting the user ID permits the system to delete various files associated with it.

Notes:

If a user is in the process of logging on with the ID to be deleted or is modifying that ID, the DUI command returns an error message.

When DUI is used in a file security environment, you must have read and write access rights to the .S$CLF file.

Related Commands:

AUI	(Assign User ID)
LUI	(List User IDs)
MPC	(Modify Passcode)
MUI	(Modify User ID)

EBATCH (END BATCH EXECUTION)

4.69 EBATCH (END BATCH EXECUTION)

Purpose:

The EBATCH command indicates that there are no more commands in the batch stream to process. EBATCH functions as a no-operation in interactive mode and flags partially completed CFKEY sequences in batch mode. It is recommended that the Begin Batch Execution (BATCH) command be the first command in a batch stream and the EBATCH command be the last command in a batch stream.

If you do not include the EBATCH command in a batch stream, you must use the .STOP primitive.

Example:

In the following example, the EBATCH command specifies that there are no more commands to process, requests that the synonyms be listed to the batch listing file, and specifies a new message be passed back to the foreground SCI:

```
BATCH
  *
  *
  *
EBATCH LS=YES,TEXT="BATCH COMPLETE, NO ERRORS"
```

Notes:

The EBATCH command is not intended for interactive use. The following prompts are available in batch stream format when using the EBATCH command:

```
END BATCH EXECUTION
          LS (LIST SYNONYMS)?:  [YES/NO]          (NO)
     LIST LOGICAL NAMES?:       [YES/NO]          (NO)
                   TEXT:        [character(s)]    (BATCH SCI HAS COMPLETED)
                   CODE:        [integer]
```

The LS (LIST SYNONYMS)? prompt gives you the option to list the synonyms in the listing file. The LIST LOGICAL NAMES? prompt gives you the option to list the logical names in the listing file. The response to the TEXT prompt is a user-defined character string to be passed back to the initiating foreground SCI to be used instead of the BATCH SCI HAS COMPLETED message. The CODE prompt response sets the synonym $$BC in the initiating SCI's synonyms at the completion of the batch stream. This $$BC synonym is deleted from the foreground synonyms when any new background activity (batch stream) is started.

EBATCH (END BATCH EXECUTION)

If you execute the batch stream via the Execute Batch Job (XBJ) command, no information is passed back to the initiating SCI through the TEXT prompt nor through the CODE prompt. In the case of XBJ, you can use the Create Message (CM) command to notify the interactive user that the batch job has completed. No commands are processed after the EBATCH command is encountered in the batch stream.

You must take special care when you use the CM command with XBJ, since the synonyms ME and $$ST are not set to the station where the XBJ command is executed. (Refer to the *DNOS Systems Programmer's Guide* for information concerning the ME and $$ST synonyms.)

Messages:

The user-defined message specified in response to the TEXT prompt is passed to the foreground SCI to be displayed on your terminal when the batch stream has completed successfully, provided the batch stream is executed via the Execute Batch (XB) command.

Related Commands:

BATCH	(Begin Batch Execution)
XB	(Execute Batch)
XBJ	(Execute Batch Job)

EC (ERROR COUNT)

4.70 EC (ERROR COUNT)

Purpose:

The EC command counts the number of errors that occur during execution of commands in a batch stream. This command tests the value of synonym $$CC, and if nonzero, the error accumulation synonym EC is incremented by one and the current synonym value is written to the listing file.

The BATCH command resets the error accumulation synonym EC to zero at the beginning of the batch stream execution; at the termination of each foreground SCI command, $$CC is reset or set to a nonzero value representing the occurrence or nonoccurrence of errors.

When a nonzero $$CC is encountered, EC writes the current total error count (EC) to the BATCH LISTING FILE.

Example:

In the following example, the EC command counts errors encountered by commands in a batch stream and displays the Error Counter synonym, EC, to the batch listing file.

```
BATCH
 *
 *
 *
XMA
 *
 *
 *
EC
XMA
 *
 *
 *
EC
MSG TEXT=''ERROR COUNTER=@$E$C''
EBATCH
```

If the first XMA has no error, the first EC appears in the batch listing file as follows:

```
EC
```

If the second XMA does have an error, the second EC appears as follows:

```
EC
THE ERROR COUNT IS 1.
```

4.71 FB (FIND BYTE)

Prompts:

FIND BYTE

RUN ID:	integer	(*)
VALUE(S):	full exp list	
STARTING ADDRESS:	[full exp]	
ENDING ADDRESS:	[full exp]	

Purpose:

The FB command searches for the specified value or sequence of values in a memory area of a task. The search begins on a word boundary. If it finds the specified value(s), the corresponding memory address is displayed. If the task is not unconditionally suspended, it is temporarily suspended while the search is performed. The task then returns to its former state. An error is returned if the specified value(s) is not found.

NOTE

The Assign Breakpoint (AB) command replaces the contents of the memory address where the breakpoint is set with the value > 2FCF. You must keep this in mind when specifying the value(s) to be found in the search.

Prompt Details:

RUN ID:
A valid run ID in the user's job. Obtain current run IDs by executing the Show Task Status (STS) command.

VALUE(S):
The integer value or sequence of values to find in the memory area of the task.

STARTING ADDRESS:
The integer value that is the starting word address of the memory area to be searched. The default is zero.

ENDING ADDRESS:
The integer value that is the ending word address of the memory area to be searched. The default is end of task.

FB (FIND BYTE)

Example:

In the following example, the FB command searches the memory area from addresses >2F through >3AC of a task with the run ID of >10, for the following byte values: 6, 7, >A, 9, >A, >BC, 2, 1. The search stops when the values are found in eight contiguous bytes.

```
[] FB

FIND BYTE
                RUN ID: >10
             VALUE(S): 6,7,>A,9,>A,>BC,2,1
      STARTING ADDRESS: >2F
        ENDING ADDRESS: >3AC
```

Related Commands:

FW (Find Word)

4.72 FL (FORWARD SPACE LUNO)

Prompts:

```
FORWARD SPACE LUNO
                    LUNO:  integer
          RECORD COUNT:  integer
```

Purpose:

The FL command moves the current data-access position forward one or more records.

Prompt Details:

LUNO:
The logical unit number assigned to the sequential device or file to be forward spaced.

RECORD COUNT:
The number of records (from 1 through 65535, inclusive) that the sequential file or device is to be forward spaced. If the FL command encounters an end-of-file before forward spacing the number of records specified, it positions the sequential device or file at the beginning of the record following the end-of-file.

Example:

In the following example, the FL command forward spaces LUNO 32 three records:

```
[] FL

FORWARD SPACE LUNO
                  LUNO: 32
         RECORD COUNT: 3
```

Notes:

The operations performed by the FL command include an open call and a close call to the device.

Related Commands:

BL (Backspace LUNO)
RWL (Rewind LUNO)

FS (FIND STRING)

4.73 FS (FIND STRING)

After you activate the Text Editor, the FS command searches forward in the current file to locate a particular occurrence of a specified string. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the FS command and the following prompts:

```
FIND STRING
        OCCURRENCE NUMBER:  integer          (1)
             START COLUMN:  integer          (1)
               END COLUMN:  integer          (80)
                   STRING:  character(s)     (*)
```

4.74 FW (FIND WORD)

Prompts:

FIND WORD

RUN ID:	integer	(*)
VALUE(S):	full exp list	
STARTING ADDRESS:	[full exp]	
ENDING ADDRESS:	[full exp]	

Purpose:

The FW command searches for the specified value or sequence of values in a memory area of a task. The search begins on a word boundary. If it finds the specified value, the corresponding memory address is displayed. If the task is not unconditionally suspended, it is temporarily suspended while the search is performed. The task then returns to its former state. An error is returned if the specified value(s) is not found.

NOTE

The Assign Breakpoint (AB) command replaces the contents of the memory address where the breakpoint is set with > 2FCF. You must keep this in mind when specifying the value(s) to be found in the search.

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

VALUE(S):
The integer value or sequence of values to find in the memory area of the task.

STARTING ADDRESS:
The integer value which is the starting address of the memory area to be searched. If you specify an odd address, the address is rounded up to the nearest even value. The default is zero.

ENDING ADDRESS:
The integer value which is the ending address of the memory area to be searched. The default address is the end of task.

FW (FIND WORD)

Example:

In the following example, the FW command searches the memory area from addresses >AC to >CAE of a task with a run ID of >7F, for the following values: >80A, >80B, >80C, >80D. The search stops when these values are found in four contiguous words.

```
[] FW

FIND WORD
                  RUN ID: >7F
                  VALUES: >80A,>80B,>80C,>80D
        STARTING ADDRESS: >AC
          ENDING ADDRESS: >CAE
```

Related Commands:

FB (Find Byte)

4.75 HJ (HALT JOB)

Prompts:

HALT JOB
JOB NAME OR ID: {alphanumeric/integer} (*)

Purpose:

The HJ command suspends any job currently executing under your user ID. You can resume the job via the Resume Job (RJ) command. If the job is an interactive job, you must log on to another terminal (using the same user ID and a different job name) then issue the RJ command, or get the system operator to issue the RJ command.

Prompt Details:

JOB NAME OR ID:
The job name specified during job creation or the job ID returned when you execute the Execute Job (XJ) command. The XJ command is discussed later in this section. If a name is entered, only the first eight characters are used.

Example:

In the following example, the user halts a job named PRINT, currently executing under the user's ID, by entering the HJ command as follows:

```
[] HJ

HALT JOB
        JOB NAME OR ID: PRINT
```

Assumptions:

The job being halted is running under the same user ID as the job from which the request is made, or the request is being made by the system operator.

Notes:

It is possible for two diferent users to use the same job name. In that case, the system operator must specify the job by ID when issuing the HJ command.

If other terminals are reconnected to the job in which you are running, an attempt to halt your job will also halt the users of the reconnected terminals. At least one terminal must not be running in a halted job; otherwise, restarting the halted job(s) will be impossible.

HJ (HALT JOB)

Related Commands:

KJ	(Kill Job)
LJ	(List Jobs)
MJP	(Modify Job Priority)
RJ	(Resume Job)
SJS	(Show Job Status for < user ID>)
XBJ	(Execute Batch Job)
XJ	(Execute Job)

4.76 HO (HALT OUTPUT AT DEVICE)

Prompts:

```
HALT OUTPUT AT DEVICE
        DEVICE OR CLASS NAME:  [site]{devicename/character(s)}@    (*)
    IMMEDIATELY OR AT EOF?:    [I/E/A]                             (IMMEDIATELY)
```

Purpose:

The HO command temporarily halts queued output (for example, to change forms at a line printer) to a device or for a class name and includes the option of specifying when the output is to be halted.

Use the Resume Output (RO) command to release halted output and to allow the output to continue printing. The RO command is discussed later in this section.

Prompt Details:

DEVICE OR CLASS NAME:
Specify the name of the device at which you are halting queued output (for example, LP01) or specify the class (group of related devices) for which you are halting output (for example, FAST).

IMMEDIATELY OR AT EOF?:
If you enter I (IMMEDIATELY) in response to this prompt, the output is to be halted immediately. If you enter E (EOF) or A (AT EOF), the output is halted after the end-of-file is encountered. If your response to the DEVICE OR CLASS NAME prompt is a class name, your response to the IMMEDIATELY OR AT EOF? prompt is ignored and any output currently printing continues to print until end-of-file.

Example:

In the following example, the HO command immediately halts the output being printed at the line printer LP01:

```
[] HO

HALT OUTPUT AT DEVICE
      DEVICE OR CLASS NAME: LP01
   IMMEDIATELY OR AT EOF?: IMMEDIATELY
```

HO (HALT OUTPUT AT DEVICE)

Notes:

When you use the HO command to halt output for a class, only the output which has been designated for that particular class is halted. Any output not queued for that predefined class would be allowed to print.

This command affects only output being performed by the spooler. If other programs are writing directly to the printer, this command has no effect on those programs.

Related Commands:

KO	(Kill Output at Device)
MO	(Modify Output)
MSD	(Modify Spooler Device)
RO	(Resume Output at Device)
SOS	(Show Output Status)

4.77 HT (HALT TASK)

Prompts:

HALT TASK

RUN ID: integer (*)

Purpose:

The HT command unconditionally suspends a task at the end of the current time slice. If the task is already unconditionally suspended, this command has no effect on the task. If the task is not in the active state, the HT command waits five seconds for the task to reach unconditional suspend, then gives you the option of aborting or continuing to wait.

If the task cannot be suspended, the following message is displayed:

UNABLE TO SUSPEND TASK. CURRENT STATE = XX. WAIT FOR TASK TO SUSPEND?

If you enter a YES response, another attempt is made to suspend the task. If unsuccessful, the message is displayed again. A NO response to the preceding message causes the following message to be displayed:

CURRENT STATE = XX. EXECUTE COMMAND WITHOUT SUSPENDING?

If you enter a YES response, the debugger will attempt to perform the specified debug command without waiting for the task to suspend. In this case, the task will not be allowed to become active. A NO response to the preceding message causes the following message to be displayed:

DO YOU WISH TO LEAVE SUSPENSION PENDING?

A YES response leaves the suspension pending, while a NO response terminates the suspension attempt.

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

HT (HALT TASK)

Example:

In the following example, the HT command suspends the task with the run ID of >7.

```
[] HT

HALT TASK
                RUN ID:  >7
```

Assumptions:

The task to be suspended is not in a suspended state. If the task to be halted is a system task, you must have a user ID with a privilege level of 2 or higher.

Related Commands:

AT	(Activate Task)
RT	(Resume Task)
XD	(Execute in Debug Mode)

4.78 IBMUTL (IBM DISKETTE CONVERSION UTILITY)

Prompts:

The prompts for the IBMUTL command are displayed in TTY mode format (one line at a time), even if the terminal is in VDT mode, as follows:

```
IBMUTL                  IBM DISKETTE CONVERSION UTILITY

                    COUNTRY NAME(USA):   character(s)
                      SELECTED COUNTRY:  character(s)
                    DISKETTE DRIVE NAME: devicename@
SERVICE?F = FORMAT.T = TRANSFER:         {F/T}
```

If the response to the SERVICE?F = FORMAT.T = TRANSFER prompt is F, the following prompt is displayed:

ARE YOU SURE? Y/N: YES/NO

If you enter Y (YES), the diskette is formatted. If you enter N (NO), the following prompt is displayed:

SERVICE?F = FORMAT.T = TRANSFER

If your response to the SERVICE?F = FORMAT.T = TRANSFER prompt is T, the following prompt is displayed:

ARE THESE RPGII FILES? Y/N: YES/NO

After you enter a response for the previous prompt, the following prompts are displayed (one line at a time):

FUNCTION?F = FILE TO DATASET,D = DATASET TO FILE: {F/D}
CHARACTERS PER RECORD?2 MIN. – 128 MAX: integer

If you enter a D in response to the FUNCTION? prompt, the following prompt is displayed:

ARE THERE PACKED DECIMAL DATA FIELDS?Y/N: Y/N

If the response to ARE THERE PACKED DECIMAL DATA FIELDS?Y/N is Y (YES), the following two prompts are displayed on the user's terminal:

SPECIFY STARTING POSITION IN RECORD: integer
SPECIFY ENDING POSITION IN RECORD: integer

These prompts continue to be displayed on your terminal until you enter a null response for the SPECIFY STARTING POSITION IN RECORD prompt.

IBMUTL (IBM DISKETTE CONVERSION UTILITY)

The following prompts are displayed if an F response is entered for the FUNCTION? prompt or after responses are entered for the prompts displayed if D is entered for the FUNCTION? prompt:

```
                    TI FILE PATHNAME:  pathname@
TI FILE PHYSICAL RECORD LENGTH:  integer
                    IBM DATASET NAME:  character(s)
```

When reading multivolume diskettes, the following prompt is displayed at the end of any volume expecting a continuation of another volume:

```
MOUNT NEXT VOLUME, REPLY "C" WHEN READY:  C
```

Purpose:

The IBMUTL command transfers standard IBM-formatted diskette datasets to DNOS files and DNOS files to standard IBM-formatted diskette datasets. IBMUTL also formats diskettes to standard IBM specifications for single-density diskettes (as designated in The IBM Diskette General Information Manual, GA21-9182, which may be ordered from IBM). IBMUTL works only with single-density diskettes which can be inserted in a double-density diskette drive (FD1000).

The IBMUTL command allows you to read or write datasets on an IBM-formatted diskette used by systems and devices based on IBM sequentially-sectored diskettes that use the EBCDIC character set. The IBM-formatted diskette may already contain datasets created by another process or may have been newly formatted by this utility or by other means. All preexisting datasets are preserved.

The basic exchange datasets supported by DNOS have requirements that assure that diskettes can be exchanged between systems. These requirements are as follows:

- Exchange indicator (dataset label position 44) must be blank.
- Dataset is organized sequentially.
- Records are a maximum of 128 bytes long.
- Records are of a fixed length, unblocked, and unspanned.
- Dataset identifier (dataset label positions 6 – 22) is not longer than eight positions.
- Diskette is initialized with physically sequential records (volume label positions 77 – 78 equal blanks or 01).
- Basic exchange datasets use only tracks 1 – 73.
- An alternative physical record relocation is not used in a basic exchange dataset.

IBMUTL (IBM DISKETTE CONVERSION UTILITY)

Entering the F (format) response to the SERVICE? prompt causes a diskette to be formatted to IBM format. If any bad tracks are found on a diskette used in an FD1000 diskette drive, IBMUTL aborts. You can use only diskettes with all good tracks in an FD1000 diskette drive. Track 0 contains the dataset headers (sectors 8-26) and other information about the diskette (sectors 1-7). The dataset headers are written to include name, record length, beginning of extent (BOE), end of extent (EOE), and end of data (EOD) fields only. All others are left in the initialized state (blank).

You must specify DNOS files that are to be converted to IBM format with a standard DNOS pathname. The new dataset begins with the first available label following the last used label in the IBM diskette directory. Empty labels between used labels are skipped by this utility. The name of the new dataset can be the same as an already existing dataset, but the existing dataset will be replaced. If the new dataset is shorter than the existing dataset, the EOD and EOE are modified to reflect the change. If the new dataset is longer than the already existing dataset, the following message is displayed:

TOO MUCH DATA

Multivolume datasets cannot be created by this utility on IBM diskettes.

To transfer IBM datasets to DNOS files, you must specify the desired dataset label and the DNOS pathname. The dataset labels from the IBM diskette are displayed when you enter the transfer command. If the DNOS file does not already exist, it is created as a noncontiguous sequential file. Multivolume datasets can be transferred from IBM diskettes to DNOS disks.

There are two special characters recognized by IBMUTL, as follows:

Character	Definition
*	When entered in response to a prompt and followed by a carriage return, IBMUTL is terminated.
&	When entered in response to a prompt and followed by a carriage return, IBMUTL is restarted at the DISKETTE DRIVE NAME prompt.

Table 4-8 lists the errors which may be encountered during execution of IBMUTL. Whenever recovery from such errors is possible, IBMUTL returns to a logical restart point and continues execution.

IBMUTL (IBM DISKETTE CONVERSION UTILITY)

Prompt Details:

COUNTRY NAME(USA):

If the files were created in a country other than the United States, this prompt allows you to specify the name of the country. The conversion tables are modified to correspond to the EBCDIC codes for that country. You must terminate the utility, then restart, to specify a new country.

Legal country names and abbreviations that can be specified are as follows:

Country Name	Abbreviation
Austria	AU
Belgium	B
Denmark	D
Finland	FI
France	F
French Word Processing	FWP
Germany	G
Japan	J
Norway	N
Spain	SP
Sweden	S
Switzerland	SWIS
United Kingdom	UK
United States	US

The default value, US, should be entered for any country not listed above.

SELECTED COUNTRY:

No response is necessary for this prompt. IBMUTL displays this prompt and the country specified as the response to the COUNTRY NAME(USA) prompt.

DISKETTE DRIVE NAME:

The device name of the device which contains the IBM diskette.

SERVICE?F = FORMAT. T = TRANSFER:

If you enter F in response to this prompt, the formatting function is initiated. You should select this function only when first writing on the diskette. If you enter T, the transfer function is initiated.

ARE YOU SURE(Y/N):

If you specify Yes (Y), the conversion proceeds. If you specify No (N), the SERVICE?F = FORMAT. T = TRANSFER prompt is displayed on your terminal.

ARE THESE RPGII FILES? Y/N:

If you enter YES, the files being converted are used by RPG, which requires special conversion characters. Otherwise, you should enter NO.

IBMUTL (IBM DISKETTE CONVERSION UTILITY)

FUNCTION?F = FILE TO DATASET,D = DATASET TO FILE:

If you enter F, the file-to-dataset function is initiated, which implies a conversion from DNOS file to IBM diskette file. If you enter D, the dataset-to-file function is initiated, which implies a conversion from IBM diskette to DNOS file.

CHARACTERS PER RECORD? 2 MIN. – 128 MAX:

The integer value that is the number of characters per record output to the DNOS file or IBM diskette file, depending on the response to the FUNCTION? prompt. The default value is 80.

ARE THERE PACKED DECIMAL DATA FIELDS?Y/N:

This prompt appears only if you specified the DATASET TO FILE function. If you enter YES, the transfer contains packed decimal data fields; otherwise, enter NO. The YES response inhibits EBCDIC to ASCII conversion in the specified positions of each record transferred.

SPECIFY STARTING POSITION IN RECORD:
SPECIFY ENDING POSITION IN RECORD:

These prompts appear only if you enter YES in response to the ARE THERE ANY PACKED DECIMAL DATA FIELDS?Y/N prompt. The leftmost (starting) and rightmost (ending) character positions of the packed decimal field are entered. These prompts continue to be displayed until you enter a null value for the SPECIFY STARTING POSITION IN RECORD prompt. You can specify a maximum of 20 fields.

The field(s) selected are not converted by IBMUTL and binary fields are specified. Numeric values for the field positions in the range of 1 through 128, inclusive, are permitted, and numbers need not be justified. Also, leading and trailing blanks in the field are accepted.

TI FILE PATHNAME:

The file name of the DNOS file to be used in the conversion, which can have a maximum length of 26 characters. Synonym values are not accepted.

TI FILE PHYSICAL RECORD LENGTH:

This prompt appears only if you specify the DATASET TO FILE function. The TI file is created with the physical record length specified, or the default physical record length of 256 bytes can be taken.

IBM DATASET NAME:

The character(s) that identifies the diskette dataset label of the diskette to be used in the conversion, with a maximum length of eight characters.

MOUNT NEXT VOLUME, REPLY "C" WHEN READY:

When reading multivolume diskettes, this prompt appears at the end of any volume expecting a continuation of another volume. Enter the C response only after you have mounted the next diskette. (You cannot create multivolume diskettes using the IBMUTL command.)

Many of the preceding prompts will be displayed after the new diskette has been mounted and the conversion continues.

IBMUTL (IBM DISKETTE CONVERSION UTILITY)

Example:

The following examples discuss the formatting of a diskette to IBM format and the FILE TO DATASET and DATASET TO FILE functions.

The first example shows the responses to format a diskette to IBM format. In the second example, a DNOS file is converted and transferred to an IBM diskette file with the maximum value 128 specified for the CHARACTERS PER RECORD prompt.

In the third example, an IBM diskette file is converted and transferred to a DNOS file. The maximum response to the CHARACTERS PER RECORD prompt is taken, and the starting and ending positions of a packed decimal field to be transferred are specified.

In all of the examples, messages which will appear at certain points throughout the conversion are shown. Refer to the Messages portion of this command for an explanation of these messages.

The following example formats a diskette to IBM format:

```
[] IBMUTL

IBMUTL              IBM DISKETTE CONVERSION UTILITY

COUNTRY NAME(USA): US
SELECTED COUNTRY : United States

DISKETTE DRIVE NAME: DS04
SERVICE?F=FORMAT.T=TRANSFER: F
ARE YOU SURE? Y/N: Y
FORMAT IN PROGRESS...
FORMAT COMPLETE
SERVICE?F=FORMAT.T=TRANSFER: *
UTILITY SERVICE TERMINATED
```

IBMUTL (IBM DISKETTE CONVERSION UTILITY)

The following example transfers a DNOS file to a diskette dataset:

```
IBMUTL              IBM DISKETTE CONVERSION UTILITY

COUNTRY NAME(USA): US
SELECTED COUNTRY : United States

DISKETTE DRIVE NAME: DS04
SERVICE?F=FORMAT.T=TRANSFER: T
ARE THESE RPGII FILES?Y/N: Y
FUNCTION?F=FILE TO DATASET,D=DATASET TO FILE: F
CHARACTERS PER RECORD? 2 MIN. -128 MAX: 128
TI FILE PATHNAME: DS03.USER33.CONVERT
TI FILE PHYSICAL RECORD LENGTH: 256
IBM DATASET NAME: SAMPLE
FILE TRANSFER IN PROGRESS...
TRANSFER COMPLETE

SERVICE?F=FORMAT.T=TRANSFER: *
UTILITY SERVICE TERMINATED
```

The following example transfers a diskette dataset to a DNOS file:

```
IBMUTL              IBM DISKETTE CONVERSION UTILITY

COUNTRY NAME(USA): US
SELECTED COUNTRY : USA

DISKETTE DRIVE NAME: DS04
SERVICE?F=FORMAT.T=TRANSFER: T
ARE THESE RPGII FILES?Y/N: N
FUNCTION?F=FILE TO DATASET,D=DATASET TO FILE: D
CHARACTERS PER RECORD? 2 MIN.-128 MAX: 128
ARE THERE ANY PACKED DECIMAL DATA FIELDS?Y/N: Y
SPECIFY STARTING POSITION IN RECORD: 81
SPECIFY ENDING POSITION IN RECORD: 86
SPECIFY STARTING POSITION IN RECORD:
TI FILE PATHNAME: DS03.USER33.CONVERT
TI PHYSICAL RECORD LENGTH: 288
IBM DATASET NAME: SAMPLE
FILE TRANSFER IN PROGRESS...
TRANSFER COMPLETE

SERVICE?F=FORMAT.T=TRANSFER: *
UTILITY SERVICE TERMINATED
```

IBMUTL (IBM DISKETTE CONVERSION UTILITY)

Assumptions:

A valid device name is specified for the DISKETTE DRIVE NAME prompt, and the correct TI file pathname is specified.

Messages:

The following messages are displayed on your terminal at certain points in the conversion process:

DISKETTE NOT IBM FORMAT
: This message appears if the diskette is not an IBM-formatted diskette. Proper format is verified when listing dataset labels. The DISKETTE DRIVE NAME prompt is again displayed.

FILE TRANSFER IN PROGRESS...
: This message indicates that a transfer of data is in progress.

FORMAT IN PROGRESS...
FORMAT COMPLETE
: These two messages appear only if you have selected the FORMAT process. The DISKETTE DRIVE NAME prompt is again displayed.

IBMUTL-IBM DISKETTE CONVERSION UTILITY
: This is the first message to appear when IBMUTL execution begins.

SELECTED COUNTRY:
: The name specified in response to the COUNTRY NAME(USA) prompt is displayed. (The specified country name controls the character sets used in converting the files.)

TRANSFER COMPLETE
: This message indicates successful completion of the transfer function. The SERVICE? prompt is then displayed for the next dataset.

UTILITY SERVICE TERMINATED
: This message appears if you respond to any prompt with an asterisk (*) indicating that no more datasets are to be transferred.

IBMUTL (IBM DISKETTE CONVERSION UTILITY)

Table 4-8. IBMUTL Error Messages

Message	Description
DATASET NAME NOT FOUND	Meaning: Specified dataset name not found on diskette. Recovery: Program control returns to the dataset name prompt. Respond with the correct dataset name.
DISKETTE ACCESS ERROR nn	Meaning: An error was encountered while accessing an IBM dataset. Refer to the *DNOS Messages and Codes Reference Manual* for an SVC internal message code 00nn. Recovery: Program control returns to the diskette drive prompt. Respond according to the specified error code.
DISKETTE DIRECTORY FULL	Meaning: An attempt to exceed the maximum number of datasets (19) allowable per IBM- formatted diskette. Recovery: Program control returns to the diskette drive prompt. Install new IBM-formatted diskette and retry transfer.
DISKETTE NOT IBM FORMAT	Meaning: Specified diskette is not IBM-formatted diskette. Recovery: Insert a properly formatted diskette in the specified drive, or return to diskette name prompt (enter &) and input correct drive name.
DISKETTE NOT SINGLE DENSITY	Meaning: The specified diskette is not single density. Recovery: Insert a single-density diskette in the drive. IBMUTL supports only single-sided, single-density diskettes.
FATAL I/O ERROR	Meaning: An unrecoverable read/write was encountered, and IBMUTL was terminated. Recovery: Execute IBMUTL again.

IBMUTIL (IBM DISKETTE CONVERSION UTILITY)

Table 4-8. IBMUTL Error Messages (Continued)

Message	Description
FILE SERVICE ERROR nn	Meaning: An error has been encountered while accessing a DNOS user file. Refer to the *DNOS Messages and Codes Reference Manual* for an SVC internal message code 00nn. Recovery: Program control returns to the diskette drive prompt. Respond according to the specified error code.
IBM DISKETTE IS FULL	Meaning: Data capacity of IBM-formatted diskette has been exceeded. The dataset being written to contains as much data as could be transferred. Also occurs when the file-to-dataset function is selected and the diskette is already full. Recovery: Program control returns to the diskette drive prompt. Multivolume diskettes cannot be produced using the IBMUTL utility.
I/O ERROR nn	Meaning: An I/O error was encountered during program execution. Refer to the *DNOS Messages and Codes Reference Manual* for an SVC internal message code 00nn. Recovery: Program control returns to the diskette drive prompt. Respond according to the specified error code.
PATHNAME UNDEFINED	Meaning: Illegal pathname has been entered. Recovery: Validate pathname and reenter.
TOO MANY BAD TRACKS – FORMAT ABORTED	Meaning: Bad diskette. Recovery: Program control returns to the diskette drive prompt. Install a new diskette and retry format function.

Table 4-8. IBMUTL Error Messages (Continued)

Message	Description
TOO MUCH DATA	Meaning: The IBM diskette is not a basic exchange diskette. The dataset to be transferred extends past track 73. Recovery: None. The IBMUTL utility only handles basic exchange datasets.
WRONG DEVICE TYPE	Meaning: A device other than a disk device (DSxx) was specified in response to the DISKETTE DRIVE NAME prompt. Recovery: Execute IBMUTL again and identify a disk device in response to the DISKETTE DRIVE NAME prompt.

IDS (INITIALIZE DISK SURFACE)

4.79 IDS (INITIALIZE DISK SURFACE)

Prompts:

```
INITIALIZE DISK SURFACE
                    UNIT NAME:  devicename@
  CONTINUE SUSPENDED IDS?:  YES/NO                                (NO)
    INITIALIZE NEW VOLUME?:  YES/NO                               (YES)
      LISTING ACCESS NAME:  [site:][pathname]@
     EXECUTION MODE(F,B):  {F/FOREGROUND/B/BACKGROUND}  (FOREGROUND)
```

If you enter NO in response to the CONTINUE SUSPENDED IDS? prompt, the following prompts are displayed:

```
INITIALIZE DISK SURFACE
    LENGTH OF ANALYSIS(S,M,L):  [S/SHORT/M/MEDIUM/L/LONG]  (LONG)
      MARK MARGINAL TRACKS?:  [YES/NO]
      BAD TRACK ACCESS NAME:  [site:][pathname]@
HARDWARE INTERLEAVING FACTOR:  [integer]
```

If you enter YES in response to the INITIALIZE NEW VOLUME? prompt, the following prompts are displayed:

```
INITIALIZE NEW VOLUME
                VOLUME NAME:  alphanumeric@
NUMBER OF VCATALOG ENTRIES:  [integer]
DEFAULT PHYSICAL RECORD SIZE:  [integer]
        USED AS SYSTEM DISK?:  YES/NO                           (YES)
```

Purpose:

The IDS command performs an analysis of the disk surface, checking for flaws in the media and marking any bad tracks that are found. This command does not check the disk drive or controller. IDS may not find all tracks that are bad; therefore, you *must* enter any bad tracks listed on the disk label by the manufacturer. Any bad track information that is on the disk when the surface is initialized is in the list of bad tracks which you specify. Any other information on the disk when the IDS command is executed is unrecoverable.

If your disk type supports bad track mapping, bad tracks will be mapped to alternate tracks by IDS. This mapping is transparent to the user and gives the impression of error free disks. If your disk type does not support bad track mapping, the bad tracks are marked for avoidance. DNOS file management will not allocate file space on the tracks marked bad.

IDS (INITIALIZE DISK SURFACE)

All disks must have their surface initialized before they can be used. Disks supplied by Texas Instruments (except for flexible diskettes, DS10s, and DS31s) are preinitialized and need not be reinitialized. Once the surface is initialized, the disk is ready for volume initialization using the Initialize New Volume (INV) command. Alternatively, the IDS command can perform both the surface and volume initialization. Neither the IDS nor the INV command executes if the disk to be initialized is installed. If the disk is installed, execute the Unload Volume (UV) command prior to executing either the IDS or INV command. If you issue the Show Volume Status (SVS) command while IDS is executing, a series of dollar signs ($) is displayed in the NAME INSTALLED field of the SVS listing. This indicates that an uninstalled volume is in use by IDS.

The time required for the IDS utility to complete depends upon the type of disk being initialized and upon the amount of other system activities. If IDS is executed in foreground with the terminal in VDT mode, a regularly updated graph is displayed on your screen which shows the percentage of the analysis completed. Throughout the IDS procedure, you can suspend the analysis and resume it at a later time. To suspend IDS when it is executing in foreground mode, use the hard break key sequence as follows: press the Attention key, release it, and hold down the Control key while you press the X key. To suspend IDS when it is executing in background mode, use the KILL BACKGROUND TASK (KBT) command. To continue a suspended IDS operation issue the IDS command and answer YES to the CONTINUE SUSPENDED IDS? prompt.

The IDS command writes a message to the system log specifying the user ID, the station name, volume name, and unit name associated with the disk being initialized.

Prompt Details:

UNIT NAME:
The device name of the disk unit that contains the volume with a surface to be initialized.

CONTINUE SUSPENDED IDS?:
Enter YES to continue a suspended IDS operation. IDS can be continued after being terminated by execution of the hard break key sequence, execution of the KBT command, or a system crash.

INITIALIZE NEW VOLUME?:
If you enter YES, the volume is analyzed and initialized. If you enter NO, the disk surface is analyzed.

LISTING ACCESS NAME:
The pathname of the file or device to which IDS should list the results of the disk surface initialization. The listing contains the head and cylinder location of all bad tracks found on the disk. The listing also indicates whether bad tracks have been mapped to alternate tracks or marked for avoidance. A null response specifies the terminal local file.

Some disks have reserved tracks that are used for diagnostics or bad track mapping. If any of these tracks are bad, the listing indicates that the track is reserved for special functions. Bad track mapping and diagnostics will still function normally when a reserved track is bad.

IDS (INITIALIZE DISK SURFACE)

EXECUTION MODE(F,B):

If you enter F or FOREGROUND, the command executes in foreground mode. If you enter B or BACKGROUND, the command executes in background mode. The initial value is FOREGROUND. However, in a batch stream, this command always executes in background mode.

LENGTH OF ANALYSIS(S,M,L):

Enter S (SHORT), M (MEDIUM), or L (LONG) to indicate the length of surface analysis. In general, the more time spent on the surface analysis, the more accurate the results. In some cases, the increase in accuracy attained by the long analysis is unnecessary for the intended use of the disk.

MARK MARGINAL TRACKS?:

Enter YES to make the criteria used for determining bad tracks less tolerant. The additional tracks marked bad are those which have a small percentage of recoverable errors. These relatively rare errors may be caused by disk surface imperfections or random atmospheric electrical activity. Since the actual increase in the number of tracks marked bad is small, you need only enter YES when the integrity of the data on the disk is extremely critical or if you are encountering numerous disk error log messages and wish to eliminate them. If you enter SHORT or MEDIUM in response to the LENGTH OF ANALYSIS(S,M,L) prompt, it is recommended that you answer YES to MARK MARGINAL TRACKS? prompt.

BAD TRACK ACCESS NAME:

The pathname of the file or device that is to supply a list of known bad tracks. The default is a null response which indicates there is no list of known bad tracks. If you enter ME in response to this prompt, SCI prompts for the list of known bad tracks as follows:

```
ENTER BAD TRACKS IN THE FORMAT:

 HEAD, CYLINDER;
       OR
 HEAD, CYLINDER; HEAD, CYLINDER; ETC.

TO END LIST, ENTER AN EMPTY LINE
```

Enter the list of known bad tracks in the requested format using decimal notation (for example, 2, 235; 0,15;). If you enter a pathname in response to the BAD TRACK ACCESS NAME prompt, the file must contain the list of known bad tracks in the same HEAD, CYLINDER; format (for example, 2, 235; 0, 15;). All entries, including the last, must end with a semicolon. Improper positioning or absence of the punctuation marks causes an error to be returned.

HARDWARE INTERLEAVING FACTOR:

The interleaving factor used for double-density diskettes. For hard disk (DS10, DS50, and so on) the interleaving factor is ignored. The default is the interleaving factor for your disk type.

Refer to the INV command for information concerning responses to the additional prompts that appear if you enter YES in response to the INITIALIZE NEW VOLUME? prompt.

IDS (INITIALIZE DISK SURFACE)

Example:

In the following example, the IDS command analyzes the surface of the disk mounted in device DS02. The list of bad tracks is displayed on the terminal local file.

```
[]IDS

INITIALIZE DISK SURFACE
                    UNIT NAME: DS02
     CONTINUE SUSPENDED IDS?: NO
      INITIALIZE NEW VOLUME?: NO
         LISTING ACCESS NAME:
         EXECUTION MODE(F,B): FOREGROUND


 INITIALIZE DISK SURFACE
   LENGTH OF ANALYSIS(S,M,L): MEDIUM
        MARK MARGINAL TRACKS: NO
       BAD TRACK ACCESS NAME: .BADTRK
HARDWARE INTERLEAVING FACTOR: 1
```

Notes:

The IDS operation takes a considerable length of time to execute. You can execute several IDS commands simultaneously as long as each command specifies a unique disk unit. If you specify the INITIALIZE NEW VOLUME prompt, a unique volume name is also required.

We recommend that you analyze the disk surface with the disk mounted in the drive in which it will normally be used.

After it starts, you should not interrupt IDS for 10 minutes or until the first graph display update, whichever occurs first. If IDS is interrupted before the analysis of cylinder 0 is complete, the bad track list could be lost.

If the disk being analyzed is on the same controller as the system disk, system performance will be degraded considerably because of the IDS activity.

Four additional prompts are available in expert or batch mode. Details of these prompts are as follows:

DELETE BAD TRACK LIST?:

Enter YES to delete the existing list of known bad tracks. The list of known bad tracks is deleted from the disk being initialized and is written to the file or device specified in the LISTING ACCESS NAME prompt. This feature allows you to ignore a list of bad tracks which contains incorrect information as the result of a system hardware malfunction. If you specify YES to DELETE BAD TRACK LIST? prompt, you should have IDS perform the long surface analysis (enter L to the LENGTH OF ANALYSIS prompt).

IDS (INITIALIZE DISK SURFACE)

RESTORE BAD TRACK LIST?:

Enter YES to restore the list of known bad tracks. If the list of known bad tracks on an initialized disk is destroyed, but the disk still has a good diagnostic track, you can use this feature to restore the list of known bad tracks. This avoids the necessity of repeating the full IDS procedure. You can also enter the list of known bad tracks through the BAD TRACK ACCESS NAME prompt. If you use this feature, any data on the disk is lost, and the INV command must be executed.

DATA PATTERN 1:

If the manufacturer of the disk specifies a particular data pattern for the disk analysis, enter the four-digit hexadecimal value. If you enter a null response, IDS uses the standard data pattern for your disk type.

DATA PATTERN 2:

If the manufacturer of the disk specifies a particular data pattern for the disk analysis, enter the four-digit hexadecimal value. If you enter a null response, IDS uses the standard data pattern for your disk type.

Related Commands:

INV (Initialize New Volume)

IDT (INITIALIZE DATE AND TIME)

4.80 IDT (INITIALIZE DATE AND TIME)

Prompts:

INITIALIZE DATE AND TIME

YEAR:	integer
MONTH:	{character(s)/integer}
DAY:	integer
HOUR:	integer
MINUTE:	integer

Purpose:

The IDT command reinitializes the system date and time at any time. Specify the year, month, day, hour, and minute and upon receipt of the command, the system will start counting the seconds at zero.

Prompt Details:

YEAR:
Two or four numeric digits representing the current year. For example, the digits 84 or 1984 may be entered.

MONTH:
One or two numeric digits representing the current month or any unique alphabetic characters of the month. For example, January can be listed as JAN, but not JY, since it may be confused with July.

DAY:
One or two numeric digits representing the current day.

HOUR:
One or two numeric digits representing the current hour (according to the 24-hour clock).

MINUTE:
One or two numeric digits representing the current minute.

Example:

In the following example, the IDT command will initialize the system time and date for 1:34 p.m. on June 5, 1982.

```
[] IDT

INITIALIZE DATE AND TIME
              YEAR: 1982
             MONTH: 6
               DAY: 5
              HOUR: 13
            MINUTE: 34
```

IF (INSERT FILE)

4.81 IF (INSERT FILE)

After you activate the Text Editor, the IF command causes a copy of the contents of a different file to be inserted into the current file being edited. It is inserted after the specified line (number). Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the IF command and the following prompts:

```
INSERT FILE
        INSERT AFTER LINE:  [BEGINNING/END/line number]
            FILE PATHNAME:  [site:]filename@
```

4.82 IGS (INSTALL GENERATED SYSTEM)

Prompts:

```
INSTALL GENERATED SYSTEM
        TARGET DISK/VOLUME:  {devicename/alphanumeric}    (*)
              SYSTEM NAME:   alphanumeric                 (*)
```

Purpose:

After testing the system by using the Test Generated System (TGS) command, you can use the IGS command to install it. The TGS command is described later in this section.

The following procedure includes the steps to test and install a generated system:

1. Enter the TGS command to SCI:

   ```
   [ ]TGS

   TEST GENERATED SYSTEM
        TARGET DISK/VOLUME:  {devicename/alphanumeric}    (*)
              SYSTEM NAME:   alphanumeric                 (*)
   ```

   where the response to the SYSTEM NAME prompt is the name assigned to the new system in response to the OUTPUT CONFIGURATION prompt in the Execute System Generation Utility (XSGU) command.

2. Perform an initial program load (IPL) using the new operating system. If the new system fails to operate, you can reload the old system using the same IPL procedure.

3. If the new system includes all of the devices specified during system generation, you can install it as the primary operating system by entering the IGS command:

   ```
   [ ]IGS

   INSTALL GENERATED SYSTEM
        TARGET DISK/VOLUME:  {devicename/alphanumeric}    (*)
              SYSTEM NAME:   alphanumeric                 (*)
   ```

4. If the new system is not correct or satisfactory, you can reload the original system software kit using the IPL procedure. The new system can be completely regenerated or changed using the system generation commands. Refer to the *DNOS System Generation Reference Manual* for a detailed description on generating or changing a new system.

IGS (INSTALL GENERATED SYSTEM)

Prompt Details:

TARGET DISK/VOLUME:
The device or volume name that contains the newly generated system.

SYSTEM NAME:
A one- to eight-character alphanumeric string that is the name of the system that was generated. The response to this prompt should be the same as the OUTPUT CONFIGURATION prompt response in the Execute System Generation Utility (XSGU) command.

Example:

In the following example, a new system has been generated and is contained in disk drive DS01. The name specified for the system is SYS1. The TGS command tests the new system with the following prompt responses:

```
[] TGS

TEST GENERATED SYSTEM
     TARGET DISK/VOLUME: DS01
            SYSTEM NAME: SYS1
```

If the system operates correctly after the IPL, you can install the new system as the primary system using the IGS command as follows:

```
[] IGS

INSTALL GENERATED SYSTEM
     TARGET DISK/VOLUME: DS01
            SYSTEM NAME: SYS1
```

Related Commands:

ALGS	(Assemble and Link Generated System)
PGS	(Patch Generated System)
TGS	(Test Generated System)

4.83 INV (INITIALIZE NEW VOLUME)

Prompts:

```
INITIALIZE NEW VOLUME
                          UNIT NAME:  devicename@
                        VOLUME NAME:  alphanumeric@
      NUMBER OF VCATALOG ENTRIES:     [integer]
     DEFAULT PHYSICAL RECORD SIZE:    [integer]
  HARDWARE INTERLEAVING FACTOR:       [integer]
           FORCE CLEARING OF DISK?:   [YES/NO]                (NO)
             USED AS SYSTEM DISK?:    YES/NO                  (YES)
              LISTING ACCESS NAME:    [site:][pathname]@
           EXECUTION MODE(F,B):       {F/FOREGROUND/B/BACKGROUND}(FOREGROUND)
```

Purpose:

The INV command formats a disk volume for use under the operating system. Its primary function is to create the primary directory of the disk (.VCATALOG), assign a volume name to the volume, and install the volume (the INV command will not execute if the volume is installed). Disk volume names must not be the same as disk device names (for example, DS01, DS02). If you enter null responses for NUMBER OF VCATALOG ENTRIES, DEFAULT PHYSICAL RECORD SIZE, or HARDWARE INTERLEAVING FACTOR, proper default values are assigned according to the disk type.

Before you execute the INV command, you must execute the Initialize Disk Surface (IDS) command to check the disk for bad tracks. Once the IDS command has initialized the surface of a disk, you can use the INV command as often as you want. All bad track information is saved when the INV command is performed.

The FORCE CLEARING OF DISK? option allows you to ensure that any sensitive information on the disk is totally destroyed. Normally, INV deletes all files from the file structure, but the file contents still exist on the disk.

The INV command writes a message to the system log specifying the user ID, the station name, volume name, and unit name associated with the disk being initialized.

Prompt Details:

UNIT NAME:
The device name of the disk unit that contains the volume to be initialized.

VOLUME NAME:
A one- to eight-character string (the first character of the string must be alphabetic) that is to identify the disk cartridge. Several disk cartridges can have the same volume name; however, you can only install one on the system at a time.

INV (INITIALIZE NEW VOLUME)

NUMBER OF VCATALOG ENTRIES:
The maximum number of entries (channels, directories, and files) in the master volume directory (VCATALOG). The number you enter is rounded up to the nearest prime number. The maximum number of entries that can be specified is 65,521. If you leave this blank, a default value based on your disk type is used.

DEFAULT PHYSICAL RECORD SIZE:
The physical record size to be used in all subsequent file creations on the disk. If you leave this blank, a default value based on your disk type is used.

HARDWARE INTERLEAVING FACTOR:
The interleaving factor used for double-density diskettes. For hard disk (DS10, DS50, and so on) the interleaving factor is ignored. If you leave this blank, a default value based on your disk type is used.

FORCE CLEARING OF DISK?:
Enter YES to ensure all information on the disk is totally destroyed; otherwise, enter NO. If you enter YES, the INV command takes considerably longer to execute than if you enter NO. This should usually be answered NO. The default value is NO.

USED AS SYSTEM DISK?:
Enter YES (accept the default value) if the disk is to be used as a system disk. If you enter YES, a standard system loader is installed on the disk. Enter NO if the disk is not going to be used as a system disk and does not need the loader installed. NO is the standard response for data disks.

LISTING ACCESS NAME:
The pathname of the file or device to which the INV should list the results of the disk initialization. A null response specifies the terminal local file.

EXECUTION MODE(F,B):
If you enter F or FOREGROUND, the command executes in foreground mode. If you enter B or BACKGROUND, the command executes in background mode. The initial value is FOREGROUND. However, in a batch stream, this command always executes in background mode.

INV (INITIALIZE NEW VOLUME)

Example:

In the following example, the INV command initializes a disk volume named VOL1 on disk unit DS02. VOL1 can contain a maximum of 101 files (the supplied value of 100 rounded up to the nearest prime number, 101) and is used as a system disk.

```
[] INV

INITIALIZE NEW VOLUME
                        UNIT NAME: DS02
                      VOLUME NAME: VOL1
     NUMBER OF VCATALOG ENTRIES: 100
  DEFAULT PHYSICAL RECORD SIZE: 768
  HARDWARE INTERLEAVING FACTOR: 1
        FORCE CLEARING OF DISK?: NO
           USED AS SYSTEM DISK?: YES
            LISTING ACCESS NAME:
            EXECUTION MODE(F,B): FOREGROUND
```

Notes:

Initializing a new volume requires less than a minute. If you specify the FORCE CLEARING OF DISK, the disk initialization takes longer, depending on the disk type.

When you use the INV command in expert mode or within a batch stream, the LOADER ACCESS NAME prompt is accessible. This allows you to install a specific loader on the disk. If you do not enter a response, the standard system loader is installed on the disk.

An additional prompt is available in the expert or batch mode. Details of this prompt are as follows:

BAD TRACK ACCESS NAME:
The pathname of the file or the device name that is to supply bad track information. The default is a null response which indicates that there is no bad track information. The bad track information must be contained on the file or device in the following format:

HEAD, CYLINDER;
OR
HEAD, CYLINDER; HEAD, CYLINDER; ETC.

Enter bad track information in this format using decimal notation (for example, 2, 235; 0, 15;). If you enter a pathname in response to the BAD TRACK ACCESS NAME prompt, the file must contain the list of known bad tracks in the same HEAD, CYLINDER; format (for example, 2, 235; 0, 15;). All entries, including the last, must end with a semicolon. Improper positioning of the punctuation marks causes an error to be returned.

Related Commands:

IDS	(Initialize Disk Surface)
IV	(Install Volume)
UV	(Unload Volume)

IO (INSTALL OVERLAY)

4.84 IO (INSTALL OVERLAY)

Prompts:

INSTALL OVERLAY		
PROGRAM FILE OR LUNO:	{[site:]filename/integer}@	(*)
OVERLAY NAME:	[alphanumeric]	
OVERLAY ID:	[integer]	
OBJECT PATHNAME OR LUNO:	[site:]{pathname/integer}@	(*)
RELOCATABLE?:	YES/NO	(NO)
DELETE PROTECT?:	YES/NO	(NO)
ASSOCIATED SEGMENT NAME/ID:	[{character(s)/integer}]	(*)
ASSOCIATED SEGMENT TYPE:	[T/TASK/P/PROGRAM]	(*)

Purpose:

The IO command places an overlay optionally associated with a task or program segment on a program file. You can specify the overlay by name or ID. If you associate the overlay with a task or program segment, that segment must be installed on the same program file before the overlay is installed.

Prompt Details:

PROGRAM FILE OR LUNO:
The file name of, or LUNO assigned to, the program file on which the overlay is to be installed. If you specify a LUNO in response to this prompt, you must assign it to the program file prior to the execution of the IO command. If you specify zero, the .S$SHARED program file is assumed. If you specify >FF, the utility program file is assumed.

OVERLAY NAME:
A user-defined character string, a maximum of eight characters, that is unique to the program file. If you specify a null response, the system uses the IDT name of the object module as the name of the overlay.

OVERLAY ID:
An integer value in the range of 0 through 255 that is associated with the overlay name and is unique to the program file. If you specify zero or a null response, the system assigns an ID to the overlay.

OBJECT PATHNAME OR LUNO:
The name of, or the LUNO assigned to, the device or file where the object module for the overlay resides.

RELOCATABLE?:
If you enter YES, the overlay is allowed to be loaded at an address other than its natural load address.

IO (INSTALL OVERLAY)

DELETE PROTECT?:
If you enter YES, the overlay cannot be deleted from the program file unless you execute the Modify Overlay Entry (MOE) command to unprotect the overlay prior to the execution of the Delete Overlay (DO) command. If you specify NO, you can delete the overlay by executing the DO command.

ASSOCIATED SEGMENT NAME/ID:
The name or ID of a previously installed segment on the same program file as the overlay. The overlay is automatically deleted when the segment is deleted.

ASSOCIATED SEGMENT TYPE:
If the segment associated with this overlay is a task segment, respond TASK; otherwise, respond PROGRAM.

Example:

In the following example, the IO command installs a relocatable overlay named OLAY1 on the .S$SHARED program file. LUNO > 0A has been assigned to the file where the object module resides, and ID > 83 identifies the associated segment.

```
[]IO

INSTALL OVERLAY
        PROGRAM FILE OR LUNO: 0
                OVERLAY NAME: OLAY1
                  OVERLAY ID: 0
    OBJECT PATHNAME OR LUNO: >0A
                RELOCATABLE?: YES
             DELETE PROTECT?: NO
 ASSOCIATED SEGMENT NAME/ID: >83
    ASSOCIATED SEGMENT TYPE: TASK
```

Notes:

It is recommended that the segments be installed on your own program file rather than on the system program file.

Related Commands:

DO	(Delete Overlay)
IP	(Install Procedure Segment)
IPS	(Install Program Segment)
IRT	(Install Real-Time Task Segment)
IT	(Install Task Segment)
MOE	(Modify Overlay Entry)

IP (INSTALL PROCEDURE SEGMENT)

4.85 IP (INSTALL PROCEDURE SEGMENT)

Prompts:

```
INSTALL PROCEDURE SEGMENT
        PROGRAM FILE OR LUNO:  {[site:]filename/integer}@      (*)
            PROCEDURE NAME:    [alphanumeric]
              PROCEDURE ID:    [integer]
  OBJECT PATHNAME OR LUNO:     {[site:]pathname/integer}@     (*)
          MEMORY RESIDENT?:    YES/NO                         (NO)
          DELETE PROTECT?:     YES/NO                         (NO)
```

After you enter responses for the previous prompts, the following prompts are displayed on your terminal:

```
990/12 FLAGS
          EXECUTE PROTECT?:    YES/NO                         (NO)
            WRITE PROTECT?:    YES/NO                         (NO)
WRITABLE CONTROL STORAGE?:     YES/NO                         (NO)
```

Purpose:

The IP command places a procedure segment on a program file and assigns a procedure ID for use by subsequent Install Task Segment (IT) commands.

Prompt Details:

PROGRAM FILE OR LUNO:
The file name or the LUNO assigned to the program file on which the procedure segment is to be installed. If you specify a LUNO in response to this prompt, you must assign it to the program file prior to execution of the IP command. If you specify zero, the .S$SHARED program file is assumed. If you specify > FF, the utility program file is assumed.

PROCEDURE NAME:
A user-defined character string, up to eight characters, that identifies the procedure segment. If you do not specify the procedure name, the system assigns the IDT name of the object module as the procedure name.

PROCEDURE ID:
An integer value in the range of 0 through 255 that is associated with the procedure name and is unique to all other procedures and program segments in the specified program file. If you specify zero or a null response, the system assigns an ID.

OBJECT PATHNAME OR LUNO:
The name or the LUNO assigned to a device or file where the object module for the procedure segment resides.

IP (INSTALL PROCEDURE SEGMENT)

MEMORY RESIDENT?:
The procedure segment is loaded into memory during initial program load (IPL) where it stays in memory even when terminated if you enter YES and the procedure segment is installed on the .S$SHARED program file or the applications program file specified at system generation.

DELETE PROTECT?:
If you enter YES, you cannot delete the procedure segment from the program file unless you execute the Modify Procedure Segment Entry (MPE) command to unprotect the procedure segment prior to the execution of the Delete Procedure Segment (DP) command. If you specify NO, you can delete the procedure segment by executing the DP command.

EXECUTE PROTECT?:
If you enter YES, the procedure segment cannot be executed. The protection is enforced only on a 990/12 computer.

WRITE PROTECT?:
If you enter YES, the procedure segment cannot be modified when in memory. The protection is enforced only on a 990/12 computer.

WRITABLE CONTROL STORAGE?:
If you enter YES, the procedure segment uses the writable control storage area. Writable control storage is available only with supplied software on a 990/12 computer.

Example:

In the following example, the IP command installs a procedure segment named MYPROC on the user's program file. LUNO > AA has been assigned to the program file, and LUNO > BB has been assigned to the file where the object module resides. The system assigns a procedure ID.

```
[]IP

INSTALL PROCEDURE SEGMENT
        PROGRAM FILE OR LUNO: >AA
              PROCEDURE NAME: MYPROC
                PROCEDURE ID: 0
     OBJECT PATHNAME OR LUNO: >BB
             MEMORY RESIDENT?: NO
              DELETE PROTECT?: NO

  990/12 FLAGS
            EXECUTE PROTECT?: NO
              WRITE PROTECT?: NO
   WRITABLE CONTROL STORAGE?: NO
```

IP (INSTALL PROCEDURE SEGMENT)

Assumptions:

The program file specified has previously been created, and the procedure name and/or ID specified is unique to the program file.

Notes:

It is recommended that procedure segments be installed in your own program file, rather than in the system program file.

Related Commands:

DP	(Delete Procedure Segment)
IO	(Install Overlay)
IPS	(Install Program Segment)
IRT	(Install Real-Time Task Segment)
IT	(Install Task Segment)
MPE	(Modify Procedure Segment Entry)

4.86 IPS (INSTALL PROGRAM SEGMENT)

Prompts:

```
INSTALL PROGRAM SEGMENT
        PROGRAM FILE OR LUNO:  {[site:]filename/integer}@      (*)
                SEGMENT NAME:  [alphanumeric]
                  SEGMENT ID:  [integer]
   OBJECT PATHNAME OR LUNO:    {[site:]pathname/integer}@      (*)
     DEFAULT SEGMENT FLAGS?:   YES/NO                         (YES)
```

If the response to the DEFAULT SEGMENT FLAGS? prompt is NO, the following set of prompts is displayed on your terminal:

```
DEFINE SEGMENT FLAGS
          SYSTEM SEGMENT?:  YES/NO  (NO)
        MEMORY RESIDENT?:   YES/NO  (NO)
        DELETE PROTECT?:    YES/NO  (NO)
              UPDATABLE?:   YES/NO  (NO)
               SHARABLE?:   YES/NO  (NO)
            REPLICATABLE?:  YES/NO  (NO)
     IN MEMORY REUSABLE?:   YES/NO  (NO)
     IN MEMORY COPYABLE?:   YES/NO  (NO)
```

After you enter the responses for the previous prompts, the following prompts are displayed on your terminal:

```
990/12 FLAGS
          EXECUTE PROTECT?:  YES/NO  (NO)
            WRITE PROTECT?:  YES/NO  (NO)
WRITABLE CONTROL STORAGE?:   YES/NO  (NO)
```

Purpose:

The IPS command installs a program segment on a program file and assigns it an ID.

Prompt Details:

PROGRAM FILE OR LUNO:

The file name or the LUNO assigned to the program file on which the program segment is to be installed. If you specify a LUNO in response to this prompt, you must assign it to the program file prior to execution of the IPS command. If you specify zero, the .S$SHARED program file is assumed. If you specify >FF, the utility program file is assumed.

SEGMENT NAME:

A user-defined character string, up to eight characters, composed of characters which are legal in pathnames. The segment name must be unique to all other procedure and program segment names in the specified program file. If you specify a null response, the IDT name of the object file is used as the segment name.

IPS (INSTALL PROGRAM SEGMENT)

SEGMENT ID:
An integer value in the range of 0 through 255 that is associated with the program segment and is unique to all other procedure and program segments in the specified program file. If you specify zero or a null response, the system assigns the ID.

OBJECT PATHNAME OR LUNO:
The pathname or the LUNO assigned to the device or file where the object module for the program segment resides.

DEFAULT SEGMENT FLAGS?:
If you enter YES, the initial values are used for the flags. If you enter NO, you have the option of specifying which program segment flags will be modified.

SYSTEM SEGMENT?:
If you enter YES,the program segment can be accessed only by a system task.

MEMORY RESIDENT?:
The program segment is loaded into memory during initial program load (IPL) and remains in memory even when terminated if you enter YES and the program segment is installed on the .S$SHARED program file or the applications program file specified at system generation.

DELETE PROTECT?:
If you enter YES, you cannot delete the program segment from the program file unless you execute the Modify Program Segment Entry (MSE) command to unprotect the program segment prior to the execution of the Delete Program Segment (DPS) command. If you specify NO, you can delete the program segment by executing the DPS command.

UPDATABLE?:
If you enter YES, the data of a program segment can be modified, and the program segment is written to disk with the new data modifications after the program segment is no longer used.

SHARABLE?:
If you enter YES, the program segment can be shared concurrently with more than one task.

REPLICATABLE?:
If you enter YES, multiple copies of the program segment can be in memory simultaneously.

IN MEMORY REUSABLE?:
If you enter YES, the program segment in memory can be reused after termination by another task rather than a new copy being read from disk.

IN MEMORY COPYABLE?:
If you enter YES, the program segment can be copied from memory rather than being copied from disk. This situation can occur when the program segment is in memory and another user wants to use the program segment.

IPS (INSTALL PROGRAM SEGMENT)

EXECUTE PROTECT?:
If you enter YES, execution of the program segment is prohibited. The protection is enforced only on a 990/12 computer.

WRITE PROTECT?:
If you enter YES, the program segment cannot be modified in memory. The protection is enforced only on a 990/12 computer.

WRITABLE CONTROL STORAGE?:
If you enter YES, the program segment uses the writable control storage area which is available only for supplied software on a 990/12 computer.

Example:

In the following example, the IPS command installs a program segment named PRINT with an ID of >3B on a user's program file named VOL1.USER. The object module for the program segment resides on a file named SYS1.KC0017.PROBJ.

The program segment is to be installed with the following attributes: disk resident, delete protected, sharable, and replicatable.

```
[] IPS

INSTALL PROGRAM SEGMENT
     PROGRAM FILE OR LUNO: VOL1.USER
             SEGMENT NAME: PRINT
               SEGMENT ID: 03B
  OBJECT PATHNAME OR LUNO: SYS1.KC0017.PROBJ
   DEFAULT SEGMENT FLAGS?: NO

DEFINE SEGMENT FLAGS
          SYSTEM SEGMENT?: NO
         MEMORY RESIDENT?: NO
          DELETE PROTECT?: YES
               UPDATABLE?: NO
                SHARABLE?: YES
            REPLICATABLE?: YES
      IN MEMORY REUSABLE?: NO
      IN MEMORY COPYABLE?: NO

990/12 FLAGS
        EXECUTE PROTECT?: NO
          WRITE PROTECT?: NO
WRITABLE CONTROL STORAGE?: NO
```

IPS (INSTALL PROGRAM SEGMENT)

Assumptions:

The specified program segment name is unique to the program file where the program segment is to be installed.

Related Commands:

DPS	(Delete Program Segment)
IO	(Install Overlay)
IP	(Install Procedure Segment)
IRT	(Install Real-Time Task Segment)
IT	(Install Task Segment)
MSE	(Modify Program Segment Entry)

4.87 IRT (INSTALL REAL-TIME TASK SEGMENT)

Prompts:

```
INSTALL REAL-TIME TASK SEGMENT
      PROGRAM FILE OR LUNO:  {[site:]filename/integer}@          (*)
                 TASK NAME:  [alphanumeric]
                   TASK ID:  [integer]
  OBJECT PATHNAME OR LUNO:  {[site:]pathname/integer}@          (*)
                  PRIORITY:  integer
  DEFAULT TASK FLAGS?:  YES/NO                                  (YES)
  ATTACHED PROCEDURES?:  YES/NO                                 (NO)
```

If the response to the DEFAULT TASKS FLAGS? prompt was NO, the following prompts are displayed on your terminal:

```
DEFINE TASK FLAGS
          PRIVILEGED?:  YES/NO  (NO)
        SYSTEM TASK?:  YES/NO  (NO)
     MEMORY RESIDENT?:  YES/NO  (NO)
       REPLICATABLE?:  YES/NO  (YES)
     DELETE PROTECT?:  YES/NO  (NO)
  IN MEMORY COPYABLE?:  YES/NO  (NO)
  IN MEMORY REUSABLE?:  YES/NO  (NO)
          UPDATABLE?:  YES/NO  (NO)
  SOFTWARE PRIVILEGED?:  YES/NO  (NO)
```

After you enter responses for the previous prompts, the following prompts are displayed on your terminal:

```
990/12 FLAGS
        EXECUTE PROTECT?:  YES/NO  (NO)
  OVERFLOW CHECKING?:  YES/NO  (NO)
WRITABLE CONTROL STORAGE?:  YES/NO  (NO)
```

If the response to the ATTACHED PROCEDURES prompt was YES, the following prompts are displayed on your terminal:

```
ATTACH TASK PROCEDURES
          1ST PROCEDURE ID:  integer  (0)
P1 FROM TASKS PROGRAM FILE?:  [YES/NO]  (YES)
          2ND PROCEDURE ID:  [integer]  (0)
P2 FROM TASKS PROGRAM FILE?:  [YES/NO]  (YES)
```

IRT (INSTALL REAL-TIME TASK SEGMENT)

Purpose:

The IRT command places an executable real-time task segment into a program file. If the task has attached procedure segments, it is advisable to install the procedure segments on the program file before installing the task segment. If you install the task segment first, you must execute the Modify Task Segment Entry (MTE) command and specify the information for the attached procedure segments.

The priorities of real-time task segments are lower than system tasks but higher than user tasks. Therefore, when system, real-time, and user tasks are waiting to be executed, real-time tasks execute after system tasks but before user tasks.

Prompt Details:

PROGRAM FILE OR LUNO:
The file name or the LUNO assigned to the program file on which the task segment is to be installed. If you specify a LUNO in response to this prompt, you must assign it to the program file prior to execution of the IRT command. If you specify zero, the .S$SHARED program file is assumed. If you specify > FF, the utility program file is assumed.

TASK NAME:
A user-defined character string, up to eight ASCII characters, which is the name of the task segment to be installed on the specified program file. If you enter a null response, the system assigns the IDT name of the object module as the name of the task segment.

TASK ID:
An integer value in the range of 0 through 255 that is associated with the task name and is unique to all other tasks in the specificied program file. If the response to this prompt is zero or a null response is entered, the system assigns an available ID.

OBJECT PATHNAME OR LUNO:
The pathname of or the LUNO assigned to the device or file where the object module for the task segment resides.

PRIORITY:
The integer value that represents the execution priority level of the task segment. Priorities range from 1 through 127, with 1 being the highest priority.

DEFAULT TASK FLAGS?:
If you enter NO in response to this prompt, you have the option to set the task flags. If you enter YES, the initial values are used for the flags.

PRIVILEGED?:
If you enter YES, the task is allowed to execute privileged hardware instructions. Privileged hardware instructions should be executed cautiously and only if you are very familiar with the system.

IRT (INSTALL REAL-TIME TASK SEGMENT)

SYSTEM TASK?:
If you enter YES, the task is allowed to execute in system memory space. Tasks should be executed in system memory space with caution and only by the user who is very familiar with the system.

MEMORY RESIDENT?:
The task is loaded into memory during initial program load (IPL) and remains in memory when terminated if YES is entered and the task is installed on the .S$SHARED program file or the applications program file specified at system generation.

REPLICATABLE?:
If you enter YES, multiple copies of the task can be in memory simultaneously.

DELETE PROTECT?:
If you enter YES, the task segment cannot be deleted from the program file unless the Modify Task Segment Entry (MTE) command is used to unprotect the task segment prior to the execution of the Delete Task Segment (DT) command. If you specify NO, the task segment can be deleted by executing the DT command.

IN MEMORY COPYABLE?:
If you enter YES, the task segment can be copied from memory rather than being copied from disk. This situation can occur if the task is in memory and another user wants to execute the task.

IN MEMORY REUSABLE?:
If you enter YES, the task segment memory can be reused by another task rather than being copied from disk or from one memory location to another.

UPDATABLE?:
If you enter YES, the data of the task segment can be modified, and the task segment is written to disk with the new data modifications after the task terminates.

SOFTWARE PRIVILEGED?:
If you enter YES, the task is allowed to execute privileged supervisor calls.

EXECUTE PROTECT?:
If you enter YES, execution of the task is prohibited. The protection is enforced only on a 990/12 computer.

OVERFLOW CHECKING?:
If you enter YES, the occurrence of arithmetic overflow causes control of the task to pass to the end action routine of the task. Overflow checking is available only on a 990/12 computer.

WRITABLE CONTROL STORAGE?:
If YES, the task uses the writable control storage area. Writable control storage is available only with supplied software on a 990/12 computer.

IRT (INSTALL REAL-TIME TASK SEGMENT)

ATTACHED PROCEDURES?:
If you enter YES in response to this prompt, you are promtped for the ID(s) of procedure segments attached to this task segment and asked if the procedures reside on the same program file as the task.

1ST PROCEDURE ID:
The integer value representing the ID of a procedure segment attached to the task segment. If you enter zero, there are no procedures.

P1 FROM TASKS PROGRAM FILE?:
If YES, the attached procedure segment whose ID was specified for the 1ST PROCEDURE ID prompt resides on the same program file as the task segment. If NO, that procedure segment must reside on the .S$SHARED program file.

2ND PROCEDURE ID:
The integer value representing the ID of a procedure segment attached to the task segment. If you enter zero, there is no second procedure segment.

P2 FROM TASKS PROGRAM FILE?:
If you enter YES, the attached procedure segment whose ID was specified for the 2ND PROCEDURE ID prompt resides on the same program file as the task segment. If you enter NO, that procedure segment must reside on the .S$SHARED program file.

Example:

In the following example, the IRT command installs a task segment onto a user program file, with one attached procedure segment on the .S$SHARED program file. LUNO >4A is assigned to the file where the object module resides. An available task ID is allocated for the task. The default for task flags is taken.

```
[] IRT

INSTALL REAL-TIME TASK SEGMENT
        PROGRAM FILE OR LUNO: VOL1.MYFILE
                   TASK NAME: MYFILE
                     TASK ID:
     OBJECT PATHNAME OR LUNO: >4A
                    PRIORITY: 3
         DEFAULT TASK FLAGS?: YES
        ATTACHED PROCEDURES?: YES

ATTACH TASK PROCEDURES
             1ST PROCEDURE ID: >42
P1 FROM TASKS PROGRAM FILE?: NO
             2ND PROCEDURE ID: 0
P2 FROM TASKS PROGRAM FILE?: NO
```

IRT (INSTALL REAL-TIME TASK SEGMENT)

Notes:

It is recommended that the task segment be installed on your own program file, rather than on the system program file.

Execute privileged hardware instructions and tasks which execute in system memory space with caution and only if you are very familiar with the system.

Related Commands:

DT	(Delete Task)
IO	(Install Overlay)
IP	(Install Procedure Segment)
IPS	(Install Program Segment)
IT	(Install Task Segment)
MTE	(Modify Task Segment Entry)

ISL (INITIALIZE SYSTEM LOG)

4.88 ISL (INITIALIZE SYSTEM LOG)

Prompts:

```
INITIALIZE SYSTEM LOG
                ATTENTION DEVICE: [devicename]@          (*)
                      LOG DEVICE: [devicename]@          (*)
      MODIFY FILE PROCESSING?: YES/NO                    (NO)
                  RECREATE FILES?: YES/NO                (NO)
```

If the response to the MODIFY FILE PROCESSING? prompt is YES, the following prompts are displayed:

```
INITIALIZE SYSTEM LOG
SYSTEM LOG PROCESSOR TASK ID: {0, 051, 052}            (*)
 USER LOG PROCESSOR TASK ID: integer                   (*)
```

If the response to the RECREATE FILES? prompt is YES, the following prompt is displayed:

```
INITIALIZE SYSTEM LOG
                    ALLOCATION: [integer]              (*)
```

Purpose:

System log information is recorded on system log files and, optionally, at a physical device such as a line printer or terminal. During system operation or after a logging error, you may want to reinitialize the log to change logging devices. You may also need to reinitialize a device that was used for another purpose. Perform the reinitialization of the system log by using the ISL command.

When you reinitialize the system log-on files, the system begins recording the log on whichever file was last updated. Log messages are logged in that file until it is full. Then the following message appears at the device specified as the attention device:

```
********* LOG FILE x FULL
```

where:

x is the number 1 or 2, representing Log File 1 or Log File 2, respectively.

The system log full message tells you that the contents of the log file (.S$LOG1 or .S$LOG2) should be copied to another file or device because it will be overwritten if the secondary system log file fills up. Optionally, you can specify a task to be bid whenever a log file is full.

In addition to log files, you can keep the log on a logging device. The advantage is that an error at the logging device does not disable logging to the system log files.

ISL (INITIALIZE SYSTEM LOG)

Prompt Details:

ATTENTION DEVICE:
A device name to specify a device as the receiver of attention messages output by the system during the logging process. These messages can consist of:

- Device hardware errors
- Task abnormal termination errors
- Memory errors
- Messages generated by user programs
- Messages generated by DNOS utilities
- Device statistics

A null response specifies there is no attention device.

LOGGING DEVICE:
A device name to specify a device to which the system log is output. Specify a null response if you do not want to log to a device. However, if you do specify a null response, messages continue to be written to the log file.

MODIFY FILE PROCESSING?:
If you enter YES in response to this prompt, the processing of log files is to be modified. If you enter NO, processing of log files is unchanged. If YES, a response to the SYSTEM LOG PROCESSOR TASK ID and the USER LOG PROCESSOR TASK ID is required.

SYSTEM LOG PROCESSOR TASK ID:
Respond to this prompt with the task ID of an installed task on the utility program file. The task is bid whenever a log file is filled. A zero specifies that none is desired.

USER LOG PROCESSOR TASK ID:
Respond to this prompt with the task ID of an installed task on the utility program file. The task is bid whenever a log file is filled and allows you to bid a task to summarize system log information, if desired. Use this parameter with a user-supplied task. A zero specifies that none is desired.

RECREATE FILES?:
If you enter YES in response to this prompt, the log files are deleted and then recreated with the allocation specified for the ALLOCATION prompt. A NO response will not recreate the file.

ALLOCATION:
An integer that specifies the initial file size, in logical records, of the recreated log files. This prompt is displayed only if the response to the RECREATE FILES? prompt is YES. A null response specifies that the files are to be recreated at the same size.

ISL (INITIALIZE SYSTEM LOG)

Example:

In the following example, the ISL command modifies the attention and log devices.

```
[] ISL

INITIALIZE SYSTEM LOG
                  ATTENTION DEVICE: ST01
                    LOGGING DEVICE: ST01
          MODIFY FILE PROCESSING?: NO
                   RECREATE FILES?: NO
```

Notes:

If the log files are to be deleted and then recreated, the system and user log tasks will not be able to process the deleted log files, and the information that could have been obtained by these tasks will be lost. The defaults displayed for the prompts are the current values that the system is using.

The ISL command is processed by the Execute System Configuration Utility (XSCU). If a System Configuration Utility (SCU) session is in progress, an ISL command modifies the log-on characteristics of the system being modified by the SCU session (not the running system).

4.89 IT (INSTALL TASK SEGMENT)

Prompts:

```
INSTALL TASK SEGMENT
        PROGRAM FILE OR LUNO:  {[site:]filename/integer}@        (*)
                   TASK NAME:  [alphanumeric]
                     TASK ID:  [integer]
   OBJECT PATHNAME OR LUNO:  {[site:]pathname/integer}@          (*)
                    PRIORITY:  [integer]                        (4)
  DEFAULT TASK FLAGS?:  YES/NO                                  (YES)
  ATTACHED PROCEDURES?:  YES/NO                                 (NO)
```

If the response to the DEFAULT TASKS FLAGS? prompt is NO, the following sets of prompts are displayed on your terminal:

```
DEFINE TASK FLAGS
          PRIVILEGED?:  YES/NO                                  (NO)
          SYSTEM TASK?:  YES/NO                                 (NO)
       MEMORY RESIDENT?:  YES/NO                                (NO)
         REPLICATABLE?:  YES/NO                                 (YES)
       DELETE PROTECT?:  YES/NO                                 (NO)
   IN MEMORY COPYABLE?:  YES/NO                                 (NO)
   IN MEMORY REUSABLE?:  YES/NO                                 (NO)
          UPDATABLE?:  YES/NO                                   (NO)
   SOFTWARE PRIVILEGED?:  YES/NO                                (NO)
```

After you enter the responses to the previous prompts, the following prompts are displayed on your terminal:

```
990/12 FLAGS
       EXECUTE PROTECT?:  YES/NO                                (NO)
     OVERFLOW CHECKING?:  YES/NO                                (NO)
WRITABLE CONTROL STORAGE?:  YES/NO                              (NO)
```

If the response to the ATTACHED PROCEDURES prompt is YES, the following set of prompts is displayed on your terminal:

```
ATTACH TASK PROCEDURES
          1ST PROCEDURE ID:  integer                            (0)
P1 FROM TASKS PROGRAM FILE?:  [YES/NO]                          (YES)
          2ND PROCEDURE ID:  [integer]                          (0)
P2 FROM TASKS PROGRAM FILE?:  [YES/NO]                          (YES)
```

IT (INSTALL TASK SEGMENT)

Purpose:

The IT command places an executable task segment on a program file. If the task segment has attached procedure segments, it is advisable that the procedure segments be installed before installing the task segment. If the task segment is installed first, you must then execute the Modify Task Segment Entry (MTE) command specifying the attached procedure segment information.

Prompt Details:

PROGRAM FILE OR LUNO:
The file name or the LUNO assigned to the program file on which the task segment is to be installed. If you specify a LUNO in response to this prompt, you must assign it to the program file prior to the execution of the IT command. If you specify zero, the .S$SHARED program file is assumed. If you specify >FF, the utility program file is assumed.

TASK NAME:
A user-defined character string, up to eight ASCII characters long, which is the name of the task segment to be installed on the specified program file. If you specify a null response, the system assigns the IDT name of the object module as the task segment name.

TASK ID:
An integer value in the range of 0 through 255 that is associated with the task name and is unique to all other tasks in the specified program file. If you specify zero or a null response, the system assigns an available ID.

OBJECT PATHNAME OR LUNO:
The pathname of or the LUNO assigned to a device or file where the object module of the task segment resides.

PRIORITY:
The integer value that represents the execution priority level of the task. Priorities can range from zero through four. Priorities zero through three are fixed, with priority zero as the highest level and three the lowest. Priority four is dynamically managed by the operating system. Four is the default priority level.

DEFAULT TASK FLAGS?:
If you enter NO in response to this prompt, you have the option to set the task flags. If you enter YES, the initial values are used for the flags.

PRIVILEGED?:
If you enter YES, the task is allowed to execute privileged hardware instructions.

SYSTEM TASK?:
If you enter YES, the task is allowed to execute in system memory space.

IT (INSTALL TASK SEGMENT)

MEMORY RESIDENT?:
The task is loaded into memory during initial program load (IPL) and remains in memory when terminated if you enter YES and the task is installed on the .S$SHARED program file or the applications program file specified at system generation.

REPLICATABLE?:
If you enter YES, multiple copies of the task can be in memory simultaneously.

DELETE PROTECT?
If you enter YES, the task segment cannot be deleted from the program file unless you use the Modify Task Segment Entry (MTE) command to unprotect the task segment prior to the execution of the Delete Task Segment (DT) command. If you specify NO, you can delete the task segment by the DT command.

IN MEMORY COPYABLE?:
If YES, the task segment can be copied from memory rather than being copied from disk. This situation can occur if the task is in memory and another user wants to execute the task.

IN MEMORY REUSABLE?:
If YES, the task segment memory can be reused by another task rather than being copied from disk or from one memory location to another.

UPDATABLE?:
If YES, the data of a task can be modified and the task segment is written to disk with the new data modifications when the task terminates.

SOFTWARE PRIVILEGED?:
If you enter YES, the task is allowed to execute privileged supervisor calls.

EXECUTE PROTECT?:
If you enter YES, execution of the task segment is prohibited. The protection is enforced only on a 990/12 computer.

OVERFLOW CHECKING?:
If you enter YES, the occurrence of arithmetic overflow causes control of the task to pass to the end action routine of the task. Overflow checking is enforced only on a 990/12 computer.

WRITABLE CONTROL STORAGE?:
If you enter YES, the task uses the writable control storage area. Writable control storage is available only with supplied software on a 990/12 computer.

ATTACHED PROCEDURES?:
If you enter YES in response to this prompt, you are prompted for the ID(s) of procedure segments attached to this task segment and asked if the procedures reside on the same program file as the task.

1ST PROCEDURE ID:
The integer value representing the ID of a procedure attached to the task segment. If you enter zero, there are no procedures.

IT (INSTALL TASK SEGMENT)

P1 FROM TASKS PROGRAM FILE?:
If you enter YES, the attached procedure segment whose ID was specified for the 1ST PROCEDURE ID prompt resides on the same program file as the task segment. If NO, that procedure segment must reside on the .S$SHARED program file.

2ND PROCEDURE ID:
The integer value representing the ID of a procedure segment attached to the task segment. If you enter zero, there is no second procedure.

P2 FROM TASKS PROGRAM FILE?:
If YES, the attached procedure segment whose ID was specified for the 2ND PROCEDURE ID prompt resides on the same program file as the task. If you enter NO, that procedure segment must reside on the .S$SHARED program file.

Example:

In the following example, the IT command installs a task with one attached procedure on the .S$SHARED program file. An available task ID is allocated for the task. The default to the task flags prompt is taken.

```
[] IT

INSTALL TASK SEGMENT
      PROGRAM FILE OR LUNO: 0
                 TASK NAME: MYFILE
                   TASK ID:
  OBJECT PATHNAME OR LUNO: >4A
                 PRIORITY: 3
       DEFAULT TASK FLAGS?: YES
      ATTACHED PROCEDURES?: YES

ATTACH TASK PROCEDURES
          1ST PROCEDURE ID: >42
P1 FROM TASKS PROGRAM FILE?: NO
          2ND PROCEDURE ID: 0
P2 FROM TASKS PROGRAM FILE?: NO
```

Notes:

Tasks should be installed in your own program file rather than in the system program files.

Related Commands:

DT	(Delete Task)
IO	(Install Overlay)
IP	(Install Procedure Segment)
IPS	(Install Program Segment)
IRT	(Install Real-Time Task Segment)
MTE	(Modify Task Segment Entry)

4.90 IV (INSTALL VOLUME)

Prompts:

```
INSTALL VOLUME
                UNIT NAME:  devicename@
              VOLUME NAME:  alphanumeric@
```

Purpose:

After you mount the disk volume in a disk unit, the IV command installs an initialized disk volume for operation. You cannot install a volume when the name specified with the IV command does not match the volume name assigned to the volume. A volume being installed cannot have the same volume name as one already installed (including the system disk).

The IV command writes a message to the system log specifying the user ID, the station name, volume name, and unit name associated with the disk being installed.

Prompt Details:

UNIT NAME:
The device name of the disk unit on which the volume is being installed. Mount the volume on this device.

VOLUME NAME:
The one- to eight-character string beginning with a letter, naming the volume being installed.

Example:

In the following example, the IV command installs a disk volume named VOL1 on disk unit DS02:

```
[] IV

INSTALL VOLUME
          UNIT NAME: DS02
        VOLUME NAME: VOL1
```

Notes:

The DS10 and CD1400 disk drives are each physically organized as a removable platter and a nonremovable platter turned by a single disk spindle. When the removable platter is being loaded or unloaded, the nonremovable platter is inaccessible. Before changing the removable platter, ensure that neither platter is in use.

Related Commands:

CRV	(Check and Reset Volume)
INV	(Initialize New Volume)
UV	(Unload Volume)

KBT (KILL BACKGROUND TASK)

4.91 KBT (KILL BACKGROUND TASK)

Purpose:

The KBT command forces the termination of a program that is executing in background mode.

Example:

In the following example, the KBT command terminates a background task.

```
[] KBT

KILL BACKGROUND TASK
```

The following message is displayed on the terminal where the KBT command is executed when the termination completes successfully:

```
EXECUTION TERMINATED
```

Messages:

I UTILITY-0350 EXECUTION TERMINATED
I UTILITY-0190 THERE IS NO BACKGROUND ACTIVITY AT THIS TERMINAL
I UTILITY-0191 TASK WAS NOT FOUND IN THE SYSTEM

Notes:

The task state displayed is the state of the task at the time the attempt was made to kill the task. KBT displays the task state if the task is not terminated within 20 seconds.

Related Commands:

KT	(Kill Task)
SBS	(Show Background Status)
WAIT	(Wait for Background Completion)

4.92 KJ (KILL JOB)

Prompts:

KILL JOB
JOB NAME OR ID: {alphanumeric/integer}

Purpose:

The KJ command kills any job currently executing under your user ID or allows the system operator to kill any job.

Prompt Details:

JOB NAME OR ID:
The job name specified during job creation or the job ID returned when the Execute Job (XJ) command executed. The XJ command is discussed later in this section.

Example:

In the following example, the user kills a job named PRINT currently executing under the user's ID by entering the KJ command as follows:

```
[] KJ

KILL JOB
        JOB NAME OR ID: PRINT
```

Notes:

If other terminals are reconnected to the job under which your tasks are executing, an attempt to kill your job will also kill the sessions for users of the reconnected terminals.

Related Commands:

HJ	(Halt Job)
LJ	(List Jobs)
MJP	(Modify Job Priority)
RJ	(Resume Job)
SJS	(Show Job Status for < user ID>)
XBJ	(Execute Batch Job)
XJ	(Execute Job)

KO (KILL OUTPUT AT DEVICE)

4.93 KO (KILL OUTPUT AT DEVICE)

Prompts:

```
KILL OUTPUT AT DEVICE
     DEVICE OR CLASS NAME:  [site:]{devicename/character(s)}@      (*)
                  SPOOL ID:  {alphanumeric/ALL}
```

Purpose:

The KO command allows the system operator or the owner of the print request to terminate queued output.

Prompt Details:

DEVICE OR CLASS NAME:
The name of the device or class at which queued output is to be terminated. A response is required if you specify ALL for the SPOOL ID prompt.

SPOOL ID:
Unique six-character ID assigned by the Spooler Subsystem when a print request is processed. Determine the spool ID by executing the Show Output Status (SOS) command. To kill all of your print requests queued for the specific device or class, you should enter ALL in response to this prompt.

Example:

In the following example, the output of one file at a line printer LP01 is terminated through the KO command as shown below:

```
[] KO

KILL OUTPUT AT DEVICE
         DEVICE NAME: LP01
            SPOOL ID: S00001
```

Notes:

Unless you are the system operator, only your own print requests can be killed.

Related Commands:

HO	(Halt Output at Device)
MO	(Modify Output at Device)
RO	(Resume Output at Device)
SOS	(Show Output Status)

4.94 KOM (KILL OPERATOR MESSAGES)

Purpose:

The KOM command stops sending operator messages to you, canceling the Receive Operator Messages (ROM) command. The KOM command has no effect unless executed after the ROM command.

Example:

In the following example, the KOM command halts the relaying of operator messages to the user's terminal:

```
[] KOM
```

Notes:

You or the system operator can receive one more message (or set of messages) after this command is accepted if the message(s) is sent while you are entering and processing the command.

The KOM command is only valid interactively.

Related Commands:

COM	(Create Operator Message)
KOR	(Kill Operator Interface Request)
LOM	(List Operator Messages)
QOI	(Quit Operator Interface)
ROM	(Receive Operator Messages)
ROR	(Respond to Operator Interface Request)
XOI	(Execute Operator Interface)

KOR (KILL OPERATOR INTERFACE REQUEST)

4.95 KOR (KILL OPERATOR INTERFACE REQUEST)

Prompts:

KILL OPERATOR INTERFACE REQUEST
REQUEST ID: integer

After you enter a response for the REQUEST ID prompt, the following prompt is displayed if a response is required:

KILL REQUEST: YES/NO

Purpose:

The KOR command specifies that the system operator is unable to fulfill a request. The operator enters the KOR command specifying the request ID associated with the request and is then shown the text of the request and asked to verify that this is the request to be denied. If the operator answers affirmatively, a reply message is sent to the requesting task saying the request will not be fulfilled. If the request does not require a response, the user receives a message noting that the request has been acknowledged, instead of a second prompt.

Pressing the F5 terminal function key is equivalent to entering the KOR command if the Execute Operator Interface (XOI) command has previously been entered and the XOI task is active in foreground mode.

Prompt Details:

REQUEST ID:
The request ID associated with the request to be canceled.

KILL REQUEST?:
If you specify YES, a negative reply is sent to the requester. If NO, the request is left pending. Pressing the Command key has the same effect as entering NO.

KOR (KILL OPERATOR INTERFACE REQUEST)

Example:

In the following example, the operator has received a request to mount tape 444555 on magnetic tape unit MT02 which is in use by a previous request:

```
OR    123* FROM USER1234 AT 12:34 -
  MOUNT TAPE 444555 ON DRIVE MT02
```

The operator enters the KOR command to cancel the request as shown below:

```
[] KOR

KILL OPERATOR INTERFACE REQUEST
  REQUEST ID: 123
```

The operator is then prompted with the message as follows:

```
MOUNT TAPE 444555 ON DRIVE MT02
KILL REQUEST?:
```

If the operator request ID is not followed by an asterisk, the mount request message and the KILL REQUEST? prompt are not displayed. However, the following message is displayed, acknowledging the KOR request:

```
I UTILITY-08531 REQUEST 123 ACKNOWLEDGED
```

Assumptions:

The person entering the KOR command has previously successfully entered the command; or the user has entered the ROM command and no other user has entered the XOI command. Also, the request ID corresponds to a pending operator request and no other user is in the process of responding to that request.

Notes:

If the operator request ID is followed by an asterisk, the associated task is suspended until you enter a response for the request. If the request ID is not followed by an asterisk, the associated task is not suspended. However, the operator must enter the KOR or Respond to Operator Interface Request (ROR) command to complete the request. The KOR command is invalid in batch mode.

Related Commands:

COM	(Create Operator Message)
KOM	(Kill Operator Messages)
LOM	(List Operator Messages)
QOI	(Quit Operator Interface)
ROM	(Receive Operator Messages)
ROR	(Respond to Operator Interface Request)
XOI	(Execute Operator Interface)

KT (KILL TASK)

4.96 KT (KILL TASK)

Prompts:

```
KILL TASK
                    RUN ID:  integer          (*)
          STATION NUMBER:  integer          (*)
```

Purpose:

The KT command terminates an active task. Tasks which perform end action are allowed to perform the end action prior to termination. The state of the task is displayed on your terminal at the time it is terminated.

Tasks in the following states either proceed to end-action and terminate or terminate immediately after a KT command is executed:

00	Task awaiting time slice allotment or active
01	Task awaiting memory
04	Task is terminated
05	Task in time delay
06	Task unconditionally suspended
09	Task suspended for I/O
0F	Task suspended for abort I/O
17	Task awaiting coroutine activation
19	Task has exceeded initiate mode I/O threshold
24	Task terminated awaiting service queue input
30	Task awaiting system table area
36	Waiting for any I/O SVC
3D	Task suspended for semaphore
42	Waiting for event completion

Tasks in the following states do not terminate when the KT command is executed until converted to one of the previously listed states:

02	Job is in nonexecutable state
03	Task awaiting task load
14	Task awaiting overlay loader services
1E	Task is waiting for access to door (usually locked directory)
1F	Waiting for Scheduled Bid Task SVC
20	Waiting for Install Volume SVC
22	Waiting for Disk Manager SVC
25	Waiting for Install Task SVC
26	Waiting for Install Procedure SVC
27	Waiting for Install Overlay SVC
28	Waiting for Delete Task SVC
29	Waiting for Delete Procedure SVC
2A	Waiting for Delete Overlay SVC
2B	Waiting for Bid Task SVC
2D	Waiting for Read/Write Task SVC
31	Waiting for Map Program Name to ID SVC
34	Waiting for Unload Volume SVC
37	Waiting for Assign Program File Space SVC
38	Waiting for Initialize New Volume SVC
40	Waiting for Segment Manager SVC
43	Waiting for Name Manager SVC
48	Waiting for Job Manager SVC
4A	Waiting for Forced Roll SVC
4C	Waiting for Return Code Processor

Prompt Details:

RUN ID:
The run-time ID assigned to the task at the time it was activated.

STATION NUMBER:
The station number of the station with which the task is associated when the task is activated. Enter >FF if the task is not associated with any station. Zero specifies not to validate the station ID. For channel owner tasks bid by an Assign LUNO to the channel, you must specify either >FF or zero to kill the task.

KT (KILL TASK)

Example:

In the following example, the KT command terminates a task with a run-time ID of >5C that is associated with station number 4 (the station with device name ST04). The task is terminated and a message is printed indicating that the task was awaiting the completion of I/O (task state 09).

```
[ ] KT

KILL TASK
                RUN ID:  >5C
        STATION NUMBER:  4

TASK STATE=>9
```

Messages:

After you execute the KT command, a message is displayed indicating what task state the task was in when it was terminated. This message is displayed on your terminal local file.

Notes:

Tasks that are killed cannot reset their end-action status with a Reset End Action SVC.

Related Commands:

KBT	(Kill Background Task)
STS	(Show Task Status for < user ID>)

4.97 LAG (LIST ACCESS GROUPS)

Prompts:

```
LIST ACCESS GROUPS
                    PASSCODE:   – character(s)
          LISTING ACCESS NAME:  [site:][pathname]@
```

Purpose:

The LAG command produces a list of those access groups of which your user ID is a member. The list also indicates which groups you are the leader of, and shows your current creation access group. If your user ID is a member of the SYSMGR group, the LAG command shows all access groups in the system.

Prompt Details:

PASSCODE:
Enter your assigned log-on passcode.

LISTING ACCESS NAME:
The pathname of a device or file to which the list of access groups is written. Since the information being listed is security related, you should give consideration to the degree of security afforded the device which is to receive the listing. The default value is the terminal local file.

LAG (LIST ACCESS GROUPS)

Example:

In the following example, a user has an assigned passcode of P89J55 and user ID of DEBBIE. The LAG command lists all access groups of which DEBBIE is a member to the terminal local file. (The passcode is not echoed to the user's terminal.)

```
[] LAG

LIST ACCESS GROUPS
                PASSCODE:
   LISTING ACCESS NAME:


   LIST OF ACCESS GROUPS FOR DEBBIE
   --------------------------------
 * PUBLIC

   CLERKS
   COWBOYS              LEADER
   HAWKS
   GROUP1
   GROUP2
   RAIDERS              LEADER

   * - FILE CREATION ACCESS GROUP

16:19:33  TUESDAY  MARCH 22, 1983
```

Related Commands:

SCAG (Set Creation Access Group)

4.98 LAGFR (LIST ACCESS GROUP FILE RIGHTS)

Prompts:

```
LIST ACCESS GROUP FILE RIGHTS
                    PASSCODE:  – character(s)
                    PATHNAME:  pathname@
          ACCESS GROUP NAME:   alphanumeric
        LISTING ACCESS NAME:   [site:][pathname]@
       EXECUTION MODE(F,B):    {F/FOREGROUND/B/BACKGROUND} (FOREGROUND)
```

Purpose:

For a specified access group, the LAGFR command lists the files that have access rights defined for that group. For each file listed, it also displays which access rights the group has. The user must be a member of the access group specified for LAGFR, or the user must be a member of the SYSMGR access group.

Prompt Details:

PASSCODE:
Enter your assigned log-on passcode.

PATHNAME:
Your response can be a file, directory, volume name, or disk name. LAGFR will examine all files in the specified PATHNAME to see which have access rights defined for the given access group. If no files have access rights defined, the output listing will include a message to that effect.

ACCESS GROUP NAME:
Specify the name of the access group for which access is to be checked. A user issuing this command must be a member of the specified access group or must be a member of the system manager access group, SYSMGR. The output shown will include only the access available by virtue of belonging to this access group; no other rights (such as those of PUBLIC) will be included.

If you specify the access group SYSMGR, no list of files is generated; a message is shown saying that SYSMGR can access all files. If you specify PUBLIC, all files explicitly set for access by PUBLIC are shown, along with all files that are not secured.

LISTING ACCESS NAME:
The pathname of a device or file to which the result of LAGFR is written. Since the information being listed is security related, you should give consideration to the degree of security afforded the device which is to receive the listing. The default value is the terminal local file.

EXECUTION MODE(F,B):
If you enter F or FOREGROUND, the command executes in the foreground. If you enter B or BACKGROUND, the command executes in the background. The initial value is foreground. However, in a batch stream this command always executes in background mode.

LAGFR (LIST ASCCESS GROUP FILE RIGHTS)

Example:

In the following example, a user specifies a disk named DS03 and a access group named WORKERS. LAGFR executes in foreground mode and writes the following listing to the terminal local file:

```
[] LAGFR

LIST ACCESS GROUP FILE RIGHTS
                    PASSCODE:
                    PATHNAME: DS03
          ACCESS GROUP NAME: WORKERS
        LISTING ACCESS NAME:
        EXECUTION MODE(F,B): FOREGROUND


           LIST OF ACCESS GROUP FILE RIGHTS
           --------------------------------

   FILES FROM: DS03
 ACCESS GROUP: WORKERS

 RIGHTS      PATHNAME
 ---------   --------
 R W D E C   DS03.A.B
         C   DS03.A.C.TEST1.PRACTICE
 R W         DS03.A.C.TEST1.SAMPLE.EXAMPLE
 R     E     DS03.A.C.TEST1.SAMPLE.REHEARSE.TEMPDIRC.FILELONG
 R     E     DS03.TEST.TEST.TEST.TEST

 R = READ ACCESS
 W = WRITE
 D = DELETE
 E = EXECUTE
 C = CONTROL

14:15:45 MONDAY, APRIL 25, 1983
```

Related Commands:

LAG	(List Access Groups)
LAGM	(List Access Group Members)
LSAR	(List Security Access Rights)

4.99 LAGM (LIST ACCESS GROUP MEMBERS)

Prompts:

```
LIST ACCESS GROUP MEMBERS
                    PASSCODE:  – character(s)
           ACCESS GROUP NAME:  alphanumeric
       LISTING ACCESS NAME:  [site:][pathname]@
```

Purpose:

The LAGM command produces a list of all users which are members of the specified access group. The user of this command must be the leader of the specified access group or must be a member of the SYSMGR access group.

Prompt Details:

PASSCODE:
Enter your assigned log-on passcode.

ACCESS GROUP NAME:
Specify the name of the access group to be listed.

LISTING ACCESS NAME:
The pathname of a device or file to which the list of members is written. Since the information to be listed is security related, you should give consideration to the degree of security afforded the device which is to receive the listing. The default value is the terminal local file.

LAGM (LIST ACCESS GROUP MEMBERS)

Example:

In the following example, a user has an assigned passcode of P89J55, a user ID of DEBBIE, and is the leader of the access group named PROJECT2. The LAGM command lists the user IDs of all PROJECT2 access group members to the terminal local file. (The passcode is not echoed to the user's terminal.)

```
[] LAGM

LIST ACCESS GROUP MEMBERS
                PASSCODE:
      ACCESS GROUP NAME: PROJECT2
     LISTING ACCESS NAME:


    USERS IN ACCESS GROUP PROJECT2
    ------------------------------

    BARBARA
    DEBBIE              LEADER
    JIM
    JOYCE
    MICHAEL

  10:38:21  WEDNESDAY  MARCH 23, 1983
```

Related Commands:

MAG (Modify Access Group)

4.100 LB (LIST BREAKPOINTS)

Prompts:

```
LIST BREAKPOINTS
            RUN ID:  integer     (*)
```

Purpose:

The LB command displays the breakpoints for a specified task. If the breakpoints are to be displayed for a system task, you must have a user ID with a privilege level of two or higher.

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

Example:

In the following example, the LB command displays the breakpoints for a task with a run ID of >4C.

```
[] LB

LIST BREAKPOINTS
            RUN ID: >4C
```

Related Commands:

AB	(Assign Breakpoint)
DB	(Delete Breakpoints)

LBP (LIST BREAKPOINTS — PASCAL)

4.101 LBP (LIST BREAKPOINTS — PASCAL)

The LBP command lists the breakpoints assigned for a Pascal task. Refer to the *DNOS TI Pascal Programmer's Guide* for further information on the LBP command and the following prompt:

```
LIST BREAKPOINTS – PASCAL
          RUN ID:  integer exp      (*)
```

4.102 LD (LIST DIRECTORY)

Prompts:

```
LIST DIRECTORY
                    PATHNAME:   [site:]pathname@
        LISTING ACCESS NAME:    [site:][pathname]@
```

Purpose:

The LD command lists the names of all files, channels, aliases, and subdirectories in a directory. The names of files within subdirectories are not listed. The LD command cannot process directories with 3000 or more used entries.

Table 4-9 describes the fields of the LD command output.

Prompt Details:

PATHNAME:
The pathname that identifies the directory with elements to be listed. You can specify the pathname by a synonym, synonym followed by a pathname (synonym.pathname), logical name, or logical name followed by a pathname (logical name.pathname). If you use a volume name or device name, the VCATALOG directory for the resource is listed.

LISTING ACCESS NAME:
The pathname of the device or file where the output of the directory file names and subdirectory names will be listed. The default value is the terminal local file of the terminal.

Specify the pathname by a synonym, synonym followed by a pathname (synonym.pathname), logical name, or logical name followed by a pathname (logical name.pathname).

LD (LIST DIRECTORY)

Example:

In the following example, the file names and subdirectory names of directory SYS2.KC0017 are listed to the terminal local file. The synonym KC has been assigned to the directory pathname.

```
[] LD

LIST DIRECTORY
                PATHNAME: KC
     LISTING ACCESS NAME:

DIRECTORY LISTING OF:  DS07.VCATALOG
  MAX # OF ENTRIES: 347    # OF ENTRIES AVAILABLE: 334

 DIRECTORY  ALIAS OF  ENTRIES    LAST UPDATE          CREATION
  S$SYSLIB      *        29    02/02/83  17:49:45    02/02/83  17:49:42
  S$SYSTEM      *        23    02/02/83  17:50:03    02/02/83  17:49:46
  VCATALOG      *       347    02/11/83  14:30:83    01/24/82  15:21:40

 FILE        ALIAS OF  RECORDS    LAST UPDATE       FMT  TYPE   BLK PROTECT
  CDATA          *       200    01/24/82  11:25:52  BS   N KEY  YES
  FDATA          *        65    02/04/83  15:25:52  BS   N SEQ  YES
  S$DIAG         *        40    01/24/82  15:21:40  NBS  C REL  NO   WRT DEL
  S$IPL          *        29    02/09/83  11:03:12  NBS  C ING  NO
  S$SHIP         *       595    02/09/83  11:08:12  NBS  N PRO  NO
  S$UTIL         *      4321    02/04/83  10:13:49  NBS  N PRO  NO   DEL

  CHANNEL     PRG FILE   ID   TYPE  SCOPE RESOURCE ASSIGNS  SHARED  MAX
  S$DSTCHN   S$UTIL      4D   M/S     GL    DEV       NO      YES   362
  S$MAIL     S$UTIL       7   M/S     GL    DEV       NO      YES   348
  S$SPOOL    S$UTIL      5B   M/S     TL    DEV       YES     NO    510
11:50:32 WEDNESDAY, MAR 09, 1983.
```

Related Commands:

CD	(Copy Directory)
CFDIR	(Create Directory File)
DD	(Delete Directory)

Table 4.9. LD Command Output Field Descriptions

Field	Description
MAX # OF ENTRIES:	The number of entries that can be contained in this directory. This value is specified when the directory is created via the CFDIR command.
# OF ENTRIES AVAILABLE:	This is the number of entries available in this directory.
DIRECTORY	The name of a subdirectory within the directory being listed.
ALIAS OF	The alias, if any have been assigned, of a subdirectory or file. The alias must be assigned using the AA command.
ENTRIES	The maximum number of entries specified when this subdirectory was created.
LAST UPDATE	The date (month, day, and year) and time (hour, minute, and second) when the last modification was performed. This field applies to updates of a subdirectory in the listed directory or file in the listed directory.
CREATION	The date (month, day, and year) and time (hour, minute, and second) when the subdirectory was created.
FILE	The name of a file in the listed directory.
RECORDS	The number of records in a file.
FMT	The format of a file. The format of a file is NBS (nonblank-suppressed) or BS (blank-suppressed).
TYPE	This field specifies if a file is noncontiguous (N) or contiguous (C) and the file type (SEQ, REL, PRO, IMG, or KEY).
BLK	This field specifies if the file is blocked (YES) or unblocked (NO).
PROTECT	This field specifies if the file is write (WRT) protected and/or delete (DEL) protected.
CHANNEL	The name of a channel within the directory being listed.
PRG FILE	The program file in which the channel owner task resides.
ID	The owner task ID in the program file.
TYPE	Specifies if this is a Symmetric (SYM) or Master/Slave (M/S) channel.

LD (LIST DIRECTORY)

Table 4-9. LD Command Output Field Descriptions (Continued)

Field	Description
SCOPE	Indicates that the channel is global (GL), job-local (JL), or task local (TL).
RESOURCE	Indicates whether the channel is a simple channel or what type of device or file it is simulating.
ASSIGNS	If the owner of a master/slave channel processes assigns, a YES appears in this column; otherwise, a NO is output.
SHARED	YES indicates that the channel can be shared by several requesters at the same time. NO indicates that it cannot be shared.
MAX	The maximum number of bytes allowed in messages sent to the channel.

LDC (LIST DEVICE CONFIGURATION)

4.103 LDC (LIST DEVICE CONFIGURATION)

Prompts:

LIST DEVICE CONFIGURATION
LISTING ACCESS NAME: [site:][pathname]@

Purpose:

The LDC command lists the devices generated for the system by the Execute System Generation Utility (XSGU) command and any devices added to the system by the Modify Device Configuration (MDC) command.

The information returned by LDC is the software configuration of the executing system, unless a System Configuration Utility (SCU) session is in progress. Refer to the Execute System Configuration Utility (XSCU) for information concerning the SCU session.

Prompt Details:

LISTING ACCESS NAME:
The device or file name to which the information is to be returned. The default value is the terminal local file.

Example:

In the following example, the LDC command displays the devices in the running system to the terminal local file.

```
[] LDC

LIST DEVICE CONFIGURATION
     LISTING ACCESS NAME:
```

NAME	DEVICE TYPE	CRU/ TILINE	MUX CHANNEL	INTERRUPT	CHASSIS	POSITION	MODE	8-BIT CHARS
CR01	CARD READER	0040		4			ON	NO
DS01	TILINE DISK	F800		13			ON	NO
DS02	TILINE DISK	F800		13			ON	NO
DS03	TILINE DISK	F810		12			OFF	NO
LP01	LP: SERIAL	0460		7	1	14	SPLR	YES
LP04	LP: SERIAL	FA00	2	15			ON	NO
MT01	MAG TAPE	F880		9			ON	NO
ST01	ASR - 733	0000		6			ON	NO
CS01	CASSETTE	0000		6			ON	NO
CS02	CASSETTE	0000		6			ON	NO
ST02	VDT - 911	0400		7	1	16	ON	NO
ST03	VDT - 911	0420		7	1	17	ON	NO
ST05	VDT - 931	FA00	1	15			ON	NO
LP03	LP: SERIAL	FA00	1	15			ON	NO
ST06	VDT - 931	FA00	3	15			ON	NO

LDC (LIST DEVICE CONFIGURATION)

Notes:

The following device modes are supported:

Mode	Definition
DIAG	Device in use for diagnostic tasks
ON	Device is online and available — Line printers used as SHARED or REMOTE devices must have mode set to ON, not SPLR
OFF	Device is offline and unavailable
SPLR	Device is allocated to spooler and may only be used for output

Related Commands:

MDC	(Modify Device Configuration)
MDS	(Modify Device State)
XSCU	(Execute System Configuration Utility)

LHPC (LIST HARDCOPY TERMINAL PORT CHARACTERISTICS)

4.104 LHPC (LIST HARDCOPY TERMINAL PORT CHARACTERISTICS)

Prompts:

```
LIST HARDCOPY TERMINAL PORT CHARACTERISTICS
        OUTPUT ACCESS NAME:  [site:][pathname]@
```

Purpose:

The LHPC command enables you to obtain a table of the port characteristics for all hardcopy teleprinter device (TPD) terminal ports.

Prompt Details:

OUTPUT ACCESS NAME:
The response to OUTPUT ACCESS NAME can be a file name or device name. If you do not enter a response, the table is displayed on your terminal local file.

Example:

In the following example, all TPD port characteristics will be displayed at the terminal defined as ST01 during system generation.

```
[] LHPC

LIST HARDCOPY TERMINAL PORT CHARACTERISTICS
     OUTPUT ACCESS NAME: ST01
```

Messages:

Completion messages are displayed at the controlling station.

Related Commands:

ANS	(Answer Incoming Call)
CALL	(Call Terminal)
DISC	(Terminal Disconnection)
MHPC	(Modify Hardcopy Terminal Port Characteristics)

LJ (LIST JOBS)

4.105 LJ (LIST JOBS)

Prompts:

```
LIST JOBS
                JOB NAME OR ID:   [{alphanumeric/integer}]
              TASK INFORMATION?:  YES/NO                  (NO)
          LISTING ACCESS NAME:    [site:][pathname]@
```

Purpose:

The LJ command allows the system operator to list the status of a particular job or all jobs currently executing in the system. You can also list status of tasks associated with the job(s). Users other than the system operator can list only jobs and task information associated with their user IDs. Refer to the Show Task Status for < user ID> (STS) command for information concerning displayed status of tasks.

Prompt Details:

JOB NAME OR ID:
The job name specified at job creation or the job ID returned when the Show Job Status (SJS) or Execute Job (XJ) command was executed. If you enter a null value, the status of all jobs currently executing is listed.

TASK INFORMATION?:
If you specify YES, the status of the tasks running under the listed jobs are also listed.

LISTING ACCESS NAME:
The device name or file name to which SCI displays the job(s) information. The default value is the terminal local file.

LJ (LIST JOBS)

Example:

In the following example, the status of jobs currently executing under a user's ID and status of tasks associated with the jobs are listed to the operator's terminal by entering the LJ command as follows:

```
[] LJ

LIST JOBS
          JOB NAME OR ID: STRESS
       TASK INFORMATION?: YES
     LISTING ACCESS NAME:

USER-ID   JOB-NAME   ID    PRI  STATE  JCASIZE  CURRENT  MAXIMUM
STRESS    SORT       000D   20    2      6144     2394     2844

      TASK     ID RUN-ID STN  PRI R-PRI STATE WP    PC   FLAG1 FLAG2  CPU SECS
      SORTMG   2A   1A        3   197   09   0006  26DC  3000  0000       0.0
      FILEMGR  05   02        0   0     24   906C  C05C  D100  0000       7.1
      SCI990   01   01       B4   192   17   89EE  4040  1000  0000       3.9
      SCI990   01   14       B4   202   17   89EE  4040  1000  0800       1.5
      TIME1    03   18       B4   241   09   5DA0  5B26  1000  0800      11.7
      SORTMG   2A   19        3   224   09   0006  041C  3000  0000      11.7

USER-ID   JOB-NAME   ID    PRI  STATE  JCASIZE  CURRENT  MAXIMUM
STRESS    COMPARE    000B   20    5      6144     2394     2844

      TASK     ID RUN-ID STN  PRI R-PRI STATE WP    PC   FLAG1 FLAG2  CPU SECS
      FILEMGR  05   02        0   0     02   75CA  0264  D101  0000     403.5
      SCI990   01   01       B4   194   17   89EE  4040  1000  0000       2.5
      SCI990   01   0A       B4   201   17   89EE  4040  1000  0800       2.4
      SCI990   01   2D       B4   199   17   89EE  4040  1000  0800       8.2
      XCP      8A   40       B4   228   09   70BC  3684  1000  0400       5.6
```

Related Commands:

HJ	(Halt Job)
KJ	(Kill Job)
RJ	(Resume Job)
SJS	(Show Job I/O Status for < user ID>)
STS	(Show Task Status for < user ID>)
XBJ	Execute Batch Job)
XJ	(Execute Job)

LLN (LIST LOGICAL NAMES)

4.106 LLN (LIST LOGICAL NAMES)

Prompts:

LIST LOGICAL NAMES
LISTING ACCESS NAME: [site:][pathname]@

Purpose:

The LLN lists the logical names currently assigned and the corresponding pathname(s) for each logical name. The job local names are listed first, followed by the global logical names.

Prompt Details:

LISTING ACCESS NAME:
The device or file name to which the system lists the logical names. If you enter a null response, the names are displayed on the terminal local file.

LLN (LIST LOGICAL NAMES)

Example:

In the following example, the LLN command lists the currently assigned logical names to the terminal local file:

```
[] LLN

LIST LOGICAL NAMES
     LISTING ACCESS NAME:


                  JOB LOCAL LOGICAL NAMES

        LOGICAL NAME            PATHNAME(S)

          PDWSVOL               DS01

          SLOW                  .S$SPOOL
                              SPOOLER PARAMETERS
                                        ANSI FORMAT: N
                                      BANNER SHEET: N
                                NUMBER OF LINES/PAGE: 62
                                    NUMBER OF COPIES: 1
                                                FORM: STANDARD
                                   DEVICE/CLASS NAME: LP01
                                SPOOLER LOGICAL NAME: SLOW

                     GLOBAL LOGICAL NAMES

        LOGICAL NAME            PATHNAME(S)

          S$COBOL               .S$LANG

          S$DBMS                .S$DBMS

          S$QUERY               .S$QUERY

          S$TIP                 .TIP

          TIPE                  .S$TIPE
```

Related Commands:

ALN	(Assign Logical Names)
RLN	(Release Logical Names)
SGND	(Snapshot Global Name Definition)

LLR (LIST LOGICAL RECORD)

4.107 LLR (LIST LOGICAL RECORD)

Prompts:

```
LIST LOGICAL RECORD
                    PATHNAME:  [site:]pathname@      (*)
              STARTING RECORD:  integer              (0)
            NUMBER OF RECORDS:  [integer]
          LISTING ACCESS NAME:  [site:][pathname]@
      MAXIMUM RECORD LENGTH:  [integer]
```

Purpose:

The LLR command lists the contents of a record or records in a file. The contents of the record or records specified are listed in both hexadecimal and ASCII representation. The amount displayed per record is decimal 512 (hexadecimal 200) or the logical record length of the file, whichever is less.

Prompt Details:

PATHNAME:
The pathname that identifies the file in which the records to be listed reside.

STARTING RECORD:
A decimal or hexadecimal integer that dentifies the first record that has contents to be listed.

NUMBER OF RECORDS:
A decimal or hexadecimal integer that identifies how many records are to be listed. A null response specifies that all records are to be listed.

LISTING ACCESS NAME:
The device name of a device or the pathname of a file to which the LLR command should write the contents of the record(s) specified. The default value is the terminal local file.

MAXIMUM RECORD LENGTH:
A decimal or hexidecimal integer which specifies the size in bytes of the largest record you anticipate in the file being listed. The default value is 512 bytes.

LLR (LIST LOGICAL RECORD)

Example:

In the following example, the contents of the first three records of a file are listed to the user's terminal. The word SAME in the example indicates that the last word printed on the preceding line is the same as all following words until the end of record is reached or until a different word is encountered. The last word of each record is always printed.

The responses to the LLR command prompts and a listing of the contents of the record are shown below:

```
[] LLR


LIST LOGICAL RECORD
                PATHNAME: SYS2.KC0017.LLR
          STARTING RECORD: 0
        NUMBER OF RECORDS: 3
      LISTING ACCESS NAME:
    MAXIMUM RECORD LENGTH:


FILE ACCESS NAME:  .KC0047.SCI.LLR
RECORD:  000000
0000  2F53 4554 204F 5248 3D7E 4C4C 5220 284C    /S ET  O RH =~ LL R  (L
0010  4953 5420 4C4F 4749 4341 4C20 5245 434F    IS T  LO GI CA L  RE CO
0020  5244 297E 2020 2020 2020 2020 2020 2020    RD )~
  SAME
004E  2020
RECORD:  000001
0000  2F53 4554 2045 4C48 3D7E 4C4C 5220 284C    /S ET  E LH =~ LL R  (L
0010  4953 5420 4C4F 4749 4341 4C20 5245 434F    IS T  LO GI CA L  RE CO
0020  5244 297E 2020 2020 2020 2020 2020 2020    RD )~
  SAME
004E  2020
RECORD:  000002
0000  2F50 4220 2020 2020 2020 2020 2020 2020    /P B
  SAME
004E  2020
```

LM (LIST MEMORY)

4.108 LM (LIST MEMORY)

Prompts:

```
LIST MEMORY
                        RUN ID: integer              (*)
              STARTING ADDRESS: full exp
               NUMBER OF BYTES: [full exp]
          LISTING ACCESS NAME: [site:][pathname]@
```

Purpose:

The LM command lists the specified memory area of a task to a specified output device or file. If the task is not unconditionally suspended, it is temporarily suspended while the listing is being formatted.

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

STARTING ADDRESS:
The integer value that is the starting word address of the memory area to be listed.

NUMBER OF BYTES:
The integer value that is the number of bytes of memory to be listed, beginning with the specified starting word address. This value will be rounded up to the nearest multiple of 16. The default value is 16 bytes.

LISTING ACCESS NAME:
The device name or file name of the device or file where the memory list is to be output. The default value is the terminal local file.

LM (LIST MEMORY)

Example:

In the following example, the LM command lists the memory area, beginning at address >102, of a task with a run ID of >80. The number of bytes specified, >14A, is listed to the file name MY.OUTPUT

```
[] LM


LIST MEMORY
                    RUN ID: >80
          STARTING ADDRESS: >102
           NUMBER OF BYTES: >14A
      LISTING ACCESS NAME: MY.OUTPUT
```

Related Commands:

LSM	(List System Memory)
MIR	(Modify Internal Registers)
MM	(Modify Memory)
MWR	(Modify Workspace Registers)
SP	(Show Panel)

LOM (LIST OPERATOR MESSAGES)

4.109 LOM (LIST OPERATOR MESSAGES)

Prompts:

```
LIST OPERATOR MESSAGES
        LISTING ACCESS NAME:   [site:][pathname]@
```

Purpose:

The LOM command lists all current outstanding operator messages to a specified output device or file. This command is only valid interactively.

Prompt Details:

LISTING ACCESS NAME:
The device name or the file pathname to which the outstanding operator messages are to be listed. The default is the terminal local file.

Example:

In the following example, the outstanding operator messages are listed to a file .MESSAGES by entering the LOM command as follows:

```
[] LOM

LIST OPERATOR MESSAGES
     LISTING ACCESS NAME: .MESSAGES
```

Assumptions:

The user entered either a Receive Operator Messages (ROM) or Execute Operator Interface Request (XOI) command.

Notes:

If the user has entered the ROM command and another user has entered the XOI command, the user will be shown operator commands without request IDs. This indicates that the user cannot respond to them. If the user entered MY in response to the MESSAGES (MY/ALL) prompt for the ROM command, only messages originated by or for that user are listed.

Related Commands:

COM	(Create Operator Message)
KOM	(Kill Operator Messages)
KOR	(Kill Operator Interface Request)
QOI	(Quit Operator Interface)
ROM	(Receive Operator Messages)
ROR	(Respond to Operator Interface Request)
XOI	(Execute Operator Interface)

4.110 LPS (LIST PASCAL STACK)

The LPS command lists a specified portion of the stack frame for a specified Pascal task. Refer to the *DNOS TI Pascal Programmer's Guide* for further information on the LPS command and the following prompts:

```
LIST PASCAL STACK
                    RUN ID:  integer exp              (*)
              ROUTINE NAME:  [character(s)]
            STARTING OFFSET:  integer exp             (0)
           NUMBER OF BYTES:  [integer exp]
     LISTING ACCESS NAME:  [site:][pathname]@
```

LS (LIST SYNONYMS)

4.111 LS (LIST SYNONYMS)

Prompts:

LIST SYNONYMS
LISTING ACCESS NAME: [site:][pathname]@

Purpose:

The LS command displays a list of all synonyms and synonym values that are currently assigned under a user ID.

Prompt Details:

LISTING ACCESS NAME:
The device name or file name to which the system lists the synonyms and synonym values assigned under a user ID. Specify the pathname by a synonym, synonym followed by a pathname (synonym.pathname), logical name, or logical name followed by a pathname (logical name.pathname). The default value is the terminal local file.

Example:

In the following example, the user enters the LS command to display a list of synonyms and synonym values at a terminal with the device name ST09 as shown below:

```
[] LS

LIST SYNONYMS
     LISTING ACCESS NAME: ST09
```

Notes:

If the synonym table for a user ID is full, the LS command may not work. Use the Modify Synonym (MS) command to delete or display synonyms. The MS command is discussed later in this section.

Related Commands:

AS (Assign Synonym)
MS (Modify Synonyms)

4.112 LSAR (LIST SECURITY ACCESS RIGHTS)

Prompts:

```
LIST SECURITY ACCESS RIGHTS
                    PASSCODE:  – character(s)
                   FILE NAME:  filename@
          LISTING ACCESS NAME:  [site:][pathname]@
```

Purpose:

The LSAR command displays all access groups and corresponding rights for the specified file. The user of this command must have the control access right for the specified file or be a member of the SYSMGR access group.

Prompt Details:

PASSCODE:
Enter your assigned log-on passcode.

FILE NAME:
Enter the file name of the file for which you want the security access rights listed. An error will be returned if you do not have the control access right for the specified file.

LISTING ACCESS NAME:
The pathname of a device or file to which the list of access rights is written. Since the information to be listed is security related, you should give consideration to the degree of security afforded the device which is to receive the listing. The default value is the terminal local file.

LSAR (LIST SECURITY ACCESS RIGHTS)

Example:

In the following example, a user has an assigned passcode of P89J55 and has the control access right to the file named VOL2.BATCH.SALES1. The LSAR command lists all access groups and corresponding rights associated with the file VOL2.BATCH.SALES1 to the terminal local file. (The passcode is not echoed to the user's terminal.)

```
[] LSAR

LIST SECURITY ACCESS RIGHTS
             PASSCODE:
           FILE NAME: VOL2.BATCH.SALES1
  LISTING ACCESS NAME:

SECURITY ACCESS RIGHTS FOR VOL2.BATCH.SALES1

ACCESS GROUP            ACCESS RIGHTS
-------------------------------------------------------
  PUBLIC              READ  WRITE  DELETE

  CLERKS              READ  WRITE
  RAIDERS             READ  WRITE
  HAWKS                             DELETE  EXECUTE
  COWBOYS             READ  WRITE  DELETE  EXECUTE  CONTROL
  GROUP1              READ  WRITE          EXECUTE
  GROUP2              READ  WRITE  DELETE

09:58:45  THURSDAY  MARCH 24, 1983
```

Related Commands:

LAG	(List Access Groups)
LAGM	(List Access Group Members)
MSAR	(Modify Security Access Rights)

4.113 LSB (LIST SIMULATED BREAKPOINTS)

Purpose:

The LSB command displays all current simulated breakpoints.

The first column of the breakpoints display lists the numbers assigned when the breakpoints were set; the numbers start at one and are consecutive. The TYPE column lists letters for the ON prompt of an Assign Simulated Breakpoint command to identify the value on which the breakpoint was set, and the FROM and THRU columns list the range of values which apply. The COUNT column lists the count operand entered when the breakpoints were set, and the REMAINING column lists the number of remaining times the program must go through the breakpoint. The DISPLAY column lists the address to be displayed when the task is halted at the breakpoint.

All values, except the breakpoint number and type, are listed as hexadecimal numbers.

Example:

In the following example, the LSB command lists all active simulated breakpoints.

```
[]LSB

LIST SIMULATED BREAKPOINTS
1  TYPE=P FROM=>000A THRU=>0064 COUNT=>000A REMAINING=>000A DISPLAY=>0021
2        S       >0016       >0038       >0058           >0058          >0009
3        C       >0016       >0078       >0037           >0037          >0000
4        A       >0016       >2710       >00C8           >00C8          >0000
5        R       >0000       >2710       >002C           >002C          >0000
```

Related Commands:

ASB	(Assign Simulated Breakpoint)
DSB	(Delete Simulated Breakpoints)
ST	(Simulate Task)
XD	(Execute in Debug Mode)

LSC (LIST SOFTWARE CONFIGURATION)

4.114 LSC (LIST SOFTWARE CONFIGURATION)

Prompts:

```
LIST SOFTWARE CONFIGURATION
HISTORY FILE VOLUME NAME:   [site:][pathname]@
      LISTING ACCESS NAME:  [site:][pathname]@
```

Purpose:

The LSC command lists the most recent modifications or updates for each unique software product that is currently documented in the system history file (<volume name>.S$SYSTEM.S$HSTRY).

Software package installation and patch batch streams supplied by your software representative follow a convention of updating the history file on the running system disk or, if specified, on a target system disk. If you do not modify the history file, it functions as a valuable modification record to assist your system manager and customer support personnel in isolating system software problems.

Prompt Details:

HISTORY FILE VOLUME NAME:
The name of the disk that contains the pathname .S$SYSTEM.S$HSTRY.

LISTING ACCESS NAME:
The device or file name to which the system will list the Software Configuration summary. The pathname can be specified by synonym, a synonym followed by a pathname (synonym.pathname), a logical name, or a logical name followed by a pathname (logical name.pathname). A null response specifies the terminal local file.

LSC (LIST SOFTWARE CONFIGURATION)

Example:

In the following example, the software history file generates an output summary. The name of the running system is SEP23 and the system utility program file is .S$UTIL. Table 4-10 describes the information output to the device or file specified in response to the LISTING ACCESS NAME prompt.

```
[] LSC

LIST SOFTWARE CONFIGURATION
 HISTORY FILE VOLUME NAME: SYSVOL
      LISTING ACCESS NAME:


              S O F T W A R E     U P D A T E     P R O F I L E

                                                RELEASE    INSTALL     PATCH    LAST
  SOFTWARE PACKAGE                  REVISION     DATE       DATE       DATE    PATCH

DNOS DBMS-990                        2.2.0    07/09/82                05/20/82  00000
DNOS TIFORM                          2.1.0    09/20/81   05/20/82     05/20/82  1480
DNOS (SEP23 KERNEL)                  1.1.1    01/13/83   01/15/83     02/16/83  2650
DNOS (SEP23 UTILITY)                 1.1.1    01/13/83                02/10/83  2632


11:06:10 THURSDAY, MAY 19, 1983.
```

Notes:

The DNOS operating system listed and the utility program file listed are for the current running system (whether they are the last ones listed in the .S$SYSTEM.S$HSTRY file or not). All other package names listed reflect the last updated version encountered whenever the history file is read from the beginning to the end of the file.

Text edit or delete outdated records from the beginning of the history file to purge outdated information from the history file. Take care not to delete records for products for which the information could be useful.

The commands to update the history file are documented in the *DNOS Systems Programmer's Guide*.

LSC (LIST SOFTWARE CONFIGURATION)

Table 4-10. LSC Output Field Description

Field	Description
SOFTWARE PACKAGE	This is the name of the unique software package.
REVISION	This is the last RELEASE NUMBER= encountered for the given SOFTWARE PACKAGE name.
RELEASE DATE	This is the last RELEASE DATE= encountered for the given SOFTWARE PACKAGE name.
INSTALL DATE	This is the last TEST GENERATED SYSTEM or the last INSTALLATION COMPLETED date encountered for the given SOFTWARE PACKAGE name.
PATCH DATE	This is the last PATCHING COMPLETED date encountered for the given SOFTWARE PACKAGE name.
LAST PATCH	This is the last LAST PATCH= identification (Pxxxx) encountered for the given SOFTWARE PACKAGE name.

4.115 LSM (LIST SYSTEM MEMORY)

Prompts:

```
LIST SYSTEM MEMORY
          OVERLAY NAME OR ID:  {integer/alphanumeric}
            STARTING ADDRESS:  integer
             NUMBER OF BYTES:  [integer]                (040)
          LISTING ACCESS NAME:  [site:][pathname]@
```

Purpose:

The LSM command lists the memory occupied by the operating system; this command is similar to the List Memory (LM) command except you must specify an overlay name or ID instead of a run ID.

The LSM command is intended for use only by someone very familiar with the operating system source.

LSM cannot be used to display secondary system tables. The Execute Crash Analysis Utility (XANAL) command should be used to display secondary system tables such as Segment Manager Table area.

Prompt Details:

OVERLAY NAME OR ID:
The overlay name or overlay ID specified in the Install Overlay (IO) command of the overlay that has memory to be listed. By executing the Map Program File (MPF) command on the kernel program file whose name is specified at system generation, you can inspect the acceptable overlay names and associated IDs.

STARTING ADDRESS:
The integer expression that is the starting word address of the memory area to be listed.

NUMBER OF BYTES:
The integer value that is the number of bytes of memory to be listed, beginning with the specified starting word address. This value is rounded up to the nearest multiple of 16. The initial value is > 40. If no value is given, 16 bytes are listed.

LISTING ACCESS NAME:
The device name or file name where the memory list is to be output. The default value is the terminal local file.

LSM (SIST SYSTEM MEMORY)

Example:

In the following example, the LSM command lists the memory area from byte > C000 to byte > C020 for the scheduler overlay to the terminal local file.

```
[] LSM

LIST SYSTEM MEMORY
              OVERLAY ID:  SVCSHD
        STARTING ADDRESS:  >C000
         NUMBER OF BYTES:  >20
     LISTING ACCESS NAME:


C000    0720  3096  020A  3248  C260  2E16  1379  AA60   .  0. .. 2H .' .. .y .'
C010    3116  001E  1702  05A9  001C  C0E0  2D14  1309   1. .. .. .. .. .. -. ..
```

Related Commands:

LM	(List Memory)
LSM	(List System Memory)
MIR	(Modify Internal Registers)
MM	(Modify Memory)
MSM	(Modify System Memory)
MWR	(Modify Workspace Registers)

4.116 LTS (LIST TERMINAL STATUS)

Prompts:

```
LIST TERMINAL STATUS
                TERMINAL NAME:  [stationname]@
          OUTPUT ACCESS NAME:   [site:][pathname]@
```

Purpose:

The LTS command displays the attributes assigned to any one terminal or the attributes of all terminals in the system and shows the user ID of users currently using the terminal.

Table 4-11 describes the status information given under each heading of the output of the LTS command.

Prompt Details:

TERMINAL NAME:
The station name of the terminal with the status to be displayed. The default value is a null response, which displays the status of all terminals.

OUTPUT ACCESS NAME:
The name of the device or file to which the LTS command is to display or list the results. The default value is the terminal local file.

Example:

In the following example, the LTS command lists the status of terminal ST07 to the terminal local file:

```
[] LTS

LIST TERMINAL STATUS
          TERMINAL NAME: ST07
     OUTPUT ACCESS NAME:

TERMINAL  USER ID   LOGON REQUIRED  MODE  DEFAULT
  ST07    KEITH           Y          TTY    VDT
  22:35:32 TUESDAY, JUNE 08, 1982.
```

LTS (LIST TERMINAL STATUS)

Notes:

It is possible for an entry to appear for a terminal that is not currently online. This can happen in either of the following ways:

- Someone has executed the Execute System Configuration Utility (XSCU) command to delete the terminal from the system configuration.
- Someone has executed the Modify Terminal Status (MTS) command to assign characteristics to a terminal that never existed.

The listing of this entry can be suppressed by executing the MTS command for that terminal and responding OFF to the NEW STATUS prompt.

When LTS is used in a file security environment, you must have read access rights to the .S$SCA file.

Related Commands:

MTS (Modify Terminal Status)

Table 4-11. Status Information Output by the LTS Command

Heading	Description
TERMINAL	The device name of the terminal(s) with status to be listed.
USER ID	The eight-character ID of the user who is currently logged on at the terminal.
LOGON REQUIRED	Y (yes) indicates that you log on at the terminal to use SCI. N (no) indicates that you can use SCI at the terminal without performing the log-on process.
MODE	TTY indicates that the terminal is currently set for use in TTY mode. VDT indicates that the terminal is currently set for use in video display terminal mode.
DEFAULT	TTY indicates that the terminal is always used in TTY mode unless you change its status by executing the MTS command. VDT indicates that the terminal is always used in video display terminal mode unless you change its status by executing the MTS command.

4.117 LUI (LIST USER IDS)

Prompts:

LIST USER IDS
OUTPUT ACCESS NAME: [site:][pathname]@

Purpose:

The LUI command displays a list of user IDs currently authorized to use the system. The information displayed includes each user ID, the user description for the ID, and the privilege code associated with that user ID.

Prompt Details:

OUTPUT ACCESS NAME:
The device or file name to which the LUI command should list the user IDs and the information associated with each user ID. The default value is the terminal local file.

Example:

In the following example, the LUI command lists all existing user IDs and their associated user descriptions and privilege codes within a system to the terminal local file.

```
[] LUI

LIST USER IDS
     OUTPUT ACCESS NAME:

  USER ID          DESCRIPTION          PRIVILEGE CODE
  GEORGIA     EXAMPLE USER                    4
  SUPERMAN    HERB                            7
  CLARK1      CLARK JONES                     0
  STAR        KIRBY                           6
  VISITOR     ANY LOCAL ANALYST               7
  SYSTEM00    ANY SYSTEM USER                 3

16:00:21 MONDAY, APRIL 11,1983.
```

Notes:

When LUI is used in a file security environment, you must have read access rights to the .S$CLF file.

Related Commands:

AUI	(Assign User ID)
DUI	(Delete User ID)
MPC	(Modify Passcode)
MUI	(Modify User ID)

MAD (MODIFY ABSOLUTE DISK)

4.118 MAD (MODIFY ABSOLUTE DISK)

Prompts:

```
MODIFY ABSOLUTE DISK
                DISK UNIT:  devicename@              (*)
        OUTPUT ACCESS NAME:  [site:][pathname]@      (*)
                    TRACK:  integer exp              (*)
                   SECTOR:  integer exp              (*)
               FIRST WORD:  integer exp
         VERIFICATION DATA:  [integer exp list]
                     DATA:  integer exp list
```

Purpose:

The MAD command places specified data on a disk at a specified absolute track, sector, and word address and can be executed only by privileged users. Enter data in groups of word values to be placed on disk. Separate each word value from the next with a comma. Values are loaded on disk in successive addresses. The verification parameter allows you to enter a string of words to be compared to the data at the specified address. If a bad compare results, the modification does not take place. Since the MAD command has the capability to write anything anywhere on the disk and can therefore destroy the system image, you should always use the verify option.

Prompt Details:

DISK UNIT:
The device name of the disk device assigned during system generation.

OUTPUT ACCESS NAME:
The device or file name where the contents of the specified absolute disk address are to be printed. The default value is the terminal local file.

TRACK:
The integer value which is the starting track address from which to begin the disk modification.

SECTOR:
The integer value that is the starting sector address, within the specified disk track, from which to begin the disk modification.

FIRST WORD:
he integer value that is the starting word address, within the specified disk sector, from which to begin the disk modification.

MAD (MODIFY ABSOLUTE DISK)

VERIFICATION DATA:

If specified, the integer value contained in the specified starting address. If you specify more than one integer, separate them by commas; it is assumed these values are contained in successive words, beginning with the specified first word.

DATA:

The integer value to replace the existing value contained in the specified first word. If you specify more than one value, separate them by commas; it is assumed these values are to replace the existing values contained in successive words, beginning with the first word.

Example:

In the following example, the MAD command modifies the contents of one word on the disk in disk device DS02, beginning with track 0, sector 0, word 4. The results of the modification are listed on the terminal local file of the terminal where the MAD command is executed.

```
[] MAD

MODIFY ABSOLUTE DISK
               DISK UNIT: DS02
     OUTPUT ACCESS NAME:
                   TRACK: 0
                  SECTOR: 0
              FIRST WORD: 4
      VERIFICATION DATA: 03030
                    DATA: 02020

0004 2020 2020 2610 0205 0120 0001 0000 0000      &. .. .  .. .. ..
```

Related Commands:

SAD (Show Absolute Disk)

MADU (MODIFY ALLOCATABLE DISK UNIT)

4.119 MADU (MODIFY ALLOCATABLE DISK UNIT)

Prompts:

```
MODIFY ALLOCATABLE DISK UNIT
                DISK UNIT:  devicename@              (*)
        OUTPUT ACCESS NAME:  [site:][pathname]@      (*)
                ADU NUMBER:  integer exp             (*)
             SECTOR OFFSET:  integer exp             (*)
                FIRST WORD:  integer exp             (*)
          VERIFICATION DATA: [integer exp list]
                      DATA:  integer exp list
```

Purpose:

The MADU command modifies a portion of a sector specified by allocatable disk unit (ADU) number, sector offset, and word offset within the sector. If verification data does not match the data already on the disk, the modification is not performed.

All disks are addressed in allocatable disk units. The maximum number of ADUs on a disk is 65,535. Therefore, if a disk contains more than 65,535 sectors, multiple sectors are used to form one ADU.

Prompt Details:

DISK UNIT:
The device name of the disk assigned during system generation.

OUTPUT ACCESS NAME:
The device or file name where the results of the ADU modification are to be listed. The default value is the terminal local file.

ADU NUMBER:
The integer value that is the ADU with contents to be modified.

SECTOR OFFSET:
The integer value that is the sector of the ADU with contents to be modified.

FIRST WORD:
The integer value that is the starting word offset, within the specified sector, where modifications of the ADU are to begin.

VERIFICATION DATA:
If specified, the integer value contained in the specified first word address. If you specify more than one integer, separate them by commas; it is assumed these values are contained in successive words, beginning with the specified first word.

MADU (MODIFY ALLOCATABLE DISK UNIT)

DATA:

The integer value to replace the existing value contained in the specified first word. If you specify more than one value, separate them by commas; it is assumed these values are to replace the existing values contained in successive words, beginning with the first word.

Example:

In the following example, the MADU command modifies the contents of one word of the first ADU on the disk in disk device DS02, beginning at sector 0, byte 0. The results of the modification are listed on the terminal local file where the MADU command is executed.

```
[] MADU

MODIFY ALLOCABLE DISK UNIT
               DISK UNIT: DS02
      OUTPUT ACCESS NAME:
                ADU NUMBER: 0
            SECTOR OFFSET: 0
               FIRST WORD: 0
       VERIFICATION DATA: 0
                     DATA: 0100+01000

0000  1100 0000 0000 0000 0000 0000 0000 0000    .. .. .. .. .. .. .. ..
```

Related Commands:

SADU (Show Allocatable Disk Unit)

MAG (MODIFY ACCESS GROUP)

4.120 MAG (MODIFY ACCESS GROUP)

Prompts:

```
MODIFY ACCESS GROUP
                    PASSCODE:  -character(s)
           ACCESS GROUP NAME:  alphanumeric
              ADD USER ID(S):  [alphanumeric...alphanumeric]
           DELETE USER ID(S):  [alphanumeric...alphanumeric]
           NEW LEADER USER ID: [alphanumeric]
```

Purpose:

The MAG command permits you to specify a change in leadership of an access group, add users, and delete users from the group. The user of this command must be the leader of the specified access group or must be a member of the SYSMGR group. If a response is supplied to the NEW LEADER USER ID prompt, the specified user ID is made the new leader for the access group. The current leader loses the leadership rights, but remains a member of the group. Validation is done to ensure that any ID specified for the ADD, DELETE, or NEW LEADER USER ID prompt already exists.

The system determines to which access groups a user belongs when a job is created with the user's ID. Modifications to an access group's membership do not take effect until a new job is created with that user's ID. If currently logged on to SCI, the user must log off and log on again.

Prompt Details:

PASSCODE:
Enter your assigned log-on passcode.

ACCESS GROUP NAME:
Specify the name of the access group to be modified.

ADD USER ID(S):
Enter the user ID(s) to be added as members of the access group. These must be valid, existing user IDs.

DELETE USER ID(S):
Enter the user ID(s) to be deleted from the specified access group. The user IDs must be members of the specified access group.

NEW LEADER USER ID:
Enter the user ID to be the new leader of the specified access group. The specified user ID must be a valid, existing user ID. If the response is a null string, the leader of the group is not changed.

MAG (MODIFY ACCESS GROUP)

Example:

In the following example, the group leader of the access group named PROJECT2 uses the MAG command to add three new users to the group. Users with the user IDs of FRANCES, LOIS, and JIM are added to the access group, PROJECT2. (The passcode is not echoed to the user's terminal.)

```
[] MAG

MODIFY ACCESS GROUP
                PASSCODE:
     ACCESS GROUP NAME: PROJECT2
        ADD USER ID(S): FRANCES,LOIS,JIM
     DELETE USER ID(S):
    NEW LEADER USER ID:
```

Related Commands:

CAG	(Create Access Group)
DAG	(Delete Access Group)
LAGM	(List Access Group Members)

MCC (MODIFY COUNTRY CODE)

4.121 MCC (MODIFY COUNTRY CODE)

Prompts:

```
MODIFY COUNTRY CODE
        COUNTRY NAME:  alphanumeric        (*)
```

Purpose:

The MCC command modifies the country code selected when the system was generated. Table 4-16 lists the legal country codes that can be specified.

WARNING

Incorrect use of this command can cause key indexed files to become nonusable. Modify the country code only if you want the collating sequence of the new country code.

Prompt Details:

COUNTRY:
The characters that specify the country code for the system. The default is the current value in the response field of the prompt. You can specify any of the abbreviations or country names listed in Table 4-12; that is, specify JAPAN or the character J.

Example:

In the following example, the MCC command modifies the country code for a system from its current value to that of Finland (FI).

```
[] MCC

MODIFY COUNTRY CODE
                COUNTRY: FINLAND
```

Assumptions:

A legal country code is entered in response to the COUNTRY prompt. The MCC command must be entered during an Execute System Configuration Utility (XSCU) session and cannot be used to modify the running system.

Related Commands:

SCC	(Show Country Code)
XSCU	(Execute System Configuration Utility)

MCC (MODIFY COUNTRY CODE)

Table 4-12. Country Codes

Abbreviation	Country
AU	Austria
B	Belgium
D	Denmark
FI	Finland
FRA	France
FWP	French Word Processing
G	Germany
J	Japan
N	Norway
SP	Spain
SWE	Sweden
SWI	Switzerland
UK	United Kingdom
USA	United States

MCDT (MODIFY COMMAND DEFINITION TABLE)

4.122 MCDT (MODIFY COMMAND DEFINITION TABLE)

Prompts:

```
MODIFY COMMAND DEFINITION TABLE
                VOLUME NAME:  [site:][alphanumeric]@
                SYSTEM NAME:  alphanumeric
               ADD OR DELETE:  {A/ADD/D/DELETE}      (ADD)
  DEVICE TYPE/CDT NUMBER:  {911/940/931/733/TPD/SD/VT/integer}
```

If the response to the prompt ADD OR DELETE is ADD, the following prompts are displayed:

```
MODIFY COMMAND DEFINITION TABLE
              FIVE WORD ENTRY:  integer,integer,integer,integer,integer
               DEFAULT USER ID:  [character(s)]
```

If the response to ADD OR DELETE is DELETE, the following prompt is displayed:

```
MODIFY COMMAND DEFINITION TABLE
  CHARACTER TO DELETE (HEX):  integer
```

Purpose:

The MCDT command adds or deletes a Command Definition Entry (CDE) from the specified Command Definition Table (CDT). The CDE describes a keyboard bid character and the task that it bids. You must use the Modify Device Configuration (MDC) command to enable a new CDE for a given terminal. Because the MDC command can be used only on a disk image, it is necessary to perform an IPL before the change to the CDE mask becomes effective.

Prompt Details:

VOLUME NAME:
The device name or disk name on which the system resides. The default is the current system disk.

SYSTEM NAME:
The name of the system image program file.

ADD OR DELETE:
The ADD response allows you to add new CDEs to the CDT. The DELETE response allows you to delete existing CDEs from the CDT.

MCDT (MODIFY COMMAND DEFINITION TABLE)

DEVICE TYPE/CDT NUMBER:

The device type or CDT number indicating the CDT that is to be modified. Each type of keyboard device is associated with a specific default CDT. You can enter a hexadecimal number (0 through 24) associated with a CDT or one of the following device types:

911	
940	
931	
733	
TPD	Teleprinter device
SD	Special device defined by the user
VT	Virtual Terminal

If you enter a device type as the response, the CDT associated with the device type is used. Otherwise, the CDT number specifies the table to modify.

FIVE WORD ENTRY:

A list of five hexadecimal words, separated by commas, which form the first part of the new CDE. (The response to DEFAULT USER ID composes the second part.) The CDE is composed of the following:

BYTE 0 The entry identification character. This character, when typed after the Attention key is pressed, bids the task associated with the entry identification character. Refer to the keyboard character tables in Appendix A for appropriate character codes.

BYTE 1 The bid flags. Set each bit (bid flag) according to the following specifications:

If the first bit is set to zero, the task is bid in a new job. If the first bit is set to one, the task is bid in the existing job. If a job does not exist, a new job is created.

If the second bit is set to zero the station number is passed to the log-on task. The station number is passed in the first byte of the second word of the bid parameters. If the second bit is set to one, the entry identification character is passed to the log-on task. The character is passed in the first byte of the first word of the bid parameters. If the job already exists, the second bit is ignored.

When the third bit is set to one an even loading bid is performed; that is, the task is bid in an existing job. The user ID of the job can be prompted or specified in the CDE. All jobs running under the specified user ID are searched; the task is bid in the job containing the fewest tasks.

MCDT (MODIFY COMMAND DEFINITION TABLE)

When the fourth bit is set to one the task is bid in a job with the user ID associated with the specified CDE. If this bit is set to zero, the bidding terminal is prompted for a user ID.

The remaining four bits of BYTE 1 are reserved and must be set to zero.

BYTE 2 The global LUNO used to bid the log-on task. If > FF is used, the standard supplied log-on task is bid from the utility program file. If any other LUNO is used, that global LUNO must be previously assigned to a program file containing the log-on processor written by the user.

BYTE 3 The installed ID of the log-on task that is bid. The log-on task resides in the program file associated with the LUNO assigned in BYTE 2.

BYTE 4 The LUNO assigned to the program file containing the task to be bid by the CDE. If the value is > FF, the utility program file is used.

BYTE 5 The installed ID of the task that is bid by the CDE. The task resides in the program file associated with the LUNO assigned in BYTE 4.

BYTES 6,7 The first parameter to be passed to the task that is bid.

BYTES 8,9 The second parameter to be passed to the task that is bid.

DEFAULT USER ID:

The eight character user ID associated with the task that is bid by the CDE. The response to this prompt is used when the fourth bit of BYTE 1 is set to one. If the fourth bit of BYTE 1 is set to zero, the bidding terminal is prompted for a user ID. The default is eight blanks.

CHARACTER TO DELETE (HEX):

The hexadecimal equivalent of the keyboard bid character that is deleted from the specified CDT.

MCDT (MODIFY COMMAND DEFINITION TABLE)

Example:

In the following example, the MCDT command adds a CDE to VDT number > A. After the user presses the Attention key, followed by a capital letter U (hexadecimal character code > 55), the new CDE bids task > 99 on .S$UTIL.

```
[] MCDT

MODIFY COMMAND DEFINITION TABLE
               VOLUME NAME: MYDISK
               SYSTEM NAME: SYSTEM1
            ADD OR DELETE: ADD
   DEVICE TYPE/CDT NUMBER: >A

MODIFY COMMAND DEFINITION TABLE

FIVE WORD ENTRY:  >5500,>FF0A,>FF99,0,0

DEFAULT USER ID:  MARTY
```

The next example deletes the CDE associated with the character P (hexadecimal character code > 50) from CDT number > A (the default CDT for 911 devices).

```
[] MCDT

  MODIFY COMMAND DEFINITION TABLE
                 VOLUME NAME: MYDISK
                 SYSTEM NAME: SYS2
              ADD OR DELETE: DELETE
     DEVICE TYPE/CDT NUMBER: 911

  MODIFY COMMAND DEFINITION TABLE
  CHARACTER TO DELETE (HEX): >50
```

Notes:

If a new CDE is added for a keyboard device, all the CDE masks associated with keyboard devices using the new CDE may need to be changed. Use the MDC command to change the masks.

If the CDE entry for the hard break character is deleted, the code > 18 should not be used for anything else.

Related Commands:

MDC	(Modify Device Configuration)
SCDT	(Show Command Definition Table)
XSCU	(Execute System Configuration Utility)

MD (MAP DISK)

4.123 MD (MAP DISK)

Prompts:

```
MAP DISK
                    PATHNAME:  [site:][pathname]@                    (*)
          LISTING ACCESS NAME:  [site:][pathname]@
                  SHORT FORM?:  YES/NO                               (YES)
               TOP LEVEL ONLY?: YES/NO                               (NO)
        DIRECTORY NODES ONLY?:  YES/NO                               (NO)
          EXECUTION MODE(F,B):  {F/FOREGROUND/B/BACKGROUND}          (FOREGROUND)
```

Purpose:

The MD command lists information about the contents of a disk volume, a directory, or a file. MD processes directories that contain up to 3000 or more used entries. The amount of information listed depends on the option or options you specify with the command. Table 4-13 describes the information output by the MD command when the long form option is selected. The information limits for the other options are described in the prompt details for this command.

Abort the MD command at any time by pressing the hard break key sequence (press the Attention key, release it, and hold down the Control key while you press the X key) on your terminal.

Prompt Details:

PATHNAME:
The pathname that identifies the disk volume, device, top level directory, or file with the information to be listed. If you specify a volume or device name, the VCATALOG directory for the resource is mapped. If you enter a null response, the VCATALOG directory for the system disk is mapped.

LISTING ACCESS NAME:
The device or file name to which the information produced by the MD command should be listed. The default is the terminal local file.

SHORT FORM?:
If you enter YES in response to this prompt, only the short form information is listed (the first ten items listed in Table 4-13.) If NO, the long form information is listed (all items shown in Table 4-13.)

TOP LEVEL ONLY?:
If you enter YES in response to this prompt, only information about the directory specified for the PATHNAME prompt is listed. If NO, information about all files and subdirectory files under the directory is included.

DIRECTORY NODES ONLY?:
If you enter YES in response to this prompt, only information about directories (none about files) is to be listed as follows:

- The number of subdirectories and files that have been created under each directory.
- The number of spaces for files and subdirectories that are still available for creation under the directory.
- The total size occupied by the directory and its subdirectories and files.

EXECUTION MODE(F,B):
If you enter F or FOREGROUND, the command executes in foreground mode. If you enter B or BACKGROUND, the command executes in background mode. The default is FOREGROUND. However, in a batch stream, this command always executes in background mode.

Example:

In the first example, the MD command lists the information for the directory SYS1.GVB.MENU to the terminal local file. The short form option is specified. In the second example, the long form option is specified to list the information for the directory SYS1.GVB.LIST. The information in the second example is also listed to the terminal local file.

```
[] MD

   MAP DISK
                    PATHNAME: SYS1.GVB.MENU
        LISTING ACCESS NAME:
                SHORT FORM?: YES
            TOP LEVEL ONLY?: NO
      DIRECTORY NODES ONLY?: NO
        EXECUTION MODE(F,B): FOREGROUND
```

MD (MAP DISK)

```
     DISK MAP OF SYS1.GVB.MENU
       TODAY IS 17:05:21 WEDNESDAY, AUG 03, 1983.

                                        NUMBER CURRENT TOTAL
                                  FILE    OF     EOM   ALLOC
LV NAME                           TYPE  RECORDS  ADU    ADU       LAST UPDATE
-- -----------------              -     ------  -----  -----   ---------------
    MENU                           D    30      4       4       8/ 5/83  21:43:29
 2 MENU:                                            FILES=11  AVAILABLE=18
     APPAX                         S    245     3       3       2/18/83  10:39:30
     BATCH                         S    322     5       5       2/13/83   9: 0:28
     CMDHDR                        S    30      1       1       2/15/83   9: 7: 0
     CMDLST                        S    230     8       8       2/15/83   9: 7:23
     DUMMY                         S    26      1       1       2/13/83   9: 3:26
     DUMMY1                        S    560     9       9       2/13/83  16:48:20
     M$DEVICE                      S    25      1       1       2/13/83   8:59:45
     M$EDIT                        S    18      1       1       2/13/83   9: 0:32
     M$JOB                         S    11      1       1       2/13/83   8:59:31
     M$LC                          S    23      1       1       2/13/83   8:59:33
     PROC                          S    35      1       1       2/13/83   9: 0:16
**SYS1.GVB.MENU                                    TOTAL SIZE = 32 ADUS


   [] MD

                  PATHNAME: SYS1.GVB.LIST
       LISTING ACCESS NAME:
               SHORT FORM?: NO
           TOP LEVEL ONLY?: NO
     DIRECTORY NODES ONLY?: NO
       EXECUTION MODE(F,B): FOREGROUND


     DISK MAP OF SYS1.GVB.LIST
       TODAY IS 10:52:48 MONDAY, FEB 23, 1983.
```

MD (MAP DISK)

```
                                    NUMBER CURRENT TOTAL
                                FILE   OF     EOM   ALLOC
LV NAME                         TYPE  RECORDS  ADU    ADU        LAST UPDATE
-- ------------------           -     ------  -----  -----    ---------------
    LIST                         D    12       4      4       12/ 8/82   2:46: 2

   ALLOC: PRI=4     SEC=0       #SECS=0   LRECL=134  PRECL=134  BLK/ADU=3
  WPT/DPT/BLK/PERM/FORCED/=NNNYN  DATA-FMT=BINARY   CREATED 10/ 9/82  19: 7:33

 2 LIST:                                          FILES=8  AVAILABLE=3
    BCT                          S    8        1      1       11/ 2/82  17:16:10

   ALLOC: PRI=1     SEC=1       #SECS=0   LRECL=80   PRECL=864  BLK/ADU=1
  WPT/DPT/BLK/PERM/FORCED/=NNYYN  DATA-FMT=B SUPPRS CREATED 11/ 2/82  17:16: 9

    BET                          S    13       1      1        7/28/82   8:23:37

   ALLOC: PRI=1     SEC=1       #SECS=0   LRECL=80   PRECL=864  BLK/ADU=1
  WPT/DPT/BLK/PERM/FORCED/=NNYYN  DATA-FMT=B SUPPRS CREATED  7/28/82   8:23:36
                          2
    CLEAN                        S    7        1      1        8/22/82  14:19:53

   ALLOC: PRI=1     SEC=1       #SECS=0   LRECL=80   PRECL=864  BLK/ADU=1
  WPT/DPT/BLK/PERM/FORCED/=NNYYN  DATA-FMT=B SUPPRS CREATED  8/22/82  14:19:52

    COMPARE                      S    7        1      1        8/22/82  15:30:28

   ALLOC: PRI=1     SEC=1       #SECS=0   LRECL=80   PRECL=864  BLK/ADU=1
  WPT/DPT/BLK/PERM/FORCED/=NNYYN  DATA-FMT=B SUPPRS CREATED  8/22/82  15:30:27

    MASTER                       S    8        1      1        7/29/82  10:31:10


**SYS1.GVB.LIST                                  TOTAL SIZE = 13 ADUS
```

MD (MAP DISK)

Table 4-13. Long Form Output of MD Command

Field	Description
LV	Each directory and subdirectory on the disk volume, identified by the level number (LV) at which each was created.
NAME	Name of each directory or file held in the disk volume, directory, or file being mapped.
FILE TYPE	File types in directories and subdirectories are identified as follows:

File Type	Meaning
D	Directory
I	Image file
K	Key indexed file
P	Program file
R	Relative record file
S	Sequential file

NUMBER OF RECORDS	Current number of logical records in the file. For relative record files, that number is the number of the largest numbered record written on the file.
CURRENT EOM ADU	Number of Allocatable Disk Units (ADUs) needed to contain the current amount of data, up to the end of medium (EOM).
TOTAL ALLOC ADU	Total number of ADUs allocated to the directory or file.
LAST UPDATE	Date and time when the directory or file was last updated (month/day/year, hour:minute: second).
FILES =	Number of subdirectories and files that have been created under a directory.
AVAILABLE =	Number of spaces for subdirectories and files that are still available for creation under a directory.
TOTAL SIZE	Total space (in ADUs) currently allocated for all files under the directory named at the left.
PRI =	Primary allocation size of a directory or file in ADUs.
SEC =	Secondary allocation size of a directory or file in ADUs. Zero is selected if the file is not expandable.

Table 4-13. Long Form Output of MD Command (Continued)

Field	Description
#SECS =	Number of secondary allocations that have been made in the file.
LRECL	Logical record length (in bytes) assigned to the file when it was created.
PRECL	Physical record length (in bytes) assigned to the file when it was created.
ADU/BLK (or BLK/ ADU)	Relationship of physical records (blocks) to ADU size. ADU/BLK is the number of ADUs in one block or BLK/ADU is the number of blocks in one ADU.
WPT/DPT/BLK/ PERM/FORCED	Shows whether any of the following file conditions are set: WPT — Write-protected DPT — Delete-protected BLK — Blocked PERM — Permanent FORCED — Forced write A Y appears (in a string of Ys and Ns) if the flag for the condition is set. An N appears in the string if the flag for the condition is not set.
DATA-FMT	Format of the data stored on the file. B SUPPRS indicates that the data is stored on the file in blank-suppressed format. BINARY indicates that data on the file is stored in nonblank-suppressed (straight data) format.
CREATED	Date that the file was created (month/day/ year, hour:minute: second).

MDC (MODIFY DEVICE CONFIGURATION)

4.124 MDC (MODIFY DEVICE CONFIGURATION)

Prompts:

```
MODIFY DEVICE CONFIGURATION
        DATA DISK/VOLUME:  [site:][pathname]@
```

After you enter a response to the DATA DISK/VOLUME prompt or press the Return key if you do not want to enter a response, the following prompt is displayed:

```
MODIFY DEVICE CONFIGURATION
COMMAND(CHANGE,ADD,DELETE):  {C/A/D}        (C)
```

The prompts that appear next depend on the response to the COMMAND prompt.

If you enter D (Delete), you need to specify only the device name in response to the following prompt:

```
DELETE DEVICE
        DEVICE NAME:  devicename
```

If you enter C (Change), the following prompt is displayed on your terminal:

```
CHANGE DEVICE
        DEVICE NAME:  devicename
```

If you enter A (Add), the following prompt is displayed on your terminal:

```
ADD DEVICE
        DEVICE TYPE:  character(s)
```

If you enter C or A, some of the following prompts are displayed on your terminal depending on the type of device being added or changed:

```
CASSETTE OPENS VALIDATED?:  YES/NO          (YES)
          CHARACTER QUEUE:  integer         (*)
              CRU ADDRESS:  integer         (*)
DEFAULT RECORD SIZE:  integer         (*)
                   DRIVES:  integer         (*)
       EXPANSION CHASSIS:  [integer]       (*)
      EXPANSION POSITION:  [integer]       (*)
EXTENDED CHARACTER SET?:  YES/NO          (*)
               INTERRUPT:  integer         (*)
        OPENS VALIDATED?:  YES/NO          (*)
PRINT MODE (SERIAL / PARALLEL):  {S/P}     (*)
          TILINE ADDRESS:  integer         (*)
                TIME OUT:  integer         (*)
          TERMINAL TYPE:  integer          (*)
          INTERFACE TYPE:  character(s)    (*)
          SWITCHED LINE?:  YES/NO          (*)
```

MDC (MODIFY DEVICE CONFIGURATION)

BAUD RATE:	integer	(*)
ACU PRESENT?:	YES/NO	(*)
ACU CRU:	integer	(*)
FULL DUPLEX MODEM?:	YES/NO	(*)
ECHO?:	YES/NO	(*)
VDT TYPE:	integer	(*)
ASSOCIATED PRINTER?:	YES/NO	(*)
SPEED:	integer	(*)
CDT NUMBER:	integer	(*)
CDE MASK:	integer	(*)
CHANNEL NUMBER:	integer	(*)
NUMBER OF TERMINALS:	[integer]	(*)

Purpose:

The MDC command modifies device configurations specified during system generation. Execute this command only during an Execute System Configuration Utility (XSCU) session. If you add a device type that is not in your current configuration, apply the patches for the DNOS kernel to be sure the device service routine is up to date.

Prompt Details:

DATA DISK/VOLUME:
The volume or directory name containing the system object directories. This response is necessary only when adding or modifying new devices.

COMMAND(CHANGE,ADD,DELETE):
The response to this prompt can be C, A, or D, which specifies whether the configuration for the device is to be changed, added, or deleted, respectively.

DEVICE NAME:
The name of the device with a configuration to be changed, added, or deleted.

DEVICE TYPE:
The type of device to be added. Valid responses to the prompt are as follows:

Characters	Device
VDT	Video display terminal
911	Model 911 VDT
ASR	Model 733 ASR
KSR	Model 733/743 KSR
DISK	All supported disk models
MT	Model 979A Magnetic Tape Unit or Cartridge Tape
LP	All supported printer models
CR	Model 804 Card Reader
VT	Virtual terminal

MDC (MODIFY DEVICE CONFIGURATION)

CASSETTE OPENS VALIDATED?:
If you enter YES, the cassette can be accessed by many tasks according to the access privilege requested by the task. If NO, the cassette can be accessed by many different tasks at the same time. YES is the default value. This prompt is displayed only if the device is an ASR.

CHARACTER QUEUE:
An integer value which specifies the number of characters to be buffered when entered from a keyboard. The default value is the current value.

CRU ADDRESS:
The CRU address to be assigned to the device. The CRU addresses that can be assigned range from 0 to > 1F00.

DEFAULT RECORD SIZE:
An integer value specifying the physical record size of a file when it is created on a device. This prompt is only displayed for a disk device. The default value is 768.

DRIVES:
An integer value, from one through four, to be assigned to the device which specifies the number of drives of this device type.

EXPANSION CHASSIS:
The expansion chassis, in the range of one through seven, to be assigned to the device.

EXPANSION POSITION:
The position, in the range of 0 through 23, in the expansion chassis to be assigned to the device.

EXTENDED CHARACTER SET?:
Enter YES if an extended character set is to be assigned to the device or NO if an extended character set is not to be assigned to the device.

INTERRUPT:
The interrupt level, in the range of 3 through 15, to be assigned to the device.

OPENS VALIDATED?:
If you enter YES, a device can be accessed by one or many tasks according to the access privilege requested by the task. If NO, you can access the device by many tasks at the same time. The initial value is device dependent, as follows:

Device	Initial Value
Cassette	YES
Card Reader	YES
733 ASR	NO
Line Printer	YES
VDT	NO
KSR	NO

MDC (MODIFY DEVICE CONFIGURATION)

PRINT MODE (SERIAL / PARALLEL):

The character S (serial) or P (parallel) to be assigned to the device, specifying the printer type.

TILINE ADDRESS:

The address of the TILINE device as determined by the position of switches located on the controller. Consult the configuration chart attached to the top of the chassis to determine this address. The default values are > F800 for disk devices and > F880 for magnetic tape.

TIME OUT:

An integer value that indicates the number of seconds the system waits after an I/O operation is initiated to a device before assuming an error has occurred. The initial value is device dependent, as follows:

Device	Initial Value
Card Reader	30
733 ASR	0
Cassette	3
Line Printer	30
VDT	0
KSR	0

The time out for an associated line printer is the same as the 931 or 940 VDT with which it is associated. If the time out of the VDT is 0, the time out of the associated line printer is set to 60.

TERMINAL TYPE:

The TERMINAL TYPE is any hardcopy keyboard terminal.

INTERFACE TYPE:

The INTERFACE TYPE responses for TPDs are CI401 (Comm Interface) and TTY/EIA. For serial line printers the valid responses are 9902, TTY/EIA, CI403, and CI404. For 931s and 940s the valid responses are CI401, CI403, CI404, 9902, and 9903. The interface board determines which interface type you should select.

SWITCHED LINE?:

Terminals that are directly connected or are connected by leased lines are not switched lines. Switched lines are dial-up lines. A YES response indicates a switched line. The default is NO, specifying a directly connected or leased line.

BAUD RATE:

Baud rate is the measure of the speed of data transmission. The only accepted values are 110, 300, 600, 1200, 2400, 4800, and 9600.

ACU PRESENT?:

To establish a telephone connection, enter YES; otherwise, enter NO.

ACU CRU:

The address of the automatic call unit determined by its position in the chassis.

MDC (MODIFY DEVICE CONFIGURATION)

FULL DUPLEX MODEM?:
If you do not use a modem, accept the default YES. Respond NO only if using the Bell 202S modem (or its equivalent).

ECHO?:
A YES response enables character editing at the terminal; the terminal must be reconfigured without local copy. Otherwise, enter NO.

VDT TYPE:
Enter 911 to specify the model 911 VDT; enter 940 to specify the model 940 VDT; enter 931 to specify the model 931 VDT.

ASSOCIATED PRINTER?:
To specify that an optional printer is attached to either the 931 VDT or the 940 VDT, enter YES; otherwise, enter NO.

SPEED:
This parameter specifies the BAUD rate used for a particular serial line printer, 931 VDT, or 940 VDT. The only acceptable values for 940 VDTs and 931 VDTs are 110, 300, 600, 1200, 2400, 4800, 9600, and 19200. The default value for a 931 VDT or 940 VDT is 1200. For serial line printers the only acceptable values are 110, 200, 300, 600, 1200, 2400, 4800, and 9600. The default value for serial line printers is 4800. The maximum value allowed depends on the interface used. For a communication interface (CI401, CI403, or CI404), the maximum is 9600. For all 9200 interfaces, the maximum is 9600.

CDT NUMBER:
A hexadecimal number (0 through 24) indicating which Command Definition Table (CDT) is to be modified.

CDE MASK:
The Command Definition Entry (CDE) mask corresponds to 16 bits where each bit enables some or all of the entries of the specified table. This mask can be thought of as a set of 16 flags, one flag for each entry in the specified table. For example, if bit one is set to one, then entry one is used. If bit two is set to zero, then entry two is not used.

CHANNEL NUMBER:
If the device that you are adding uses either the CI403 or the CI404 interface, you must specify which channel (0-3) the device will use.

NUMBER OF VIRTUAL TERMINALS:
A decimal or hexadecimal integer in the range of 1 to 50 that specifies the number of virtual terminals. The default is the current number.

MDC (MODIFY DEVICE CONFIGURATION)

Example:

In the following example, the MDC command adds a line printer with an extended character set to the device configuration as follows:

```
[] MDC

MODIFY DEVICE CONFIGURATION
               DATA DISK/VOLUME: VOL1

MODIFY DEVICE CONFIGURATION
    COMMAND(CHANGE/ADD/DELETE): A

ADD DEVICE
                     DEVICE TYPE: LP

ADD LINE PRINTER
               OPENS VALIDATED?: YES
  PRINT MODE (SERIAL/PARALLEL): SERIAL
       EXTENDED CHARACTER SET?: YES
                     TIME OUT: 30

ADD LINE PRINTER - SERIAL
                 INTERFACE TYPE: CI404
                          SPEED: 4800

ADD LINE PRINTER - SERIAL
                 TILINE ADDRESS: >F860
                      INTERRUPT: 15
                 CHANNEL NUMBER: 2
```

Notes:

The MDC command cannot be used to modify the running system.

Related Commands:

XSCU (Execute System Configuration Utility)

MDS (MODIFY DEVICE STATE)

4.125 MDS (MODIFY DEVICE STATE)

Prompts:

```
MODIFY DEVICE STATE
                DEVICE NAME:   devicename
                DEVICE STATE:  [{ON/OFF/DIAGNOSTIC}]
```

Purpose:

The MDS command modifies the state of specific devices without regenerating the system. This facility enables you to place devices offline or online to the system. The ability to place devices offline enables you to anticipate future system requirements by generating system configurations for an expanded system and then setting these nonexistent devices offline until they are physically present. The ability to place devices offline while they undergo routine maintenance also protects the system from erroneous access attempts, which might be made to the device while it is disconnected.

Placing the device in the online state gives the system access to the device.

The MDS command also enables or disables a bit map read after write (BRW) capability. To enhance the reliability of disk allocation, a BRW helps ensure bit map correctness. To enable the bit map read after write, you must enter the MDS command in expert mode and enter a response to the BRW prompt as shown below:

```
[ ] MDS DN = DS02,BRW = YES
```

To disable the bit map read after write, execute the MDS command in expert mode and enter NO for the BRW prompt. If the bit map read is different from the bit map written, a system crash occurs. The default setting is to have bit map read after write enabled.

NOTE

The BRW prompt is not displayed when you execute the MDS command. A response to the BRW prompt can be entered only in interactive expert mode or in a batch stream.

Prompt Details:

DEVICE NAME:
The four-character name of the device.

DEVICE STATE:
If you enter OFF in response to this prompt, either the specified device must be placed offline or the device is not physically present in the current generated configuration. If you enter ON, the specified device must be placed online.

MDS (MODIFY DEVICE STATE)

If you specify DIAGNOSTIC, the device can then be used for running online diagnostic tasks. For details concerning online diagnostics, refer to the *DNOS Online Diagnostic and System Log Analysis Tasks User's Guide*.

A device that is in use cannot be placed offline or into a diagnostic state until it is no longer in use.

Example:

In the following example, a system is configured to contain two line printers, LP01 and LP02; however, only LP01 is physically present for system access. The second line printer, LP02, is now physically connected to the system and ready for use. Place LP02 online for system access by entering the following responses to the MDS command prompts:

```
[] MDS

MODIFY DEVICE STATE
          DEVICE NAME: LP02
         DEVICE STATE: ON
```

Assumptions:

The device to be placed online has been generated in the system configuration and is physically connected for system access.

Notes:

If the state of a device has been modified and an IPL is done, execute the MDS command again to place the device in the desired state. However, if the state of a device has been modified during an Execute System Configuration Utility (XSCU) session, the modification is permanent.

Two additional prompts are available only in expert mode, CODE8 and RAW. You can use the CODE8 prompt to set a parallel line printer to use 8-bit ASCII code. The RAW prompt should not be set by a user since it is intended only for troubleshooting.

MFN (MODIFY FILE NAME)

4.126 MFN (MODIFY FILE NAME)

Prompts:

```
MODIFY FILE NAME
              OLD PATHNAME:  [site:]filename@
              NEW PATHNAME:  [site:]filename@
                   REPLACE?: YES/NO                (NO)
```

Purpose:

The MFN command changes the name of a file and retains the aliases associated with the new file name. Aliases associated with the old file name are deleted. If the file is a program file, any channels associated with the old file are retained, and the channels associated with the new one, if it exists, are deleted. In response to command prompts of the MFN command, you must enter the pathname that currently identifies a file and the new pathname that is to identify the file.

Prompt Details:

OLD PATHNAME:
The pathname that currently identifies the file.

NEW PATHNAME:
The new pathname that is to identify the file.

REPLACE?:
If you specify YES in response to this prompt, the pathname specified for NEW PATHNAME will become the file pathname of the file specified for OLD PATHNAME. If the pathname specified for the NEW PATHNAME prompt is an existing file, that file is deleted when the MFN command executes. However, any aliases for that file are retained.

If you specify NO in response to this prompt, the pathname specified for NEW PATHNAME will become the file pathname of the file specified for OLD PATHNAME unless the NEW PATHNAME prompt identifies an existing file. If the pathname for NEW PATHNAME does identify an existing file, the modification of the filename is not performed and an error is displayed on your terminal.

Example:

In the following example, the MFN command changes the name of a file from VOL1.DIR2.FILE1 to VOL1.FILEA and destroys the existing contents of file VOL1.FILEA.

```
[] MFN

MODIFY FILE NAME
         OLD PATHNAME: VOL1.DIR2.FILE1
         NEW PATHNAME: VOL1.FILEA
              REPLACE?: YES
```

MFN (MODIFY FILE NAME)

Notes:

You cannot change the names of files to which a logical unit number (LUNO) is currently assigned.

The MFN command fails if an attempt is made to modify the name of a file which is in a full directory. Since MFN must first create the new file, the MFN command to a full directory fails with a message specifying that a file cannot be created.

When MFN is used in a file security environment, you must have delete access rights to the file specified for OLD PATHNAME. If you are replacing the file specified for NEW PATHNAME, you must also have delete access to that file. In a file security environment, you cannot modify the name of a directory unless you are member of the system manage access group, SYSMGR.

MFP (MODIFY FILE PROTECTION)

4.127 MFP (MODIFY FILE PROTECTION)

Prompts:

```
MODIFY FILE PROTECTION
                    PATHNAME:  [site:]filename@
      WRITE-DELETE PROTECT?:  YES/NO              (NO)
           DELETE PROTECT?:  YES/NO               (NO)
```

Purpose:

The MFP command assigns write protection or delete protection to a file or removes existing protection from a file. All files are created unprotected and remain so until you use the MFP command to assign the protection type.

Prompt Details:

PATHNAME:
The pathname or an alias that identifies the file that is having protection added or changed.

WRITE-DELETE PROTECT?:
If you enter YES as the response to this prompt, the file is write and delete protected. If NO, the file is not write or delete protected. Write-delete protect implies that a file can only be read.

DELETE PROTECT?:
If you enter YES in response to this prompt, the file is delete protected. If NO, the file is not delete protected. A delete-protected file can be modified but not deleted.

Example:

In the following example, the MFP command assigns write-delete protection to a file identified by the pathname VOL1.DIRA.FILE2:

```
[] MFP

MODIFY FILE PROTECTION
                  PATHNAME: VOL1.DIRA.FILE2
   WRITE-DELETE PROTECT?: YES
         DELETE PROTECT?: NO
```

Notes:

You cannot modify the protection of files to which a logical unit number (LUNO) is currently assigned. Directory entries as well as data files can be protected using this command. Changing the protection of a directory entry does not change the protection of the data files of the directory. Each data file must have the protection modified individually. A batch stream created from a List Directory (LD) is a convenient method of modifying the protection of all data files of a directory.

MFP (MODIFY FILE PROTECTION)

When you specify an alias as the pathname, the alias is not protected from deletion; only the file is protected.

In a file security environment, you must have both write access and delete access to a file in order to use the MFP command.

MHPC (MODIFY HARDCOPY TERMINAL PORT CHARACTERISTICS)

4.128 MHPC (MODIFY HARDCOPY TERMINAL PORT CHARACTERISTICS)

Prompts:

```
MODIFY HARDCOPY TERMINAL PORT CHARACTERISTICS
        TERMINAL ACCESS NAME:  stationname              (*)
          IN SERVICE (YES/NO)?: [YES/NO]
          READ ASCII TIMEOUT:   [integer]
               WRITE TIMEOUT:   [integer]
          READ DIRECT TIMEOUT:  [integer]
                       SPEED:   [{ – 1,> FFFF,110,300,600,1200,4800,9600}]
               TERMINAL TYPE:   [character(s)]
  DISCONNECT SEQUENCE CODE:     [{0,1,2}]
          DUPLEX/ECHO CODE:     [integer]
USE LINE TURNAROUND (YES/NO)?:  [YES/NO]
  LINE TURNAROUND CHARACTER:    [integer]
```

Purpose:

The MHPC command allows you to access the port characteristics modification calls supported by the device service routine (DSR) of a teleprinter device (TPD).

Prompt Details:

TERMINAL ACCESS NAME:
A valid system terminal name. The terminal must be logically turned on and not currently in use.

IN SERVICE (YES/NO)?:
A YES response enables the usage of a particular port. A NO response disables the usage of a particular port. If you do not supply a response, the current parameter value is unchanged.

READ ASCII TIMEOUT:
A decimal or hexadecimal integer specifying number of seconds for READ ASCII TIMEOUT. The number zero represents infinity. A timeout value is provided with the system generation and can be changed with this prompt.

WRITE TIMEOUT:
A decimal or hexadecimal integer specifying number of seconds for WRITE TIMEOUT. The number zero represents infinity. A timeout is provided with the system generation and can be changed with this prompt.

READ DIRECT TIMEOUT:
A decimal or hexadecimal integer specifying number of seconds for READ DIRECT TIMEOUT. The number zero represents infinity. A timeout is provided with the system generation and can be changed with this prompt.

MHPC (MODIFY HARDCOPY TERMINAL PORT CHARACTERISTICS)

SPEED:

The SPEED prompt is applicable only if either the CI401 or the CI402 is used as the interface. You must use one of the following integer values:

-1
110
300
600
1200
2400
4800
9600
>FFFF

The >FFFF value is only applicable if using the 212A modem and indicates that the modem is to operate at 300 or 1200 baud. If you do not supply a response, the current parameter value is unchanged.

TERMINAL TYPE:

This prompt indicates the type of terminal used. You must use one of the following integer values:

703
707
743
745
763
765
781
783
785
787
820
825
840

If you do not supply a response, the current parameter value is unchanged.

DISCONNECT SEQUENCE CODE:

DISCONNECT SEQUENCE CODE is optional. The following integer values are valid responses and indicate specific actions:

0 Do not recognize any disconnect sequences.

1 Disconnect upon receipt of EOT.

2 Disconnect upon receipt of DLE EOT.

As a result of the system generation, the supplied value of this parameter is one.

MHPC (MODIFY HARDCOPY TERMINAL PORT CHARACTERISTICS)

DUPLEX/ECHO CODE:
The DUPLEX/ECHO CODE prompt is optional. The following integer values are valid responses:

0 Full duplex/echo. Characters transmitted by the terminal are echoed unless disabled by the application reading the data.

1 Full duplex/no echo.

2 Half duplex. No echo is automatic.

USE LINE TURNAROUND (YES/NO)?:
This prompt is optional and only applicable if the circuit uses a half duplex modem. If the response is YES and the circuit is half duplex, you must supply a LINE TURNAROUND character. The system generation supplied value is NO.

LINE TURNAROUND CHARACTER:
A response to this prompt is necessary if the response to USE LINE TURNAROUND is YES. Valid prompt responses must be a decimal or hexadecimal number less than >20. The response is appended by the DSR as the last character of any write operation. The system generation supplied value is EOT(>04).

NOTE

Notice that the use of EOT as the line turnaround character implies that the disconnect sequence code must not equal one.

Example:

In the following example, the terminal type is changed for the TPD defined as ST21 during system generation.

```
[] MHPC

MODIFY HARDCOPY TERMINAL PORT CHARACTERISTICS
         TERMINAL ACCESS NAME: ST21
          IN SERVICE(YES/NO)?:
           READ ASCII TIMEOUT:
                WRITE TIMEOUT:
                        SPEED:
                TERMINAL TYPE: 783
    DISCONNECT SEQUENCE CODE:
             DUPLEX/ECHO CODE:
USE LINE TURNAROUND(YES/NO)?:
   LINE TURNAROUND CHARACTER:
```

MHPC (MODIFY HARDCOPY TERMINAL PORT CHARACTERISTICS)

Notes:

The MHPC command does not support all possible parameter modifications. Generally, modifications which are not supported are associated with the use of the DSR to drive nonstandard terminals or a usage mode which is not encouraged. If necessary, you can develop a program which issues appropriate SVC calls to perform modifications not supported by this command.

Modifications produced by the MHPC command are not permanent. When an initial program load occurs, all modifications are lost. You can add these modifications to the .S$ISBTCH batch stream to put them into effect with each initial program load.

Related Commands:

ANS	(Answer Incoming Call)
CALL	(Call Terminal)
DISC	(Terminal Disconnection)
LHPC	(List Hardcopy Terminal Port Characteristics)

MHR (MODIFY HORIZONTAL ROLL)

4.129 MHR (MODIFY HORIZONTAL ROLL)

After you activate the Text Editor, the MHR command causes the file to roll right (toward column one) or roll left (toward the rightmost column) from the current column position. You can specify from 1 through 240 columns as an increment or decrement to the current column position. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the MHR command and the following prompts:

```
MODIFY HORIZONTAL ROLL
 NUMBER OF COLUMNS TO ROLL:   integer                (*)
```

4.130 MIR (MODIFY INTERNAL REGISTERS)

Prompts:

```
MODIFY INTERNAL REGISTERS
                        RUN ID:   integer              (*)
```

Purpose:

The MIR command modifies the internal registers (program counter (PC), workspace pointer (WP), and status register (ST)) of a task. If the task being debugged is not a privileged task, only bits zero through six of the status register can be modified with this command. If the task is not unconditionally suspended, it is temporarily suspended while the registers are being updated.

The MIR command, which cannot be used in the batch mode, is used to modify the internal registers by entering the new values and pressing the Return key. Also, press the Command key to return SCI to the command mode. If the internal registers to be modified are for a system task, you must have a user ID with a privilege level of two or higher.

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

Example:

In the following example, the MIR command displays the contents of the WP, PC, and ST registers for a task with a run ID of > 24. The contents of the WP and ST registers are modified.

```
[] MIR

MODIFY INTERNAL REGISTERS
            RUN ID:  >24

                WP:  0040     >60
                PC:  0106
                ST:  E40F     >40F
```

Notes:

In VDT mode, pressing the Return key when the cursor is positioned at the last displayed internal register or pressing the Command key regardless of cursor position causes the debug panel to appear. The explanation of the Show Panel (SP) command provides an example of the debug panel.

MIR (MODIFY INTERNAL REGISTERS)

Related Commands:

MM	(Modify Memory)
MWR	(Modify Workspace Registers)
SP	(Show Panel)

4.131 MJP (MODIFY JOB PRIORITY)

Prompts:

MODIFY JOB PRIORITY

JOB ID:	integer	(*)
PRIORITY:	integer	

Purpose:

The MJP command allows the system operator to modify the priority of a job while it is executing.

Prompt Details:

JOB ID:
Enter the job ID that is returned when the Show Job Status (SJS) or Execute Job (XJ) command is executed. List job IDs using the List Jobs (LJ) command.

PRIORITY:
The new priority to be assigned to the job. Priorities range from 1 through 31, with 1 being the highest priority.

Example:

In the following example, the job priority is changed from a priority of 3 to a priority of 1:

```
[] MJP

MODIFY JOB PRIORITY
            JOB ID: >22
          PRIORITY: 1
```

Notes:

The MJP command also affects the priorities of the tasks running in the job.

Related Commands:

HJ	(Halt Job)
KJ	(Kill Job)
LJ	(List Jobs)
RJ	(Resume Job)
SJS	(Show Job Status for < user ID>)
XBJ	(Execute Batch SCI Job)
XJ	(Execute Job)

MKF (MAP KEY INDEXED FILE)

4.132 MKF (MAP KEY INDEXED FILE)

Prompts:

```
MAP KEY INDEXED FILE
                    PATHNAME:  [site:]pathname@
          LISTING ACCESS NAME:  [site:][pathname]@
```

Purpose:

The MKF command displays the characteristics of a key indexed file. Table 4-14 describes the information given under each heading of the key indexed file map.

Prompt Details:

PATHNAME:
The pathname that identifies the key indexed file to be mapped.

LISTING ACCESS NAME:
The device or file name to which the MKF command should list the mapping information of the file specified for the PATHNAME prompt. The default value is the terminal local file.

Example:

In the following example, the contents of a key indexed file, VOL1.DIR1.FILE1, are mapped and the listing is displayed on the user's terminal by executing the MKF command.

```
[] MKF

MAP KEY INDEX FILE
                PATHNAME: VOL1.DIR1.FILE1
     LISTING ACCESS NAME: ME

          FILE MAP OF VOL1.DIR1.FILE1
            TODAY IS 10:45:22 TUESDAY, MARCH 18, 1983.

    KEYS:
                START                                  DUPLICATES
      KEY      COLUMN      LENGTH      MODIFIABLE      ALLOWED
       1          1           5             N             N
       2          8           3             Y             Y
       3         25          15             Y             N
```

Related Commands:

CFKEY (Create KIF File)

MKF (MAP KEY INDEXED FILE)

Table 4-14. Description of Information Output by MKF Command

Heading	Description
KEY	A number from 1 through 14 that identifies each key in the file. Key number 1 is the primary key.
START COLUMN	The character position in a record at which a key field starts. The first character position in a record is 1.
LENGTH	The number of characters in the key.
MODIFIABLE?	NO indicates that the contents of a key field must be present and cannot change when the record containing the key is rewritten. YES indicates that the contents of a key field can change when the record containing the key is rewritten or need not be present when the record is written.
DUPLICATES ALLOWED	YES indicates that the value of the key can be duplicated in the same key field in other records in the file. NO indicates that the value of the key cannot be duplicated in the same field of any other records in the file.

MKL (MODIFY KIF LOGGING)

4.133 MKL (MODIFY KIF LOGGING)

Prompts:

```
MODIFY KIF LOGGING
  KEY INDEXED FILE PATHNAME:   [site:]filename@
 LOGGING(FULL,PARTIAL,NONE):   {F/FULL/P/PARTIAL/N/NONE}          (*)
```

Purpose:

The MKL command selects the logging type for key indexed files. The logging type controls the state of the forced write bit. Any operation that modifies the contents of the file (insert, rewrite, or delete) will perform significantly faster when the file is in deferred write mode with the forced write bit turned off. This faster performance results from a decreased number of reads and writes to the disk. Table 4-15 illustrates the number of disk accesses required during the initial creation of a single key file in which the B-tree structure only requires a root node and a leaf node. It is assumed that the keys are inserted in sequential order. In Table 4-15 the following abbreviations are used:

Abbreviation	Definition
Y	The record must be read from or written to disk.
N	The operation will not be performed.
M	The record need not be read from disk because it is already in memory. The record is written to disk only when the file is closed or the memory it occupies is required for other buffers or tasks.

Table 4-15. Disk Accesses During KIF Creation

Operation	Full	Partial	None
Read Data Record	M	M	M
Prelog Data Record	Y	N	N
Write Data Record	Y	Y	M
Read Root Node	M	M	M
Read Leaf Node	M	M	M
Prelog Leaf Node	Y	N	N
Write Leaf Node	Y	N *	M
Modify FDR	Y	Y	M

Note:

* This is Y for single key indexed files

MKL (MODIFY KIF LOGGING)

Problems arise when errors occur while the file is in deferred write mode. Because parts of the file may have been modified before the error is detected, the file is no longer totally correct and should not be used. Therefore, before the logging is set to NONE, make a copy of it so that if an error occurs or the system crashes while the file is in deferred write mode, the file can be restored to the condition it was in before changes were made to it. The changes can then be reapplied. Also, all programs that modify the file are required to terminate whenever any error or informative code is returned. The building of a KIF from a sequential file thus can be accomplished by an MKL command followed by a Copy Sequential File to KIF (CSK) command. If the CSK does not complete normally, the reason for the failure can be corrected, the incorrect KIF deleted, a new file created, an MKL performed on the new file, and another CSK executed.

Prompt Details:

KEY INDEXED FILE PATHNAME:
The pathname of the key indexed file to prepare for initial loading.

LOGGING(FULL,PARTIAL,NONE):
The following responses indicate logging modes for modifying key indexed files and the capabilities of each response.

FULL	Full logging mode switches the forced write bit on. Key indexed files are created as forced write files.
PARTIAL	Partial logging mode switches the file to deferred write for the insert operation. The delete and rewrite behave like full logging files.
NONE	The file is in deferred write mode for all operations. After the file is closed, it returns to full logging mode.

MKL (MODIFY KIF LOGGING)

Example:

In the following example, a key indexed file is created and the MKL command is used to prepare the file for the initial loading of records as shown below:

```
[] CFKEY

CREATE KEY INDEXED FILE
                    PATHNAME: VOL1.KIF
     LOGICAL RECORD LENGTH: 80
    PHYSICAL RECORD LENGTH: 864
        INITIAL ALLOCATION:
      SECONDARY ALLOCATION:
               MAXIMUM SIZE: 125

      KEY DESCRIPTION FOR KEY NUMBER 1
            START POSITION: 1
                KEY LENGTH: 5
               DUPLICATES?: NO
               MODIFIABLE?: NO
            ANY MORE KEYS?: YES

      KEY DESCRIPTION FOR KEY NUMBER 2
            START POSITION: 8
                KEY LENGTH: 4
               DUPLICATES?: YES
               MODIFIABLE?: NO
            ANY MORE KEYS?: NO

[] MKL

MODIFY KIF LOGGING
  KEY INDEXED FILE PATHNAME: VOL1.KIF
 LOGGING(FULL,PARTIAL,NONE): NONE
```

Notes:

This command updates a sensitive disk structure (the file descriptor record (FDR)). Under abnormal system operation (that is, hardware problems), this command can result in the specified key indexed file becoming unusable. Logging is automatically turned on when the file is closed.

Partial logging significantly increases the speed of key indexed file insert operations; however, the results can cause the file to need rebuilding. If a system failure occurs after the first insert to a file and before the last close, rebuild the file with the Copy Key Indexed File Randomly (CKR) command. CKR copies the file (which may be inconsistent) to a sequential file or an existing key indexed file. All data can be recovered by CKR except the last operation processing when the system failure occurred.

The default mode for background execution is NONE; the default mode for foreground execution is FULL.

4.134 ML (MOVE LINES)

After you activate the Text Editor, the ML command moves a designated set of lines from one position to another position within the current file being edited. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the ML command and the following prompts:

```
MOVE LINES
                    START LINE:  [BEGINNING/END/line number]
                      END LINE:  [BEGINNING/END/line number]
          INSERT AFTER LINE:     [BEGINNING/END/line number]
```

MLP (MODIFY LUNO PROTECTION)

4.135 MLP (MODIFY LUNO PROTECTION)

Prompts:

```
MODIFY LUNO PROTECTION
 PROTECT OR UNPROTECT? (P/U):  {P/U}
                        LUNO:  integer                    (*)
```

Purpose:

The MLP command protects or unprotects a global LUNO.

LUNO protection is not a security feature but your discretionary option. Any user with a privilege level of 2 or higher can unprotect a LUNO and then release it.

Prompt Details:

PROTECT OR UNPROTECT? (P/U):
Specify P to protect the LUNO or U to unprotect the LUNO.

LUNO:
Specify an integer in the range of >01 to >FE.

Example:

In the following example, the global LUNO >10, which is assigned to a data file, will be protected from accidental release using the MLP command as follows:

```
[] MLP

MODIFY LUNO PROTECTION
PROTECT OR UNPROTECT? (P/U): P
                      LUNO: >10
```

Related Commands:

AGL	(Assign Global LUNO)
RGL	(Release Global LUNO)
SIS	(Show I/O Status)

4.136 MM (MODIFY MEMORY)

Prompts:

MODIFY MEMORY

RUN ID:	integer	(*)
ADDRESS:	full exp	

Purpose:

The MM command modifies the memory image of a task, starting at the address specified. If the task is not unconditionally suspended, it is temporarily suspended while the MM command is executing. Roll-in/roll-out does not affect the modification process. To modify a word, you type over the display and press the Return key. Pressing the Return key when the cursor is positioned at the last displayed memory address causes the next consecutive addresses and values to be displayed. Pressing the Previous Field key causes the cursor to move to the previous value. Pressing the Command key reactivates SCI.

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

ADDRESS:
The integer value of the first memory address to be modified.

Example:

In the following example, the MM command displays memory addresses beginning at location > 10 for a task with the run ID of > 10. Location > 12 is modified.

```
[] MM

MODIFY MEMORY
            RUN ID: >10
           ADDRESS: >10

RUN ID = 10   STATE = 06   WP = 0006   PC = 0074   <PC> = 2FE0
ST = 01CF   M

0010:       >04C0
0012:       >0502         >0500
0014:       >0100
```

MM (MODIFY MEMORY)

Notes:

In the above example, the terminal is in VDT mode. In TTY mode, the task run ID, state, and internal registers are not displayed. When you press the Command key while in VDT mode, the debug panel is displayed before SCI is reactivated. The explanation of the Show Panel (SP) command provides an example of the debug panel.

Related Commands:

MIR	(Modify Internal Registers)
MSM	(Modify System Memory)
MWR	(Modify Workspace Registers)

MO (MODIFY OUTPUT AT DEVICE)

4.137 MO (MODIFY OUTPUT AT DEVICE)

Prompts:

```
MODIFY OUTPUT AT DEVICE
                    SPOOL ID:  alphanumeric
DEVICE OR CLASS NAME:          [site:][devicename/character(s)]@
                    FORM:      [character(s)]
                    PRIORITY:  [integer]
```

Purpose:

The MO command allows the system operator or the owner of the print request to change the output device, class, or priority of a file on the print queue.

In the event that a listing device fails while printing a file, you can use this command to continue printing the file at another device. First, use the Halt Output at Device (HO) command to halt the output at the current device, then the MO command to specify the new device. Execute the Resume Output at Device (RO) command to resume printing at the halted device.

Prompt Details:

SPOOL ID:
: The ID assigned by the spooler subsystem. Obtain the ID by using the Show Output Status (SOS) command.

DEVICE OR CLASS NAME:
: The new device or class name where the file is to be printed. A null response specifies that the device or class will not change.

FORM:
: A one- to eight-character, user-defined string specifying the form type on which the output is to be placed. A null response specifies that the form type will not change.

PRIORITY:
: The new priority value that is associated with the file to be printed. Valid responses to this prompt range from 1 through 31. A null response specifies that the priority will not change.

MO (MODIFY OUTPUT AT DEVICE)

Example:

In the following example, the file in the print for LP01 needs to have its output changed from listing device LP01 to LP02. The form and priority will not change. First the operator issues the HO command, then the MO command:

```
[] HO

HALT OUTPUT AT DEVICE
        DEVICE OR CLASS NAME: LP01
      IMMEDIATELY OR AT EOF?: IMMEDIATELY

[] MO

MODIFY OUTPUT AT DEVICE
                    SPOOL ID: S00007
        DEVICE OR CLASS NAME: LP02
                        FORM:
                    PRIORITY:
```

Notes:

If the specified file is currently printing and one of the following options is entered with the MO command:

- Device or class name
- Form name
- Priority numerically greater than or equal to the current priority value

the file quits printing and is placed into the print queue at the location for its new parameters.

Related Commands:

HO	(Halt Output at Device)
KO	(Kill Output at Device)
RO	(Resume Output at Device)
SOS	(Show Output Status)

4.138 MOE (MODIFY OVERLAY ENTRY)

Prompts:

```
MODIFY OVERLAY ENTRY
     PROGRAM FILE PATHNAME:   [site:]filename@          (*)
          MODULE NAME OR ID:  {alphanumeric/integer}
```

After you enter responses to the previous prompts, the following prompts are displayed on your terminal:

```
MODIFY OVERLAY ENTRY FOR ID < n>
                        NAME:  alphanumeric            (*)
                RELOCATABLE?:  YES/NO                  (*)
             DELETE PROTECT?:  YES/NO                  (*)
```

where:

< n> is the ID of the overlay to be modified. (This ID is for user information only and cannot be modified.)

Purpose:

The MOE command alters the data supplied when the overlay was installed.

Prompt Details:

PROGRAM FILE PATHNAME:
The file name of the program file on which the overlay is installed.

MODULE NAME OR ID:
The overlay name or ID of the overlay installed on the specified program file.

NAME:
The name of the overlay. The system automatically places the associated overlay name in the response field of this prompt.

RELOCATABLE?:
If you enter YES, the overlay is allowed to be loaded at an address other than its natural load address.

DELETE PROTECT?:
If you enter YES, the overlay is protected against accidental deletion.

MOE (MODIFY OVERLAY ENTRY)

Example:

In the following example, the IO command installs a relocatable overlay named OLAY1 on a program file named .S$SHARED. LUNO >0A has been assigned to the file where the object module resides, and task >83 identifies the associated task segment. The MOE command causes the overlay to be delete protected.

```
[] IO

INSTALL OVERLAY
        PROGRAM FILE OR LUNO: 0
                OVERLAY NAME: OLAY1
                  OVERLAY ID: 0
    OBJECT PATHNAME OR LUNO: >0A
                RELOCATABLE?: YES
             DELETE PROTECT?: NO
 ASSOCIATED SEGMENT NAME/ID: >83
    ASSOCIATED SEGMENT TYPE: TASK


[] MOE

MODIFY OVERLAY ENTRY
   PROGRAM FILE PATHNAME: .S$SHARED
       MODULE NAME OR ID: OLAY1

MODIFY OVERLAY ENTRY FOR ID  >5C
                         NAME: OLAY1
                  RELOCATABLE?: YES
               DELETE PROTECT?: YES
```

Special Cases:

When executing this command in batch mode, you can enter two word lists enclosed in parentheses for the flag prompts. The first word is the verification data and the second word is the modification data. If a verification error occurs, no modifications are performed.

The following is an example of issuing the MOE command in batch mode, rather than interactively as shown in the previous example.

```
MOE    PROG=.S$SHARED,MODULE=OLAY1,
       DELETE PROTECT=(NO,YES)
```

MOE (MODIFY OVERLAY ENTRY)

Since the module was not previously delete protected, the verification was successful, and the status of the overlay is changed to be delete protected.

Related Commands:

DO	(Delete Overlay)
IO	(Install Overlay)
MPE	(Modify Procedure Segment Entry)
MSE	(Modify Program Segment Entry)
MTE	(Modify Task Segment Entry)

MPC (MODIFY PASSCODE)

4.139 MPC (MODIFY PASSCODE)

Prompts:

```
MODIFY PASSCODE
        CURRENT PASSCODE:: [character(s)]
            NEW PASSCODE:: [character(s)]
```

Purpose:

The MPC command permits you to modify your log-on passcode. Current passcodes are verified before being replaced by new passcodes.

Prompt Details:

CURRENT PASSCODE:
The current log-on passcode. The default is eight blanks, which denotes that no passcode is specified.

NEW PASSCODE:
A one- to eight-character, user-defined string. All characters in the string must be printable characters. The default is eight blanks, which denotes that no pass-code is specified.

Example:

In the following example, the MPC command replaces a user's current passcode with a new passcode.

```
[] MPC

MODIFY PASSCODE
       CURRENT PASSCODE: KAC
           NEW PASSCODE: K21
```

MPE (MODIFY PROCEDURE SEGMENT ENTRY)

4.140 MPE (MODIFY PROCEDURE SEGMENT ENTRY)

Prompts:

```
MODIFY PROCEDURE SEGMENT ENTRY
    PROGRAM FILE PATHNAME:  [site:]filename@        (*)
        MODULE NAME OR ID:  {alphanumeric/integer}
```

After you enter responses to the PROGRAM FILE PATHNAME and MODULE NAME OR ID prompts, the following set of prompts is displayed on your terminal:

```
MODIFY PROCEDURE SEGMENT ENTRY FOR ID < n>
                      NAME:  alphanumeric       (*)
          MEMORY RESIDENT?:  YES/NO             (*)
            DELETE PROTECT?:  YES/NO            (*)
```

where:

< n> is the ID of the procedure to be modified. (This ID is for user information only and cannot be modified.)

See the Special Cases paragraph of this command description for details on entering responses to the YES/NO prompts in batch mode.

After you enter responses to the above prompts, the following prompts are displayed on your terminal:

```
990/12 FLAGS
          EXECUTE PROTECT?:  YES/NO             (*)
            WRITE PROTECT?:  YES/NO             (*)
WRITABLE CONTROL STORAGE?:  YES/NO              (*)
```

Purpose:

The MPE command modifies the data supplied when the procedure segment was installed.

Prompt Details:

PROGRAM FILE PATHNAME:
The file name of the program file where the procedure segment to be modified has been installed.

MODULE NAME OR ID:
The procedure name or ID of the procedure segment installed on the specified program file.

MPE (MODIFY PROCEDURE SEGMENT ENTRY)

NAME:
The name of the procedure segment. The system automatically places the associated procedure name in the response field of this prompt.

MEMORY RESIDENT?:
The procedure segment is loaded into memory during initial program load (IPL) and remains in memory even when terminated if you enter YES and the procedure segment is installed on the .S$SHARED program file or the applications program file specified at system generation.

DELETE PROTECT?:
If you enter YES, the procedure segment is protected against accidental deletion.

EXECUTE PROTECT?:
If you enter YES, execution of the procedure segment is prohibited. The protection is enforced only on a 990/12 computer.

WRITE PROTECT?:
If you enter YES, procedure data cannot be modified in memory. The protection is enforced only on a 990/12 computer.

WRITABLE CONTROL STORAGE?:
If you enter YES, the procedure uses the writable control storage area. Writable control storage is available only with supplied software on a 990/12 computer.

MPE (MODIFY PROCEDURE SEGMENT ENTRY)

Example:

In the following example, the IP command installs a procedure named MYPROC on the user's program file. The file .MYPROG is a program file, and LUNO > BB has been assigned to the file where the object module resides. The MPE command will change the procedure name and specify that the procedure is delete protected.

```
[] IP

INSTALL PROCEDURE SEGMENT
       PROGRAM FILE OR LUNO: .MYPROG
             PROCEDURE NAME: MYPROC
               PROCEDURE ID: 0
   OBJECT PATHNAME OR LUNO: >BB
            MEMORY RESIDENT?: NO
             DELETE PROTECT?: NO

990/12 FLAGS
           EXECUTE PROTECT?: NO
             WRITE PROTECT?: NO
 WRITABLE CONTROL STORAGE?: NO

[] MPE

MODIFY PROCEDURE SEGMENT ENTRY
   PROGRAM FILE PATHNAME: .MYPROG
       MODULE NAME OR ID: MYPROC

MODIFY PROCEDURE SEGMENT ENTRY FOR ID  >4A
                          NAME: URPROC
              MEMORY RESIDENT?: NO
               DELETE PROTECT?: YES

990/12 FLAGS
              EXECUTE PROTECT?: NO
                WRITE PROTECT?: NO
    WRITABLE CONTROL STORAGE?: NO
```

Special Cases:

When executing this command in batch mode, you can enter two word lists enclosed in parentheses for the flag prompts. The first word is the verification data, and the second word is the modification data. If a verification error occurs, no modifications take place.

The following is an example of issuing the MPE command in batch mode rather than interactively as shown in the previous example:

```
MPE    PROG=.MYPROG,MODULE=MYPROC
       DELETE PROTECT=(YES,NO)
```

MPE (MODIFY PROCEDURE SEGMENT ENTRY)

If the module was previously delete protected, the status of the module is changed to not delete protected.

Related Commands:

DP	(Delete Procedure Segment)
IP	(Install Procedure Segment)
MOE	(Modify Overlay Entry)
MSE	(Modify Program Segment Entry)
MTE	(Modify Task Segment Entry)

4.141 MPF (MAP PROGRAM FILE)

Prompts:

```
MAP PROGRAM FILE
                    PATHNAME:  [site:]filename@          (*)
          LISTING ACCESS NAME:  [site:][pathname]@
```

Purpose:

The MPF command displays information about the task, procedures, program segments, and overlays contained within a program file. The flag definitions of the mapped program file are listed at the end of the output produced by the MPF command. Fields not defined on the output are described in Table 4-16.

Prompt Details:

PATHNAME:
The pathname of the program file to be mapped.

LISTING ACCESS NAME:
The device or file name to which the MPF command should list the mapping information of the file specified in the PATHNAME prompt. The default value is the terminal local file.

MPF (MAP PROGRAM FILE)

Example:

In the following example, the program file, SYS2.KC0017.PROG, is mapped and the contents are listed to the terminal local file.

```
[] MPF

MAP PROGRAM FILE
                PATHNAME: SYS2.KC0017.PROG
     LISTING ACCESS NAME:


 FILE MAP OF SYS2.KC0017.PROG
   TODAY IS  14:22:15 THURSDAY, DECEMBER 18, 1982.

TASK SEGMENTS:  MAX POSSIBLE = 255
 ID  NAME    LENGTH LOAD PRI S P M R D U RU CP E O C SP B OVLY P1/S P2/S   DATE
 01 DXLNKERR 08B6  0000  4         R                                     11/20/82
 02 TASKSAM  31C6  0000  4         R                                     10/14/82
 04 PCDUMP   2E4E  8000  4                              SP        01/Y    10/14/82
 06 PICTURE  3912  3620  4         R                               03/Y  10/14/82
 07 HASH     0810  0000  4         R                                     11/20/82

PROCEDURE/PROGRAM SEGMENTS: MAX POSSIBLE = 255
 ID  NAME    LENGTH LOAD S M R D U SH RU CP E W C                        DATE
 01 THECOUNT 8000  0000   M        SH                                   10/14/82
 02 MORCOUNT 8000  0000   M        SH                                   10/14/82
 03 PICTURE  3614  0000            SH                                   10/14/82
 60 CNTSEG   2000  0000            SH                                   11/18/82
 61 INSEG1   2000  0000            SH                                   11/18/82
 62 INSEG2   2000  0000            SH                                   11/18/82
 63 OUTSEG   2000  0000            SH                                   11/18/82

OVERLAYS:  MAX POSSIBLE = 255
 ID  NAME    LENGTH LOAD MAP  D  OVLY                                    DATE

FLAG DEFINITIONS
PRI - PRIORITY                        D   - DELETE PROTECTED
S   - SYSTEM                          U   - UPDATABLE
P   - PRIVILEGED                      O   - OVERFLOW
M   - MEMORY RESIDENT                 C   - WRITABLE CONTROL STORE
R   - REPLICATABLE                    W   - WRITE PROTECTED
RU  - REUSABLE                        SH  - SHARABLE
CP  - COPYABLE                        MAP - RELOCATION BIT MAP PRESENT
E   - EXECUTE PROTECTED               OVLY- OVERLAY LINK
SP  - SOFTWARE PRIVILEGED             B   - BYPASS SECURITY
P1/S - PROCEDURE 1 IS ON SAME PROGRAM FILE AS TASK
P2/S - PROCEDURE 2 IS ON SAME PROGRAM FILE AS TASK
```

MPF (MAP PROGRAM FILE)

Notes:

When the MPF listing shows a task with attached procedure 01/N, the attached procedure is in .S$SHARED.

Related Commands:

CFPRO (Create Program File)

Table 4-16. Program File Information Output by MPF Command

Heading	Description
ID	Identification number assigned to the task, procedure, program segment, or overlay when it was installed. IDs for real-time tasks are preceded by stars.
NAME	Name of the task, procedure, program segment, or overlay.
LENGTH	Amount of memory (bytes) that the task, procedure, program segment, or overlay would occupy in memory were it loaded in memory.
LOAD	Natural load address of the task, procedure, program segment, or overlay (the ORIGIN shown on a link edit map).
DATE	The date that the task, procedure, segment, program segment, or overlay was installed.

MPI (MODIFY PROGRAM IMAGE)

4.142 MPI (MODIFY PROGRAM IMAGE)

Prompts:

```
MODIFY PROGRAM IMAGE
                  PROGRAM FILE:  [site:]filename@          (*)
            OUTPUT ACCESS NAME:  [site:][pathname]@        (*)
                   MODULE TYPE:  {T/P/O/S}                 (*)
            MODULE NAME OR ID:   {alphanumeric/integer}    (*)
                       ADDRESS:  integer                   (*)
             VERIFICATION DATA:  [integer exp list]
                          DATA:  integer exp list
                     CHECKSUM:   integer
          RELOCATION OF DATA?:   [YES/NO . . . YES/NO]
```

Purpose:

The MPI command modifies a program (defined to be a task, procedure, program segment, or overlay) in a specified program file.

Prompt Details:

PROGRAM FILE:
The file name of the program file on which the program (task, procedure, program segment, or overlay) to be modified has been installed.

OUTPUT ACCESS NAME:
The device name or file name where the results of the memory image modification of the program are to be written. If you specify a null response, the terminal local file is used.

MODULE TYPE:
The type of program with a memory image to be modified. The following characters are valid responses:

T	=	Task
P	=	Procedure
O	=	Overlay
S	=	Program Segment

MODULE NAME OR ID:
The character(s) or the associated ID which identifies the program on the specified program file.

ADDRESS:
The integer value which is the starting address of the memory image to be modified.

MPI (MODIFY PROGRAM IMAGE)

VERIFICATION DATA:
If specified, the integer value contained in the specified starting address. If you specify more than one integer, you must separate them by commas; it is assumed these values are contained in consecutive memory addresses, beginning with the specified starting address.

DATA:
The integer value to replace the existing value contained in the specified starting address. If you specify more than one value, you must separate them by commas; it is assumed these values are to replace the existing values contained in consecutive memory addresses, beginning with the specified starting address.

CHECKSUM:
The checksum is an exclusive OR of each word of new data. If you do not know the checksum, it can be obtained by entering a null response. The checksum is printed to the device or file specified in response to the OUTPUT ACCESS NAME prompt.

RELOCATION OF DATA?:
YES specifies that the data value is relocated when the task is loaded into memory for execution. NO specifies that the data value should not be relocated. If a list of data values is specified in response to the DATA prompt and relocation is desired, you must specify which values are to be relocated. That is, you must enter a YES or NO response for each corresponding data value. If there is a list of YES or NO responses, you must separate them by commas. If no response is supplied, the previous value is used.

Example:

In the following example, the MPI command modifies one word of memory image for a procedure installed on a user's program file. The results of the modification are written to the terminal local file where the MPI command executed.

```
[] MPI

MODIFY PROGRAM IMAGE
            PROGRAM FILE: DS02.PROGRAM
      OUTPUT ACCESS NAME:
             MODULE TYPE: PROCEDURE
       MODULE NAME OR ID: PRINT
                 ADDRESS: 02000
       VERIFICATION DATA: 01EFE
                    DATA: 01FFF
                CHECKSUM:
    RELOCATION OF DATA?:
```

Related Commands:

SPI (Show Program Image)

MR (MODIFY ROLL)

4.143 MR (MODIFY ROLL)

After you activate the Text Editor, the MR command specifies the number of lines the file being edited will roll up (toward the end of the file) or roll down (toward the beginning of the file) from the current line when you activate a roll function (F1 key or F2 key). Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the MR command and the following prompt:

```
MODIFY ROLL
    NUMBER OF LINES TO ROLL:  integer                    (*)
```

4.144 MRF (MODIFY RELATIVE TO FILE)

Prompts:

```
MODIFY RELATIVE TO FILE
                  PATHNAME:  [site:]filename@          (*)
       OUTPUT ACCESS NAME:  [site:][pathname]@         (*)
             RECORD NUMBER:  integer                   (*)
                FIRST WORD:  integer                   (*)
         VERIFICATION DATA:  [integer exp list]
                      DATA:  integer exp list
                  CHECKSUM:  [integer]
```

Purpose:

The MRF command modifies the value of any word or group of words within a file. It is assumed that you have sufficient knowledge of the file structure not to destroy the file. The MRF command will optionally verify that the data you are modifying has the value(s) you expect and/or that the data you are entering has the correct checksum. It is recommended that you use the verification features whenever possible.

Prompt Details:

PATHNAME:
The file name with the data to be modified.

OUTPUT ACCESS NAME:
The device name or file name where the results of the MRF command are to be listed. If you specify a null response, the terminal local file is used.

RECORD NUMBER:
The integer value which is the physical record number within the file of the first word of data to be modified.

FIRST WORD:
The integer value which is the byte offset within the physical record of the first word of data to be modified. The byte offset must be an even number.

VERIFICATION DATA:
If specified, the integer value contained in the specified first word address. If you specify more than one integer, you must separate them by commas; it is assumed these values are contained in successive word addresses, beginning with the specified first word address.

DATA:
The integer value to replace the existing value contained in the specified first word address. If you specify more than one value, you must separate them by commas; it is assumed these values are to replace the existing values contained in successive word addresses, beginning with the specified first word address.

MRF (MODIFY RELATIVE TO FILE)

CHECKSUM:
The checksum is an exclusive OR of each word of new data. If you do not know the checksum, you can enter a null response and the checksum is printed to the device or file specified in response to the OUTPUT ACCESS NAME prompt.

Notes:

If the file can be thought of as a contiguous area of memory, it is possible to modify data from the file by address relative to the beginning of the data. To do this, specify zero for the RECORD NUMBER and enter the address of the desired data for the FIRST WORD. For example, to modify the word which is 5000 bytes into the file, enter the following:

```
        PATHNAME: .YOURFILE
RECORD NUMBER: 0
      FIRST WORD: 5000
```

This technique will not work for words more than 65535 bytes from the beginning of the file. To modify those words, it is necessary to determine the physical record length for the file. Then enter a RECORD NUMBER so that the expression:

Relative Address - (RECORD NUMBER * Physical Record Length)

has a value less than 65536. For example, to modify the word at address 66666 of a file which has a physical record length of 256, enter the following:

```
        PATHNAME: .YOURFILE
RECORD NUMBER: 20
      FIRST WORD: 66666-20*256
```

MRF (MODIFY RELATIVE TO FILE)

Example:

In the following example, the MRF command modifies the contents of one word for the first record of a user's file, beginning at word zero. The results are listed on the terminal local file.

```
[] MRF

MODIFY RELATIVE TO FILE
                PATHNAME: .KC0017.XB
     OUTPUT ACCESS NAME:
          RECORD NUMBER: 0
             FIRST WORD: 0
      VERIFICATION DATA: 0C000
                   DATA: 08000
              CHECKSUM: 08000

      VERIFICATION DATA
 0000  C000
       CURRENT DATA
 0000  C000
       NEW DATA : CHECKSUM = 0100
0000  8000 0018 000A 5842 2028 4558 4543 5554 .. .. .. XB  ( EX EC UT
```

Related Commands:

SRF (Show Relative to File)

MRM (MODIFY RIGHT MARGIN)

4.145 MRM (MODIFY RIGHT MARGIN)

After you activate the Text Editor, the MRM command establishes the column that is to be used as the right margin. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the MRM command and the following prompt:

```
MODIFY RIGHT MARGIN
        RIGHT MARGIN POSITION:   integer                (*)
```

4.146 MS (MODIFY SYNONYM)

Prompts:

MODIFY SYNONYM
SYNONYM: alphanumeric
VALUE: character(s)

Purpose:

The MS command accesses the synonyms and synonym values associated with a user ID. In VDT mode, SCI responds to the entry of an MS command by displaying the alphabetically first user-defined synonym not beginning with a $ character and its value as follows:

SYNONYM: A
VALUE: VOL1.DIRA.FILE1

In TTY mode, initially, only the synonym is printed or displayed.

When the first synonym and synonym value appear in response to entry of an MS command, you can enter a new synonym and new synonym value or modify the synonym value of an existing synonym. You cannot modify synonyms without entering a new synonym value. To modify a synonym and its value, use the keys described in Table 4-17. Additional features of the MS command include the following:

- You can display the alphabetically last synonym/value pair by clearing the synonym field and pressing the Previous Line key.
- You can display the alphabetically first $$ synonym/value pair by clearing the synonym field and pressing the Next Line key.
- If the synonym field is clear, you are not permitted to tab through the field (press the Return key). You *must* enter a response for this prompt (for example, entering of synonym characters or pressing the Previous Line or Next Line keys).
- If you clear the VALUE prompt field and press the Return key, the synonym and its value are deleted and the alphabetically next synonym/value pair is displayed.
- If the Next Line key is pressed when the alphabetically last synonym/value pair is displayed, a blank synonym/value pair is displayed. By pressing the Next Line key again, the first $$ synonym/value pair is displayed. When the last $ synonym/value pair is displayed and the Next Line key is pressed, the alphabetically first synonym/value pair is displayed.

By using the MS command, the user can selectively delete any unwanted synonyms rather than executing the Clear Synonyms (Q$SYN) command, which deletes many system synonyms beginning with the $ character.

MS (MODIFY SYNONYM)

Prompt Details:

SYNONYM:
A user-defined alphanumeric character string that is assigned the value entered in response to the VALUE prompt. This string can be a maximum of 50 characters in length.

VALUE:
The value assigned to the character string of the SYNONYM prompt field. The response to this prompt may be a maximum of 50 characters in length.

Example:

In the following example, the series of screen displays shows the modifications of synonyms and synonym values using the MS command. The actual editing steps that the user takes to change a synonym or synonym value are not shown.

1. The user enters the MS command, and the first available synonym and synonym value are displayed.

```
[] MS

MODIFY SYNONYM
                SYNONYM: AA
                  VALUE: VOL1.DIRA.FILEA
```

2. The user presses the Return key to move the cursor to the start of the value field. Then, the user enters the synonym value VOL1.DIRA.FILEAA and presses the Return key. The next synonym (in alphabetical order) and synonym value appear.

```
                SYNONYM: B
                  VALUE: VOL1.DIRA.FILEB
```

3. The user presses the Return key. The cursor moves to the start of the value field to allow modification of the value.

```
                SYNONYM: B
                  VALUE: VOL1.DIRA.FILEB
```

4. The user enters the synonym value VOL1.DIRB.FILE and presses the Return key. The next synonym (in alphabetical order) and synonym value are displayed.

```
                SYNONYM: C
                  VALUE: VOL1.DIRA.FILEC
```

5. The user presses the Previous Line key. The previous synonym (in alphabetical order) and its value are displayed.

```
              SYNONYM: B
                VALUE: VOL1.DIRB.FILEBB
```

MS (MODIFY SYNONYM)

6. The user presses the Previous Line key. The previous synonym (in alphabetical order) and its value are displayed.

```
SYNONYM: AA
  VALUE: VOL1.DIRA.FILEAA
```

7. The user presses the Next Line key. The next synonym (in alphabetical order) and its value are displayed.

```
SYNONYM: B
  VALUE: VOL1.DIRB.FILEBB
```

8. The user presses the Command key to exit from the MS command.

Notes:

The following synonyms are protected and cannot be deleted: $$12, $$CL, $$MO, $$SI, $$ST, $$UI, and ME.

Related Commands:

AS	(Assign Synonym)
LS	(List Synonyms)

MS (MODIFY SYNONYM)

Table 4-17. Generic Keys Used to Modify Synonyms and Values

Key	Use
Command	Exits from the MS command.
Next Line	Causes the display of the next available synonym (in alphabetical order).
Return	Signals that you have finished modifying a synonym or a synonym value. If you press the Return key without modifying a synonym or synonym value, the current value of the synonym or synonym value is accepted and an appropriate response is displayed. When the terminal is in VDT mode, ending the modification of a synonym (by pressing the Return key) causes the cursor to move to the synonym value field. When ending the modification of the synonym value, the next synonym and synonym value are displayed. If there are no more synonyms defined, blank synonym and value fields are displayed. In TTY mode, ending modification of a synonym causes the system to display the synonym value. Ending modification of the synonym value causes the next synonym, or, if there are no more, a blank synonym field to be displayed.
Forward Tab	Pressing the Shift and Forward Tab key has the same effect as pressing the Return key. Pressing only the Forward Tab key clears the field to the right of the cursor.
Previous Line	Causes the display of the previous synonym (in alphabetical order)
Skip	Deletes the synonym if pressed when the first character of the synonym value could be entered.
Erase Field	Clears the displayed field.

4.147 MSAR (MODIFY SECURITY ACCESS RIGHTS)

Prompts:

```
MODIFY SECURITY ACCESS RIGHTS
                    PASSCODE:  – character(s)
                   FILE NAME:  filename@
           ACCESS GROUP NAME:  alphanumeric
                READ ACCESS?:  YES/NO                (NO)
               WRITE ACCESS?:  YES/NO                (NO)
              DELETE ACCESS?:  YES/NO                (NO)
             EXECUTE ACCESS?:  YES/NO                (NO)
             CONTROL ACCESS?:  YES/NO                (NO)
```

Purpose:

The MSAR command permits you to add, delete, or modify access groups and their access rights for a specified file. To use this command, you must have the control access right for the specified file or you must be a member of the SYSMGR access group. You can use the List Security Access Rights (LSAR) command to see the current set of access rights to the file.

MSAR must be used once for each access group being modified for the file. More than one access right can be modified when the command is used. Specifying NO to all of the access rights deletes the access group from the list of groups associated with the file.

You can use MSAR to remove security from the file. By specifing PUBLIC for the ACCESS GROUP NAME prompt and YES for each of the access prompts, the file is unsecured.

By using MSAR and Create Batch (CB), you can modify the security access rights for all or part of a directory or file. The second example for MSAR shows such a case.

Prompt Details:

PASSCODE:
Enter your assigned log-on passcode.

FILE NAME:
Enter the file name of the file for which security access rights are to be modified.

ACCESS GROUP NAME:
Specify the name of the access group you wish to add, modify, or delete.

MSAR (MODIFY SECURITY ACCESS RIGHTS)

READ ACCESS?:

Specify YES to grant read access rights to the specified access group. Read access is more than just the right to read data from a file. It is the right to copy the contents of the file to another I/O resource. If the file is an SCI batch stream or procedure, read access is the right to execute the batch stream or the procedure. Read access to a program file includes the right to issue the Map Program File (MPF) command and the Show Program Image (SPI) command for any task in the file. A response of NO denies read access rights.

WRITE ACCESS?:

Specify YES to grant write access rights to the specified access group. Write access is the right to write data to a file. It includes the ability to write over the existing data as well as write new data. Write access to a progam file includes the right to install or delete tasks, segments, procedures, and overlays. Write access to a key indexed file includes the right to delete records from that file. A response of NO denies write access rights.

DELETE ACCESS?:

Specify YES to grant delete access rights to the specified access group. Delete access is the right to delete a file. A response of NO denies delete access rights.

One nonobvious requirement of delete access rights pertains to the use of the text editor features. For the Execute Text Editor (XE) command, you must have read access rights to the file specified for FILE ACCESS NAME. Then for the Quit Edit (QE) command, if the file specified for OUTPUT ACCESS NAME exists and is being replaced, you must have delete access rights.

EXECUTE ACCESS?:

If you enter YES, execute access rights are granted to the specified access group. Execute access has meaning only when it is associated with a program file. It represents the right to execute tasks, segments, procedures, and overlays within the file. A response of NO denies execute access rights.

CONTROL ACCESS?:

If you enter YES, control access rights are granted to the specified access group. Control access is the right to change the security associated with a file. This includes the right to change the set of access groups associated with the file, as well as change the access rights associated with each access group. Each secured file has one and only one access group with control access. Consequently, specifying YES transfers this right from the current control access group to the specified access group. A response of NO denies control access rights. A response of NO will result in an error if the file is currently unsecured or if the specified access group is the group which currently has the control access right.

MSAR (MODIFY SECURITY ACCESS RIGHTS)

Examples:

In the following example, the MSAR command is issued by the user with assigned passcode P89J5, who belongs to an access group which has control access rights to the file named VOL2.BATCH.SALES1. The MSAR command grants read and write access rights for the file VOL2.BATCH.SALES1 to the access group named CLERKS. The delete, execute, and control access rights are denied CLERKS for that particular file. (The passcode is not echoed to the user's terminal.)

```
[] MSAR

MODIFY SECURITY ACCESS RIGHTS
                    PASSCODE:
                   FILE NAME: VOL2.BATCH.SALES1
          ACCESS GROUP NAME: CLERKS
               READ ACCESS?: YES
              WRITE ACCESS?: YES
             DELETE ACCESS?: NO
            EXECUTE ACCESS?: NO
            CONTROL ACCESS?: NO
```

In the next example, a user has a directory named .PROJECT.DATA that he wants to make available for read access by access group CLERKS. His log-on passcode is FRIEND. The user builds this template file, named .PROJECT.TEMPLATE, for the Create Batch (CB) command to use in building a batch stream for securing all the files in the directory.

```
BATCH
* SET UP MSAR COMMAND FOR DATA DIRECTORY
!
MSAR    PASSCODE=FRIEND,
        FILENAME=.PROJECT.DATA.<?>,
        ACCESS GROUP NAME=CLERKS,
               READ ACCESS=YES,WRITE=NO,DELETE=NO,
               EXECUTE=NO,CONTROL=YES
!
EBATCH
```

Now the user issues the following CB command to create the batch stream:

```
[] CB

CREATE BATCH STREAM
       DIRECTORY TO PROCESS: .PROJECT.DATA
   BATCH TEMPLATE FILENAME: .PROJECT.TEMPLATE
     OUTPUT BATCH FILENAME: .PROJECT.BATCH
        SUBSTITUTE FOR <d>:
  PROCESS DIRECTORIES ONLY: NO
```

MSAR (MODIFY SECURITY ACCESS RIGHTS)

The batch stream in .PROJECT.BATCH will now have an MSAR command for every file in .PROJECT.DATA. If you want to, you can edit that batch stream to remove some of those commands. Once the batch stream is exactly as desired, you can use the Execute Batch (XB) command to execute .PROJECT.BATCH and secure all the files.

Related Commands:

LSAR	(List Security Access Rights)
SCAG	(Set Creation Access Group)

4.148 MSD (MODIFY SPOOLER DEVICE)

Prompts:

```
MODIFY SPOOLER DEVICE
                    DEVICE NAME:  [site:]devicename@                    (*)
       SPOOLER MODE(E,S,Q,D):  {E/EXCL/S/SHARED/Q/QUE/D/DEL}  (SHARED)
                          FORM:  [character(s)]                         (STANDARD)
       CHANGE CLASS NAME(S)?:  YES/NO                                   (NO)
```

If you enter YES in response to the CHANGE CLASS NAME(S)? prompt, the following prompt is displayed:

```
ENTER CLASS NAME CHANGE
                  CLASS NAME(S):  [character(s)...character(s)]         (*)
```

Purpose:

The MSD command modifies, adds, or deletes the attributes of a device used by the spooler subsystem.

Prompt Details:

DEVICE NAME:
The device name with attributes to be modified, added, or deleted. Only devices with names in the form of LPxy (where x and y are digits) are legal spooler devices.

SPOOLER MODE(E,S,Q,D):
If you enter E or EXCLUSIVE, the device is made totally available to the spooler subsystem. The device is in the SPLR device state and is not accessible to any other user of the system. If you enter S or SHARED, the spooler does not have exclusive access to the device, but can contend with other tasks in the system for use of the device. Requests for the device are queued for printing when the requests are made, and the files are printed when the device is not in use by other tasks. If you enter Q or QUEUE ONLY, requests for the device are queued for printing but are not printed until another MSD command is entered to set the mode to either EXCLUSIVE or SHARED. If you enter D or DELETE, the specified device is deleted from the spooler (if nothing is queued for printing), and print file requests made for this device are rejected.

FORM:
A one- to eight-character string that specifies the form type which is currently mounted on the device, if applicable. The spooler subsystem assumes that the indicated form is currently mounted on the device. The default is STANDARD.

CHANGE CLASS NAME(S)?:
A YES response, allows you to enter your class name definition on a subsequent prompt. If NO, a null string is passed to the spooler utility for class names, implying that the current names are not to be changed.

MSD (MODIFY SPOOLER DIVICE)

CLASS NAME(S):
A maximum of six eight-character strings which define the current class names of the device. You can add a class name to the current list or redefine the entire list. To remove the last class name, you need to delete the spool device and add it back again.

Example:

In the following example, the line printer LP02 has previously been used for printing accounting reports. To make the line printer available for use by several processors (including the spooler), it is specified as a SHARED device. The form type used for general printing is defined as the characters GENERAL. The MSD command modifies the line printer attributes as follows:

```
[] MSD

MODIFY SPOOLER DEVICE
               DEVICE NAME: LP02
    SPOOLER MODE(E,S,Q,D): SHARED
                     FORM: GENERAL
     CHANGE CLASS NAME(S): NO
```

Related Commands:

SOS (Show Output Status)

4.149 MSE (MODIFY PROGRAM SEGMENT ENTRY)

Prompts:

```
MODIFY PROGRAM SEGMENT ENTRY
    PROGRAM FILE PATHNAME:  [site:]filename@                (*)
        MODULE NAME OR ID:  {alphanumeric/integer}
```

After you enter responses to the PROGRAM FILE PATHNAME and MODULE NAME OR ID prompts, the following set of prompts is displayed on your terminal:

```
MODIFY PROGRAM SEGMENT ENTRY FOR ID < n>
                    NAME:  alphanumeric                 (*)
          SYSTEM SEGMENT?:  YES/NO                      (*)
        MEMORY RESIDENT?:  YES/NO                       (*)
          DELETE PROTECT?:  YES/NO                      (*)
              UPDATABLE?:  YES/NO                       (*)
               SHARABLE?:  YES/NO                       (*)
            REPLICATABLE?:  YES/NO                      (*)
    IN MEMORY REUSABLE?:  YES/NO                        (*)
    IN MEMORY COPYABLE?:  YES/NO                        (*)
```

where:

< n> is the ID of the segment to be modified. (This ID is for your information only and may not be modified.)

After you enter responses to the previous prompts, the following prompts are displayed on your terminal:

```
990/12 FLAGS
            EXECUTE PROTECT?:  YES/NO                   (*)
              WRITE PROTECT?:  YES/NO                   (*)
WRITABLE CONTROL STORAGE?:  YES/NO                      (*)
```

Purpose:

The MSE command modifies the attributes of a program segment installed on a program file.

Prompt Details:

PROGRAM FILE PATHNAME:
The file name of the program file on which the program segment to be modified has been installed.

MODULE NAME OR ID:
The segment name or ID of the program segment installed on the specified program file.

MSE (MODIFY PROGRAM SEGMENT ENTRY)

NAME:

The name of the program segment. The system automatically places the associated segment name in the response field of this prompt.

SYSTEM SEGMENT?:
If you enter YES, the program segment can be accessed only by a system task.

MEMORY RESIDENT?:
If you enter YES, and the program segment is installed on the .S$SHARED program file or the applications program file specified at system generation, the program segment will be loaded into memory during initial program load (IPL) and remain in memory even when terminated.

DELETE PROTECT?:
If you enter YES, the program segment is protected against accidental deletion.

UPDATABLE?:
If you enter YES, the data of a program segment can be modified and the program segment is written to disk with the new data modifications after the task is terminated or if the task maps the program segment out of its addressable memory area.

SHARABLE?:
If you enter YES, the program segment can be shared concurrently with more than one task.

REPLICATABLE?:
If you enter YES, multiple copies of the program segment can be in memory simultaneously.

IN MEMORY REUSABLE?:
If you enter YES, the program segment in memory can be reused after termination by another task rather than a new copy being copied from disk or being copied from one memory location to another.

IN MEMORY COPYABLE?:
If you enter YES, the program segment can be reused by copying the segment to more than one memory location rather than copying the segment from disk.

EXECUTE PROTECT?:
If you enter YES, execution of the program segment is prohibited. The protection is enforced only on a 990/12 computer.

WRITE PROTECT?:
If you enter YES, the program segment cannot be modified. The protection is enforced only on a 990/12 computer.

WRITABLE CONTROL STORAGE?:
If you enter YES, the program segment uses the writable control storage area. Writable control storage is available only with supplied software on a 990/12 computer.

MSE (MODIFY PROGRAM SEGMENT ENTRY)

Example:

In the following example, the Install Program Segment (IPS) command installs a program segment named PRINT with an ID of >3B on a user's program file named VOL1.USER. The object module for the program segment resides on a file named SYS1.KC0017.PROBJ. The program segment is installed with the following attributes: disk resident, delete protect, sharable, and replicatable.

The MSE command modifies the program segment attributes to include the following: updatable, in memory reusable, and in memory copyable.

```
[] IPS

INSTALL PROGRAM SEGMENT
            PROGRAM FILE OR LUNO: VOL1.USER
                     SEGMENT NAME: PRINT
                       SEGMENT ID: 03B
          OBJECT PATHNAME OR LUNO: SYS1.KC0017.PROBJ
           DEFAULT SEGMENT FLAGS?: NO

DEFINE SEGMENT FLAGS
                  SYSTEM SEGMENT?: NO
                 MEMORY RESIDENT?: NO
                  DELETE PROTECT?: YES
                        UPDATABLE?: NO
                         SHARABLE?: YES
                     REPLICATABLE?: YES
               IN MEMORY REUSABLE?: NO
               IN MEMORY COPYABLE?: NO

990/12 FLAGS
                  EXECUTE PROTECT?: NO
                    WRITE PROTECT?: NO
        WRITABLE STORAGE STORAGE?: NO

[] MSE

MODIFY PROGRAM SEGMENT ENTRY
             PROGRAM FILE PATHNAME: VOL1.USER
                 MODULE NAME OR ID: PRINT
```

MSE (MODIFY PROGRAM SEGMENT ENTRY)

```
MODIFY PROGRAM SEGMENT ENTRY FOR ID 03B
                        NAME: PRINT
             SYSTEM SEGMENT?: NO
            MEMORY RESIDENT?: NO
             DELETE PROTECT?: YES
                  UPDATABLE?: YES
                   SHARABLE?: YES
              REPLICATABLE?: YES
          IN MEMORY REUSABLE?: YES
          IN MEMORY COPYABLE?: YES

990/12 FLAGS
              EXECUTE PROTECT?: NO
                WRITE PROTECT?: NO
  WRITABLE CONTROL STORAGE?: NO
```

Special Cases:

When executing this command in batch mode, you can enter two word lists enclosed in parentheses for the flag prompts. The first word is the verification data, and the second word is the modification data. If a verification error occurs, no modifications take place.

Related Commands:

DPS	(Delete Program Segment)
IPS	(Install Program Segment)
MOE	(Modify Overlay Entry)
MPE	(Modify Procedure Segment Entry)
MTE	(Modify Task Segment Entry)

4.150 MSG (DISPLAY MESSAGE)

Prompts:

TEXT: character(s)
REPLY: [alphanumeric]

Purpose:

Use the MGG command to display a message at the terminal where the command was executed if SCI is the foreground task. Pressing the Return key causes the MSG command to complete and SCI then reads the next command following the MSG command.

If you enter a response for the REPLY prompt, SCI interprets the response as a synonym, waits for a user reply, and assigns the reply as the value of the synonym.

If in batch mode, the characters of the TEXT prompt are written to the batch listing file. The REPLY prompt is not valid in batch mode.

Prompt Details:

TEXT:
The user-defined character string that is to be displayed on the terminal where the MSG command is executed. The response to this prompt cannot exceed column 72.

REPLY:
An optional, user-defined, alphanumeric string that is interpreted by SCI as a synonym. The synonym value is the user response entered in reply to the message displayed on the terminal.

MSG (DISPLAY MESSAGE)

Example:

In the following example, assume a user backs up files from the system disk to a secondary disk every day, and the first component of the output file name is a synonym named DEV whose value is the device name where the secondary disk is installed. The availability of secondary disk drives varies from day to day; therefore, the same secondary disk drive usually is not used on consecutive days. The user can send a message to the terminal prior to backing up the files and reset the synonym DEV to correspond to the device name of the available secondary disk drive as follows:

```
[] MSG

          TEXT: NEW DEVICENAME
         REPLY: DEV

NEW DEVICENAME: DS02
```

In the next example, assume that within a command procedure a condition is detected that the user should be aware of. Within that procedure, the MSG command could be used to communicate with the user as follows:

```
*
*
*
MSG TEXT="WARNING: NON FATAL ERROR. CONTINUE (Y/N)?",
    REPLY=ANSWER
.IF @ANSWER,IS,ELEMENT(NO)
.EXIT
.ENDIF
*
*
*
```

Messages:

The characters entered in response to the TEXT prompt.

Related Commands:

CM (Create Message)

4.151 MSM (MODIFY SYSTEM MEMORY)

Prompts:

```
MODIFY SYSTEM MEMORY
        OVERLAY NAME OR ID:  {integer/alphanumeric}
                   ADDRESS:  integer
```

Purpose:

The MSM command modifies the memory occupied by the operating system. It is similar to the Modify Memory (MM) command, except that you can specify an overlay name or ID instead of a task run ID. Consecutive memory addresses can be displayed and/or modified by pressing the Return key. Press the Command key to return SCI to command mode.

Use the MSM command only if you are familiar with the operating system source.

Prompt Details:

OVERLAY NAME OR ID:
The overlay name or integer value that is the ID of the overlay installed on the kernel program file whose name is specified at system generation. By executing the Map Program File (MPF) command on the kernel program file, you can inspect the acceptable overlay names and associated IDs.

ADDRESS:
The integer value of the memory address at which to begin modifying memory.

Example:

In the following example, the MSM command displays the value of the specified memory address for the overlay with an ID of > 4. Consecutive memory addresses are modified, beginning with the specified address, by pressing the Return key after entering a new value for the address displayed. After the necessary memory addresses and their values have been modified and/or displayed, the Command key is pressed, returning SCI to command mode.

```
[] MSM


MODIFY SYSTEM MEMORY
              OVERLAY ID:  4
                 ADDRESS:  05900

                    5900:  >0004  6
                    5902:  >0000  1
                    5904:  >0429
```

MSM (MODIFY SYSTEM MEMORY)

Assumptions:

The user modifying the memory of the specified overlay is familiar with the operating system source.

Related Commands:

MIR	(Modify Internal Registers)
MM	(Modify Memory)
MWR	(Modify Workspace Registers)

4.152 MSP (MODIFY SCHEDULER/SWAP PARAMETERS)

Prompts:

```
MODIFY SCHEDULER/SWAP PARAMETERS

MODIFY MISCELLANEOUS VALUES
     FRONT PANEL DISPLAY-LEFT:  alphanumeric        (*)
    FRONT PANEL DISPLAY-RIGHT:  alphanumeric        (*)
          CLOCK TICKS/SLICE:    integer             (*)
     END ACTION LIMIT(STU'S):   integer             (*)
  MEMORY ERROR SAMPLE RATE:     integer             (*)
          MEDIUM JCA SIZE:      integer             (*)
```

After you enter responses for the preceding prompts, the following set of prompts are displayed on your terminal:

```
MODIFY SCHEDULER PARAMETERS
        INITIAL PRIORITY VALUES:  integer exp list      (*)
        WEIGHT OF JOB PRIORITY:   integer exp list      (*)
      DYNAMIC PRIORITY RANGE:     integer exp list      (*)
            AGING ON PRIORITY?:   YES/NO...YES/NO       (*)
      TICS BETWEEN SUSPENDS:      integer               (*)
```

After you enter responses for the preceding prompts, the following prompts are displayed on your terminal:

```
MODIFY SWAPPING PARAMETERS
            CACHABLE BUFFERS:     integer               (*)
  CACHABLE PROGRAM SEGMENTS:      integer               (*)
      MINIMUM SUSPENSION TIME:    integer               (*)
      MINIMUM EXECUTION TIME:     integer               (*)
  STATE > 24 IMMEDIATE ROLL?:     YES/NO                (*)
      LOADER TIME DELAY(STU'S):   integer               (*)
      JCA EXPANSION BOUNDARY:     integer               (*)
```

If the system being modified is not the running system, the following set of prompts is displayed:

```
MODIFY MORE MISCELLANEOUS VALUES
        FOREGROUND JOB LIMIT:  integer          (*)
            BATCH JOB LIMIT:   integer          (*)
      PHYSICAL MEMORY SIZE:    [integer]        (*)
                 SITE NAME:    alphanumeric     (*)
```

MSP (MODIFY SCHEDULER/SWAP PARAMETERS)

Purpose:

The MSP command allows system programmers to fine-tune scheduling and swapping algorithms in an attempt to improve throughput. Use this command only if you are familiar with the operating system.

Prompt Details:

FRONT PANEL DISPLAY-LEFT:

Allows specification of statistics to be displayed on the leftmost eight lights of the programmer panel. The statistics that can be specified are those listed when you execute the Execute Performance Display (XPD) command. Beginning from left to right, each light reflects 12 percent usage. Therefore, if no lights are on, the attribute is being used between 0 percent and 6 percent; if the leftmost light is on, the attribute is being used between 6 percent and 18 percent; if the two leftmost lights are on, between 18 percent and 30 percent, and so on. Refer to the listing of the XPD command for system attributes that can be monitored. These are listed in the left column of the XPD display.

FRONT PANEL DISPLAY-RIGHT:

The response to this prompt is the same as for FRONT PANEL DISPLAY (LEFT), except that the statistics are shown on the rightmost eight lights of the programmer panel. The use of the system attribute is reflected beginning with the rightmost light. For example, if there are no lights on, the system attribute is being used between 0 percent and 6 percent; if the rightmost light is on, the system attibute is being used between 6 percent and 18 percent; if the two rightmost lights are on, the attribute is being used between 18 percent and 30 percent, and so on.

CLOCK TICKS/SLICE:

The response to this prompt specifies the length of a time slice, in system time units, allotted to a task. One system time unit is equal to 50 milliseconds. If a task has not terminated by the end of the time slice, the task scheduler suspends the task if another task is ready.

END ACTION LIMIT(STU'S):

The response to this prompt specifies the amount of time, in system time units (STUs), that a task will be allowed to execute after it goes into end action before the system kills the task. If the task issues a Reset End Action SVC, this time limit is ignored.

MEMORY ERROR SAMPLE RATE:

The response to this prompt specifies the number of clock ticks (8.33 ms if 60 cycle power, 10 ms if 50 cycle power) that can elapse before the system tests for memory parity errors.

MEDIUM JCA SIZE:

An integer value, in bytes, specifying the medium size of the job communications area (JCA). This area is used by the system for overhead purposes during job execution. The medium JCA size can range from 2048 through 8000 bytes.

MSP (MODIFY SCHEDULER/SWAP PARAMETERS)

INITIAL PRIORITY VALUES:

The response to this prompt is a list of four integer values. The four integers specify the initial run-time priorities of foreground tasks which are installed with a priority of 1 through 4. The third element in the list also specifies the initial priority value for all background tasks. Initial priorities can range from 128 through 255.

WEIGHT OF JOB PRIORITY:

The response to this prompt is a list of four integer values. These integer values determine the affect job priority has on the priority of a task. Each integer value must be a power of two (0, 2, 4, 8, and so on). The maximum value that can be specified is 64. A job priority can either increase or decrease the task priority, depending on the values entered. The lower the value, the less effect a job priority has on task priority. However, the value specified can have a positive or negative affect on the task priority. For example, a value of 16 can have an affect of either + 16 or − 16 on the task run time priority. The four integers affect foreground tasks that are executed with a priority of 1 through 4. The third element in the list also specifies the weight of job priority for all background tasks.

DYNAMIC PRIORITY RANGE:

The response to this prompt is a list of four integer values. These integer values specify how much a task priority can vary if it is I/O bound. Each integer value must be a power of two (0, 2, 4, 8, and so on). The maximum value that can be specified is 128. The integer value can have a negative or positive affect on the task prioirity. For example, a value of 16 can have an affect of either + 16 or − 16 on the task run time priority, based on the I/O bound indicated. The more I/O bound results in a higher task priority. The four integers affect foreground tasks that are executed with a priority of 1 through 4. The third element in the list also specifies the dynamic priority range for all background tasks.

AGING ON PRIORITY?:

The response to this prompt is a list of four YES/NO characters. These YES/NO responses specify whether the task scheduler should increase a task priority after the task has executed for a specified time limit (system-specified time limit). The four YES/NO responses affect foreground tasks that are executed with a priority of 1 through 4. The third YES/NO response also affects all background tasks.

TICS BETWEEN SUSPENDS:

An integer value used by the system to determine if a task is I/O or CPU bound. This value should be an estimate of the average time a task will suspend. That is, if the value is 1, a task that suspends once per clock tick has its I/O bound indicator in the middle of its range. Valid responses are .5, .25, 1, 2, 4, and 8 (in clock ticks). If the line frequency is 60, there are 6 clock ticks per system time unit. If the line frequency is 50, there are 5 clock ticks per system time unit.

CACHABLE BUFFERS:

An integer value specifying the number of buffers or data segments that the system caches in memory. Cachable means the buffers are kept in memory unless the memory is needed.

MSP (MODIFY SCHEDULER/SWAP PARAMETERS)

CACHABLE PROGRAM SEGMENTS:
An integer value specifying the number of program segments the system caches in memory.

MINIMUM SUSPENSION TIME:
The response to this prompt specifies the minimum amount of time (in system time units) during which a task can be suspended before subject to roll.

MINIMUM EXECUTION TIME:
The response to this prompt specifies the minimum amount of time (system time units) that a task can execute after it is loaded before it is again subject to roll.

STATE > 24 IMMEDIATE ROLL?:
If you enter YES, Queue Server Tasks that are suspended waiting queue input (state > 24) are immediately eligible for roll when suspended. If NO, these tasks are not eligible for roll until they have been suspended for the time specified in response to the MINIMUM SUSPENSION TIME prompt. Enter YES for small memory systems; otherwise, enter NO.

LOADER TIME DELAY(STU"S):
An integer value specifying the time delay (in system time units) of the task loader when waiting for memory to become available.

JCA EXPANSION BOUNDARY:
An integer value that specifies the minimum number of bytes of free memory that the JCA can have available before the system automatically expands the JCA.

FOREGROUND JOB LIMIT:
The maximum number of foreground jobs that can be active in the system. The default value is 32.

BATCH JOB LIMIT:
The maximum number of background jobs that can be active in the system. The default value is 10.

PHYSICAL MEMORY SIZE:
A value in K bytes which represents the size of physical memory. The initial value is zero, allowing DNOS to calculate the size.

SITE NAME:
The name of the installation where the system is located. A valid character string must be entered.

MSP (MODIFY SCHEDULER/SWAP PARAMETERS)

Example:

In the following example, the MSP command specifies that disk statistics are to be monitored on the left half of the programmer's panel, and CPU statistics are to be monitored on the right half. The responses to all prompts in this example are the default values used for these parameters by the Execute System Generation Utility (XSGU).

```
[] MSP

MODIFY SCHEDULER/SWAP PARAMETERS

MODIFY MISCELLANEOUS VALUES
       FRONT PANEL DISPLAY (LEFT): DISK
      FRONT PANEL DISPLAY (RIGHT): CPU
                CLOCK TICKS/SLICE: 6
            END ACTION LIMIT(STU'S): 100
          MEMORY ERROR SAMPLE RATE: 12000
                   MEDIUM JCA SIZE: 2048

MODIFY SCHEDULER PARAMETERS
             INITIAL PRIORITY VALUES: 144,176,224,160
              WEIGHT OF JOB PRIORITY: 16,16,16,16
              DYNAMIC PRIORITY RANGE: 16,16,32,32
                  AGING ON PRIORITY?: NO,NO,YES,NO
               TICS BETWEEN SUSPENDS: 1

MODIFY SWAPPING PARAMETERS
                     CACHABLE BUFFERS: 60
           CACHABLE PROGRAM SEGMENTS: 30
             MINIMUM SUSPENSION TIME: 10
              MINIMUM EXECUTION TIME: 40
           STATE >24 IMMEDIATE ROLL?: YES
            LOADER TIME DELAY (STUS): 40
              JCA EXPANSION BOUNDARY: 400

MODIFY MORE MISCELLANEOUS VALUES
                 FOREGROUND JOB LIMIT: 32
                      BATCH JOB LIMIT: 10
                 PHYSICAL MEMORY SIZE: 0
                            SITE NAME: AUSTIN
```

Related Commands:

QSCU	(Quit System Configuration Utility)
XPD	(Execute Performance Display)
XSCU	(Execute System Configuration Utility)

MSP (MODIFY SCHEDULER/SWAP PARAMETERS)

Related Commands:

QSCU	(Quit System Configuration Utility)
XPD	(Execute Performance Display)
XSCU	(Execute System Configuration Utility)

4.153 MST (MODIFY SYSTEM TABLE SIZES)

Prompts:

```
MODIFY SYSTEM TABLE SIZES
                SYSTEM TABLE AREA:  {integer/ALL/MAX}   (*)
SEGMENT MANAGER TABLE AREA:         integer             (*)
    FILE MANAGER TABLE AREA:        integer             (*)
             BUFFER TABLE AREA:     integer             (*)
                SYSTEM JCA SIZE:    integer             (*)
```

Purpose:

The MST command lists table area sizes for the running system or modifies table area sizes specified during system configuration. MST can modify table area sizes when a Execute System Configuration Utility (SCU) session is active. The MST command can list but cannot modify table area sizes for the running system.

Prompt Details:

SYSTEM TABLE AREA:
An integer value, in bytes, specifying the maximum size of the system table area. The size of the system table area can range from 4096 bytes to the number of bytes available in the system root logical address space (usually about 16,000 bytes). Specifying ALL or MAX allocates the maximum number of bytes available.

SEGMENT MANAGER TABLE AREA:
An integer value, in bytes, specifying the maximum size of the system table area used for overhead by the segment manager when task segments are executed and during file I/O. The size of the segment manager table area can range from 2048 through 48,000 bytes.

FILE MANAGER TABLE AREA:
An integer value, in bytes, specifying the maximum size of the table area used for overhead by the file manager when it creates, deletes, or accesses files by a task or job. The file manager table size can range from 1024 through 262,144 bytes.

BUFFER TABLE AREA:
An integer value, in bytes, specifying the maximum size in memory that can be used as I/O buffers. This is the area used by the operating system to buffer I/O supervisor call blocks and data to be read or written to devices (not files). The I/O buffer area size can range from 2048 through the maximum defined during the system generation.

SYSTEM JCA SIZE:
An integer value, in bytes, specifying the size of the memory resident JCA table area used by the system job. This value can range from 2000 through 12,288 bytes.

MST (MODIFY SYSTEM TABLE SIZES)

Example:

In the following example, the MST command modifies the sizes of the segment manager table and file manager table. The default values for the system table area and the I/O buffer area are specified. When the MST command is executed, the prompts are displayed with the values specified for these areas when the system was generated or by a previous MST command. These are the default values.

```
[] MST

MODIFY SYSTEM TABLE SIZES
           SYSTEM TABLE AREA: 15000
 SEGMENT MANAGER TABLE AREA: 4096
    FILE MANAGER TABLE AREA: 8192
           BUFFER TABLE AREA: 2048
             SYSTEM JCA SIZE: 4096
```

Assumptions:

The user is aware of the memory available before modifying any table sizes.

Notes:

Use this command to modify tables only if you have executed the XSCU command.

Related Commands:

MDC	(Modify Device Configuration)
QSCU	(Quit System Configuration Utility)
XSCU	(Execute System Configuration Utility)

4.154 MT (MODIFY TABS)

After you activate the Text Editor, the MT command specifies the column where the cursor is to be positioned when the tab function is called. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the MT command and the following prompt:

```
MODIFY TABS
                TAB COLUMNS:   [integer exp list]          (*)
```

MTE (MODIFY TASK SEGMENT ENTRY)

4.155 MTE (MODIFY TASK SEGMENT ENTRY)

Prompts:

```
MODIFY TASK SEGMENT ENTRY
    PROGRAM FILE PATHNAME:  [site:]filename@                (*)
        MODULE NAME OR ID:  {Alphanumeric/Interger}
```

After you enter responses to the PROGRAM FILE PATHNAME and MODULE NAME OR ID prompts, the following set of prompts is displayed on your terminal:

```
MODIFY TASK SEGMENT ENTRY FOR ID < n>
                    NAME:  alphanumeric                 (*)
              REAL TIME?:  YES/NO                       (*)
                PRIORITY:  integer                      (*)
            MODIFY FLAGS?:  YES/NO                      (YES)
    ATTACHED PROCEDURES?:  YES/NO                       (NO)
```

where:

< n> is the ID of the task to be modified. (This ID is for user information only and cannot be modified.)

If you enter YES in response to the MODIFY FLAGS? prompt, the following prompts are displayed on your terminal:

```
MODIFY TASK FLAGS

                PRIVILEGED?:  YES/NO                    (*)
              SYSTEM TASK?:  YES/NO                     (*)
          MEMORY RESIDENT?:  YES/NO                     (*)
             REPLICATABLE?:  YES/NO                     (*)
           DELETE PROTECT?:  YES/NO                     (*)
      IN MEMORY COPYABLE?:  YES/NO                      (*)
      IN MEMORY REUSABLE?:  YES/NO                      (*)
               UPDATABLE?:  YES/NO                      (*)
    SOFTWARE PRIVILEGED?:  YES/NO                       (*)
```

After you enter responses to the previous prompts, the following prompts are displayed on your terminal:

```
990/12 FLAGS
          EXECUTE PROTECT?:  YES/NO                     (*)
        OVERFLOW CHECKING?:  YES/NO                     (*)
WRITABLE CONTROL STORAGE?:  YES/NO                      (*)
```

MTE (MODIFY TASK SEGMENT ENTRY)

If the response to the ATTACHED PROCEDURES? prompt was YES, the following set of prompts is displayed on your terminal:

MODIFY TASK-ATTACHED PROCEDURES

1ST PROCEDURE ID:	integer	(*)
P1 FROM TASKS PROGRAM FILE?:	YES/NO	(*)
2ND PROCEDURE ID:	integer	(*)
P2 FROM TASKS PROGRAM FILE?:	YES/NO	(*)

Purpose:

The MTE command alters the attributes supplied when the task segment was installed.

Prompt Details:

PROGRAM FILE PATHNAME:
The file name of the program file on which the task segment to be modified has been installed.

MODULE NAME OR ID:
The task name or ID of the task segment installed on the specified program file.

NAME:
The name of the task. The system automatically places the associated task name in the response field of this prompt.

REAL TIME?:
If you enter YES, the task segment to be modified is installed as a real-time task segment.

PRIORITY:
If you enter YES in response to the REAL TIME? prompt, the priority value specified must be in the range of 1 through 127 (inclusive). If you enter NO, the priority value specified must be in the range of 0 through 4 (inclusive).

MODIFY FLAGS?:
If you enter YES, you have the option of modifying the task flags.

PRIVILEGED?:
If you enter YES, the task is allowed to execute privileged hardware instructions.

SYSTEM TASK?:
If you enter YES, the task is allowed to execute in system memory space. For the task to be modified to become a system task, the task load address must be greater than or equal to > C000.

MEMORY RESIDENT?:
The task is loaded into memory during initial program load (IPL) and remains in memory when terminated if you enter YES and install the task on the .S$SHARED program file or the applications program file specified at system generation.

MTE (MODIFY TASK SEGMENT ENTRY)

REPLICATABLE?:
If you enter YES, multiple copies of the task can be in memory simultaneously.

DELETE PROTECT?:
If you enter YES, the task is protected against accidental deletion.

IN MEMORY COPYABLE?:
If you enter YES, the task segment can be copied from memory rather than being copied from disk. This situation can occur if the task is in memory and another user wants to execute the task.

IN MEMORY REUSABLE?:
If you enter YES, the program segment in memory can be reused after termination by another task rather than a new copy being copied from disk or being copied from one memory location to another.

UPDATABLE?:
If you enter YES, the data of a task can be modified, and the task segment is written to disk with the new data modifications when the task terminates.

SOFTWARE PRIVILEGED?:
If you enter YES, the task is allowed to execute privileged supervisor calls.

EXECUTE PROTECT?:
If you enter YES, execution of the task segment is prohibited. The protection is enforced only on a 990/12 computer.

OVERFLOW CHECKING?:
If you enter YES, the occurrence of arithmetic overflow causes control of the task to pass to the task end-action routine. Overflow checking is available only on a 990/12 computer.

WRITABLE CONTROL STORAGE?:
If you enter YES, the task uses the writable control storage area. Writable control storage is available only with supplied software on a 990/12 computer.

ATTACHED PROCEDURES?:
If you enter YES, you have the option to modify the procedures to be attached to the task segment.

1ST PROCEDURE ID:
The integer value representing the ID of a procedure segment attached to the task segment. If you enter zero, there are no procedure segments.

P1 FROM TASKS PROGRAM FILE?:
If you enter YES, the attached procedure segment with an ID specified for the 1ST PROCEDURE ID prompt resides on the same program file as the task segment. If you enter NO, the procedure segment must reside on the .S$SHARED program file.

MTE (MODIFY TASK SEGMENT ENTRY)

2ND PROCEDURE ID:
The integer value representing the ID of a procedure segment attached to the task segment. If you enter zero, there is no second procedure segment.

P2 FROM TASKS PROGRAM FILE?:
If you enter YES, the attached procedure segment with an ID specified for the 2ND PROCEDURE ID prompt resides on the same program file as the task segment. If you enter NO, the procedure segment must reside on the .S$SHARED program file.

Example:

In the following example, the Install Task (IT) command installs a task segment, having one attached procedure segment on the same program file. The MTE command is used to modify the procedure ID of the attached procedure segment. The default task flags are taken for both commands.

```
[] IT

INSTALL TASK SEGMENT
        PROGRAM FILE OR LUNO: .MY.PROGRAM
                   TASK NAME: MYFILE
                     TASK ID: 0
     OBJECT PATHNAME OR LUNO: >4A
                    PRIORITY: 3
         DEFAULT TASK FLAGS?: YES
        ATTACHED PROCEDURES?: YES

ATTACH TASK PROCEDURES
            1ST PROCEDURE ID: >42
P1 FROM TASKS PROGRAM FILE?: YES
            2ND PROCEDURE ID: 0
P2 FROM TASKS PROGRAM FILE?: NO

[] MTE

MODIFY TASK SEGMENT ENTRY
       PROGRAM FILE PATHNAME: .MY.PROGRAM
           MODULE NAME OR ID: MYFILE

MODIFY TASK ENTRY FOR ID  >83
                        NAME: MYFILE
                   REAL TIME?: NO
                     PRIORITY: 3
                MODIFY FLAGS?: NO
        ATTACHED PROCEDURES?: YES

MODIFY TASK-ATTACHED PROCEDURES
            1ST PROCEDURE ID: >45
P1 FROM TASKS PROGRAM FILE?: YES
            2ND PROCEDURE ID: 0
P2 FROM TASKS PROGRAM FILE?: NO
```

MTE (MODIFY TASK SEGMENT ENTRY)

Special Cases:

When executing this command in batch mode, you can enter two word lists enclosed in parentheses for the flag prompts. The first word is the verification data, and the second word is the modification data. If a verification error occurs, no modification takes place.

Notes:

If the task attribute is to be modified to make it a system task, the response to the ATTACHED PROCEDURES? prompt is ignored and the task load address must be greater than or equal to > C000.

Related Commands:

DT	(Delete Task)
IT	(Install Task)
MOE	(Modify Overlay Entry)
MPE	(Modify Procedure Segment Entry)
MSE	(Modify Program Segment Entry)

4.156 MTS (MODIFY TERMINAL STATUS)

Prompts:

```
MODIFY TERMINAL STATUS
                  TERMINAL NAME:  stationname@                (*)
            NEW STATUS (ON/OFF):  [{ON/OFF}]
            NEW MODE (TTY/VDT):   [{TTY/VDT}]
        DEFAULT MODE (TTY/VDT):   [{TTY/VDT}]
                ASK FOR USER ID?: YES/NO]
               ASK FOR JOB NAME?: [YES/NO]                  (YES)
            ASK FOR ACCOUNT ID?:  [YES/NO]                  (NO)
ASK FOR NAME MANAGER FILES?:      [YES/NO]                  (NO)
            RECONNECT ENABLED?:   [YES/NO]                  (NO)
```

If the response to ASK FOR USER ID? is NO, the following prompts are displayed:

```
SUPPLY DEFAULT USER ID
            DEFAULT USER ID:  alphanumeric
          DEFAULT PASSCODE:   [character(s)]
```

If the response to ASK FOR JOB NAME? is NO, the following prompt is displayed:

```
SUPPLY JOB NAME
            DEFAULT JOB NAME: [alphanumeric]                (*)
```

If the response to ASK FOR ACCOUNT ID? is NO, the following prompt is displayed:

```
SUPPLY ACCOUNT ID
            DEFAULT ACCOUNT ID: [character(s)]
```

Purpose:

The MTS command controls the following attributes of each terminal in your system:

- Terminals can be set ON or OFF. Terminals that are on are in contact with SCI and can be used to enter commands to control system operations. Terminals that are off are not in contact with SCI, cannot be used for command entry, and can only be used as physical input or output devices.
- A terminal can be set so that it requires you to log-on before it can be operated under SCI.
- VDTs can be set to operate in either TTY mode or VDT mode. In TTY mode, a VDT displays only one line of output at a time and accepts only one line of input at a time. In VDT mode, a full screen of output is displayed at one time and a full screen of input is accepted at one time. Data terminals (733/743/820) can operate only in TTY mode.

MTS (MODIFY TERMINAL STATUS)

Prompt Details:

TERMINAL NAME:
The device name of the terminal with status to be modified.

NEW STATUS (ON/OFF):
If you specify ON, you are allowed to enter commands to SCI from the terminal. If you specify OFF, you are prevented from accessing SCI at your terminal. The default value is the current status of the terminal.

NEW MODE (TTY/VDT):
Enter TTY to assign a terminal to TTY mode. Enter VDT to assign a video display terminal to VDT mode. The default mode is the current mode of the terminal.

DEFAULT MODE (TTY/VDT):
Enter TTY to assign TTY mode as the default mode of the terminal. Enter VDT to assign VDT mode as the default mode of the terminal. You must specify VDT as the DEFAULT MODE to perform horizontal scaling of data via the Show File (SF) command.

ASK FOR USER ID?:
If you specify YES, your user ID is required at the terminal before the terminal can be operated under SCI. Enter NO to access SCI at the terminal without specifying a user ID. A null response specifies that this attribute is unchanged.

ASK FOR JOB NAME?:
If you specify YES, you are prompted for a job name at log-on. If you specify NO, you will not be prompted for a job name during log-on. A null response specifies that this attribute is unchanged.

ASK FOR ACCOUNT ID?:
If YES is the response, you are prompted for an account ID at log-on. If NO, no account number is needed for terminal log-on. A null response specifies that this attribute is unchanged. An account ID can be used only if the accounting file .S$ACCVAL has been established to include this ID.

ASK FOR NAME MANAGER FILES?:
If YES is the response, you are prompted for the file names containing the synonyms and logical names. If NO, the standard manager file names for the specified user ID are used. A null response specifies that this attribute is unchanged.

RECONNECT ENABLED?:
If the response is YES when you log on, the current active jobs are scanned to see if another job with the same job name, user ID, and passcode exists. If these conditions are met, the following prompt is displayed at log-on:

RECONNECT?:

If you respond with YES, the specified task is bid in the job that is currently executing. If you enter NO, a new job is created. The default is NO.

MTS (MODIFY TERMINAL STATUS)

DEFAULT USER ID:
Enter a valid user ID. The job bid at the terminal executes under this ID.

DEFAULT PASSCODE:
Enter the passcode, if any, associated with the default user ID.

DEFAULT JOB NAME:
Enter a job name to be used whenever you log on at the terminal.

DEFAULT ACCOUNT ID:
Enter the account ID, if any, under which the job is to execute at the terminal.

Example:

In the following example, the MTS command sets the attributes of terminal ST07, and the entry of a List Terminal Status (LTS) command confirms that the attributes were set accordingly.

```
[] MTS

MODIFY TERMINAL STATUS
                TERMINAL NAME: ST07
          NEW STATUS (ON/OFF): ON
           NEW MODE (TTY/VDT): TTY
       DEFAULT MODE (TTY/VDT): VDT
             ASK FOR USER ID?: YES
            ASK FOR JOB NAME?: YES
          ASK FOR ACCOUNT ID?: YES
ASK FOR NAME MANAGER FILES?: NO
            RECONNECT ENABLED?: NO

[] LTS

LIST TERMINAL STATUS
                TERMINAL NAME: ST07
           OUTPUT ACCESS NAME: ME

TERMINAL    USER ID    LOGON REQUIRED   MODE   DEFAULT
ST07        JACKIE            Y           VDT     VDT
09:29:12 THURSDAY, NOVEMBER 03, 1983.
```

Special Cases:

When you enter a null response for an optional prompt, the existing value for that feature is unchanged. That is, if the MTS command is executed with VDT as the response to the NEW MODE prompt, then executed again with a null response to the NEW MODE prompt, the new mode remains VDT.

MTS (MODIFY TERMINAL STATUS)

Notes:

If SCI is currently activated at a terminal which is being modified, the new attributes do not take effect until after you enter the Quit (Q) command and reactivate SCI at that terminal. The Q command is discussed later in this section.

If you are using MTS in a file security environment, you must have read and write access rights to the .S$SCA file.

Related Commands:

LTS (List Terminal Status)

MTSA (MODIFY TASK SECURITY ATTRIBUTE)

4.157 MTSA (MODIFY TASK SECURITY ATTRIBUTE)

Prompts:

```
MODIFY TASK SECURITY ATTRIBUTE
     PROGRAM FILE PATHNAME:  filename@                (*)
               TASK NAME OR ID:  {character(s)/integer}
           SECURITY BYPASS?:  YES/NO                  (NO)
```

Purpose:

The MTSA command allows a user to specify or remove the security bypass attribute for a task in a program file. In most cases, this command will be restricted to use by the security manager or other designated individuals. A task that has the security bypass attribute can access any file in a DNOS system with file security without having security checked.

Prompt Details:

PROGRAM FILE PATHNAME:
Specify the filename of the program file in which the task resides.

TASK NAME OR ID:
The name or installed ID of the task to be affected.

SECURITY BYPASS?:
Respond YES to set the security bypass attribute for the specified task in the program file. A response of NO removes the security bypass attribute from the specified task.

Example:

In the following example, the MTSA command sets the security bypass attribute for the task named MYTASK in the program file named VOL2.BATCH.SALES1.

```
[] MTSA

MODIFY TASK SECURITY ATTRIBUTE
 PROGRAM FILE PATHNAME: VOL2.BATCH.SALES1
      MODULE NAME OR ID: MYTASK
       SECURITY BYPASS?: YES
```

Notes:

This command is normally restricted to use by only the security manager or other designated individuals. The user of this command must have both read and write access rights for the specified program file.

MUI (MODIFY USER ID)

4.158 MUI (MODIFY USER ID)

Prompts:

```
MODIFY USER ID
                        USER ID:  alphanumeric          (*)
          NEW USER DESCRIPTION:   [character(s)]
                  NEW PASSCODE:   [character(s)]
  USER PRIVILEGE CODE(0..7):      [integer]
```

Purpose:

The MUI command modifies the passcode and/or user privilege code associated with a user ID; the user ID itself cannot be changed.

Prompt Details:

USER ID:
A one- to eight-character user ID.

NEW USER DESCRIPTION:
A one- to twenty-character string, for information purposes only, which identifies the user. The default value is the description assigned to the user ID.

NEW PASSCODE:
A one- to eight-character string that comprises the new passcode to be associated with the user ID. The default value is the passcode previously assigned to the user ID.

USER PRIVILEGE CODE(0..7):
A number in the range of zero through seven that specifies the new privilege level at which the user can operate the system. The default value is the privilege code previously assigned to the user ID.

Example:

In the following example, a List User ID (LUI) command lists all the existing user IDs and their associated user descriptions and privilege levels, within the system, to the terminal local file. The MUI command changes the privilege level associated with the specified user ID and leaves the passcode unchanged. Another LUI command shows the user's changed status by displaying all existing user IDs and their associated information.

```
[] LUI

LIST USER IDS
     OUTPUT ACCESS NAME:

 USER ID          DESCRIPTION    PRIVILEGE CODE
 LARRY         LARRY                    4
 MARY          MARY B.                  7
 MIKE          MIKE P.                  3
 SYSTEM        SYSTEM VISITORS          4
 CRAFTY        TOM                      7

16:00:20 MONDAY, APRIL 11,1983.

[] MUI

MODIFY USER ID
                    USER ID: LARRY
        NEW USER DESCRIPTION: LARRY A. MICHAELS
                 NEW PASSCODE:
  USER PRIVILEGE CODE(0..7): 7

[] LUI

LIST USER IDS
     OUTPUT ACCESS NAME:

 USER ID          DESCRIPTION    PRIVILEGE CODE
 LARRY         LARRY A. MICHAELS        7
 MARY          MARY B.                  7
 MIKEP         MIKE P.                  3
 SYSTEM        SYSTEM VISITORS          4
 CRAFTY        TOM                      7

16:00:50 MONDAY, APRIL 11,1983.
```

MUI (MODIFY USER ID)

Notes:

In order to execute this command, the privilege level associated with the user ID must be four or higher.

When MUI is used in a file security environment, you must have read and write access rights to the .S$CLF file.

Related Commands:

AUI	(Assign User ID)
DUI	(Delete User ID)
LUI	(List User IDs)
MPC	(Modify Passcode)

4.159 MVI (MODIFY VOLUME INFORMATION)

Prompts:

```
MODIFY VOLUME INFORMATION
        CONTROL ACCESS NAME:   [site:]pathname@          (ME)
```

After you enter the response to the CONTROL ACCESS NAME prompt, the following prompt is displayed:

```
MVI
    DISK?:   [devicename@]
```

If you omit a response, the system disk is used. When you enter the response to the DISK? prompt, the following prompt is displayed:

```
COMMAND (L,C,S,Q)?: { L/C/S/Q }
```

If you enter L (List) for this prompt, the following items under control of MVI are displayed, followed by the COMMAND (L,C,S,Q)? prompt:

```
                    PRIMARY       SECONDARY          SELECT
      KERNEL FILE:  filename                             P
     UTILITY FILE:  filename                             P
     OVERLAY FILE:  filename                             P
      LOADER FILE:  filename                             P
         WCS FILE:  filename                             P
      DIAGNOSTIC:   filename      ------                 N
     VOLUME NAME:   alphanumeric
COMMAND (L,C,S,Q)?:
```

If you enter C (Change) in response to the COMMAND (L,C,S,Q)? prompt, the following is displayed:

```
WHICH ITEM (K,O,U,L,D,V,W)?: {K/O/U/L/D/V/W}
```

If the response to the WHICH ITEM (K,O,U,L,D,V,W)? prompt is K, O, U, L, or W, the following prompts are displayed:

```
     PRIMARY:   [filename]
   SECONDARY:   [filename]
      SELECT:   {P/S/T}
```

The SELECT prompt is also displayed along with the following prompt if the response to WHICH ITEM (K,O,U,L,D,V,W)? is D:

```
  DIAGNOSTIC:   [filename]
     SELECT?:   YES/NO
```

MVI (MODIFY VOLUME INFORMATION)

If you enter V in response to WHICH ITEM (K,O,U,L,D,V,W)?, the following is displayed:

VOLUME NAME: alphanumeric

If the response to the COMMAND (L,C,S,Q)? prompt was S (Swap), the following prompt is displayed:

WHICH ITEM (K,O,U,L,W)?: {K/O/U/L/W}

If you enter Q in response to the COMMAND (L,C,S,Q)? prompt, the MVI command terminates and you return to SCI.

Purpose:

The MVI command modifies DNOS disk volume information. Under MVI, specific file names can be listed (L) and/or changed (C). Terminate MVI at any time by entering the Q character in response to the COMMAND (L,C,S,Q)? prompt and then pressing the Return key. If the C (Change) or S (Swap) options are specified in response to the COMMAND (L,C,S,Q)? prompt, volume information before and after the change or swap is written to the .S$SYSTEM.S$HSTRY file. This information is written to the .S$SYSTEM.S$HSTRY file once per MVI command execution. Information written to the .S$SYSTEM.S$HSTRY file for successive executions of the MVI command where the C or S options are specified is separated by a blank line. The volume information written to this file consists of the kernel, overlay, utility, loader, and WCS file names. The information is in the same format as that displayed when the L option is specified during MVI execution.

NOTE

The Return key is the only SCI function key that is operational during MVI execution. Alphabetic and numeric keys on the terminal operate normally. The entire response to each prompt must be keyed in. That is, if you key in a response, then backspace over part of the response, you must rekey the remainder of the response.

The files and information under the control of MVI include the following:

File	Designator
Kernel	(K)
Overlay	(O)
Utility	(U)
Loader	(L)
WCS	(W)
Diagnostic	(D)
Volume Name	(V)

MVI (MODIFY VOLUME INFORMATION)

The MVI processor first prompts for the disk drive name for the disk with information to be modified. The COMMAND (L,C,S,Q)? prompt is then displayed allowing the user to List (L), Change (C), Swap (S), or Quit (Q) the MVI command. Swap (S) is used to swap the primary and secondary names. Should you want to list those items under control of MVI, the following information is displayed:

	PRIMARY	SECONDARY	SELECT
KERNEL FILE:	filename	filename	P,S, or T
UTILITY FILE:	filename	filename	P,S, or T
OVERLAY FILE:	filename	filename	P,S, or T
LOADER FILE:	filename	filename	P,S, or T
WCS FILE:	filename	filename	P,S, or T
DIAGNOSTIC:	filename		N or Y
VOLUME NAME:	alphanumeric		

NOTE

The overlay file is not used by DNOS. This file is included in the MVI command for DX10 compatibility only.

The PRIMARY and SECONDARY columns specify the file names currently installed for the functions listed at the left and selected by the SELECT field. The meaning of the SELECT field code listed under the SELECT column is as follows:

Code	Meaning
P	Primary
S	Secondary
T	Test
N	No
Y	Yes

The KERNEL FILE, OVERLAY FILE, UTILITY FILE, LOADER FILE, and WCS FILE allow the SELECT codes P, S, or T. The SELECT codes N and Y apply only to the DIAGNOSTIC function. The file names listed and those you enter in response to prompts are actual file names. You must enter these file names in the form specified. The files are assumed to be at the volume directory level, i.e., VOL1.S$UTIL would be entered as S$UTIL.

You can change any of the six files by entering a new value. For system volumes, you can select load from the primary image the next time an initial program load (IPL) is performed (P); load from the secondary image the next time an IPL is performed (S); or load from the secondary file the next time an IPL is performed (T). Use the code T for testing a newly created image of DNOS much as you would use the Test Generated System (TGS) command.

MVI (MODIFY VOLUME INFORMATION)

In order to use a new operating system after it has been generated, assembled, and link edited, you can execute the MVI command, specify the output file created by the link editor as the secondary KERNEL FILE, and enter T (Test) for the SELECT field. An IPL must be performed to activate the new system. If the system is satisfactory, you should again execute the MVI command, change the PRIMARY image to the file name which was the test version and enter P in the SELECT field.

If you enter Y (Yes) for the SELECT field for the diagnostic file, the response to the DIAGNOSTIC prompt determines the file to be loaded. Performing an IPL for the DNOS system is not necessary. As with the T response for the SELECT field, Y is a one-time function which reverts to N (No) after an IPL.

Should you want to change any of the information displayed, you can enter the character C (Change) and then press the RETURN key in response to the COMMAND (L,C,S,Q)? prompt. When you enter C for this prompt, the WHICH ITEM (K,O,U,L,D,V,W) prompt is displayed. This prompt allows you to enter the character designating the file or information to be changed. After you enter the character and press the Return key, specify the files or information in response to further prompts.

If you select the character K, O, U, L, or W, the MVI prompts for PRIMARY, SECONDARY, and SELECT. For each prompt, the current value is displayed after the prompt. You can accept the displayed value by pressing the Return key or can change it by entering the desired new value and pressing the Return key. To delete a value, you must enter as many blanks as there are characters in the original value. Refer to the Notes section of this command if the value for the KERNEL FILE prompt was entered incorrectly.

If you enter D in response to the WHICH ITEM prompt, the DIAGNOSTIC and SELECT prompts are displayed. You can accept the current DIAGNOSTIC and SELECT prompt values or change them. The same current value and change procedure as described above apply.

If you select V for the WHICH ITEM prompt, the VOLUME NAME prompt is displayed. You can accept the current value or change it. The same current value and change procedure as described above apply. Disk volume names must not be the same as disk device names (for example, DS01, DS02).

If a syntax error is found in any of the file names entered in response to prompts, the following message is displayed:

```
FILE NAME SYNTAX ERR
```

You can terminate MVI by responding to the COMMAND (L,C,S,Q)? prompt with the Q character and pressing the Return key.

MVI (MODIFY VOLUME INFORMATION)

Prompt Details:

CONTROL ACCESS NAME:
The device or file name from which MVI is to receive its input. If executed on an interactive device, prompts are displayed; otherwise, responses are accepted, one per record, without prompts.

DISK?:
The device name of the disk drive that contains the disk with information to be modified. The valid responses are in the range of DS01 through DS99 or a synonym with the value of the device name. The volume name or its synonyms can also be specified. DSO1 is the default if a null value is entered.

KERNEL FILE:
The system image file name that contains memory-resident parts of DNOS.

OVERLAY FILE:
The system overlay file name that consists of a relative record file containing disk-resident overlays of DX10. This file is included in the MVI command for DX10 compatibility only.

UTILITY FILE:
The system program file name that contains system programs and optionally can contain user programs.

LOADER FILE:
The system loader file name that is an image file that loads the DNOS operating system from the system image (kernel) file. The file name can contain a maximum of eight characters.

WCS FILE:
The Writable Control Storage (WCS) file name that contains predefined XOP instructions. The WCS file is for use on 990/12 computers with supplied software.

DIAGNOSTIC:
The file name of the diagnostic file. Under DNOS, the diagnostic file, S$DIAG, is created by the Initialize New Volume (INV) SCI command. It is a read-only file containing pseudorandom data patterns used for system verification by diagnostic programs. This file occupies exactly one disk cylinder and should be located as close as possible to the disk spindle. (If the closest cylinder has a bad track, the next closest is used, and so on. S$DIAG always occupies an error-free cylinder.)

COMMAND (L,C,S,Q)?:
If you enter L (List) in response to this prompt, the items under control of the MVI command are listed. If you enter C (Change), the WHICH ITEM prompt is displayed and you have the option of choosing which item under control of the MVI command to change. If you enter S, the primary and secondary names are swapped. If you enter Q, MVI terminates.

MVI (MODIFY VOLUME INFORMATION)

WHICH ITEM (K,O,U,L,D,V,W)?:

You can enter the character K, O, U, L, D, V, or W. These characters represent items that are under control of the MVI command and that can be changed as follows:

File	Designator
Kernel	(K)
Overlay	(O)
Utility	(U)
Loader	(L)
WCS	(W)
Diagnostic	(D)
Volume Name	(V)

PRIMARY:

The file name for an item to be loaded the next time an IPL is performed if the response to the SELECT prompt is P (primary).

SECONDARY:

The file name for an item to be loaded the next time an IPL is performed if the response to the SELECT prompt is S (secondary).

SELECT:

You can enter the following characters in response to the SELECT prompt:

Code	Meaning
P	Primary
S	Secondary
T	Test
N	No
Y	Yes

The KERNEL FILE, OVERLAY FILE, UTILITY FILE, LOADER FILE, and WCS FILE functions allow the select codes P, S, or T. If you enter P (primary), the file specified as the primary file for the function is loaded at the next IPL. If you enter S (secondary), the file specified as the secondary file for the function is loaded at the next IPL. If you enter T (test), the secondary file is loaded when an IPL is performed to the DNOS system. (The secondary file should be a generated, assembled, and linked output file for testing.) The T select reverts to a P select if an IPL to the DNOS system is performed a second time.

The select codes N (no) and Y (yes) apply only to the DIAGNOSTIC function. If you enter Y in response to this prompt, the file specified for the DIAGNOSTIC function is loaded. The Y select reverts to an N select at the next IPL.

MVI (MODIFY VOLUME INFORMATION)

VOLUME NAME:
The name of the disk volume with volume information being modified. The current volume name specified can be used, or a new name can be specified. If you specify a new volume name, the disk must be unloaded with the Unload Volume (UV) command using the old name and then installed with the Install Volume (IV) command using the new name. If you change the volume name of the system disk, the change becomes effective at the next IPL.

Example:

In the following example, the MVI command obtains a list of files and information subject to change, including the diagnostic file. User action and system returns are documented.

```
[] MVI

MODIFY VOLUME INFORMATION
    CONTROL ACCESS NAME: ME
```

System Prompts	User Responses
MVI	
DISK? DS01	Disk volume to be modified. User enters disk unit which holds the disk volume to be modified.
COMMAND (L,C,S,Q)? L	User enters list option.
PRIMARY SECONDARY SELECT KERNEL FILE: RLSYS SYS2 P OVERLAY FILE: S$OVLYA P UTILITY FILE: S$PROGA P LOADER FILE: S$LOADER P WCS FILE: DIAGNOSTIC: S$DIAG --------- N VOLUME NAME: A19	Files that can be modified are displayed.
COMMAND (L,C,S,Q)? C	User enters change option.
WHICH ITEM (K,O,U,L,D,V,W)? D DIAGNOSTIC: S$DIAG TEST SELECT: N Y	D (diagnostic) file specified. User enters secondary file to be used, and specifies Y for secondary selection.
COMMAND (L,C,S,Q)? L	User enters list option.
PRIMARY SECONDARY SELECT KERNEL FILE: RLSYS SYS2 P OVERLAY FILE: S$OVLYA P UTILITY FILE: S$PROGA P LOADER FILE: S$LOADER P WCS FILE: DIAGNOSTIC: TEST --------- Y VOLUME NAME: A19	Files that can be modified are displayed. TEST is listed as the primary diagnostic file.
COMMAND (L,C,S,Q)? Q	User enters quit option.
MVI TERMINATED	MVI terminates.

MVI (MODIFY VOLUME INFORMATION)

Messages:

BAD COMMAND
INVALID ITEM
INVALID SELECTION
FILE NAME SYSTEM ERROR
WARNING: SYSTEM IMAGE NOT VERIFIED

Notes:

If the system image (kernel) file is not found on the disk or if the disk is not installed, the following warning message is displayed:

U UTILITY - 0514 SELECTED SYSTEM DOES NOT EXIST ON THIS DISK

If you change the name of an installed volume through the VOLUME NAME prompt of the MVI command, you must unload the volume under the old name and install it with the new name.

If the MVI command is performed on an uninstalled volume, an error message SVC-0304 results because an attempt is made to write to the .S$MVI file on the volume. The MVI command executes successfully, but the .S$MVI file is not updated. The situation does not cause any further error.

If the MVI command is performed on a write-protected disk, changes cannot be made and an error message is issued.

4.160 MVS (MODIFY VOLUME SECURITY)

Prompts:

```
MODIFY VOLUME SECURITY
                    VOLUME NAME:   alphanumeric@
SECURE/UNSECURE VOLUME(S,U):   {S/SECURE/U/UNSECURE}
```

Purpose:

The MVS command permits you to specify that a volume be installable or uninstallable on a system that has no file security. Once the volume has been made uninstallable, it cannot be installed on a DNOS system that does not have file security nor can it be installed on a DX10 system.

Prompt Details:

VOLUME NAME:
Specify the name of the volume to be modified.

SECURE/UNSECURE VOLUME(S,U):
A response of S or SECURE causes the specified volume to be made uninstallable. A response of U or UNSECURE causes the volume to be made installable.

Example:

In the following example, the MVS command modifies the disk volume named VOL1 to be secure (uninstallable on a system without file security).

```
[] MVS

MODIFY VOLUME SECURITY
                    VOLUME NAME: VOL1
SECURE/UNSECURE VOLUME(S,U): SECURE
```

Notes:

If this command has not been used on a volume since the last time an Initialize New Volume (INV) command was used on it, the volume is installable on systems without file security.

In most environments, this command is likely to be restricted to use by the system security manager.

MWR (MODIFY WORKSPACE REGISTERS)

4.161 MWR (MODIFY WORKSPACE REGISTERS)

Prompts:

```
MODIFY WORKSPACE REGISTERS
                    RUN ID:  integer          (*)
         REGISTER NUMBER:    integer exp      (0)
```

Purpose:

The MWR command modifies specific workspace registers of a task. If the task is not unconditionally suspended, it is temporarily suspended while the command is modifying the registers.

As with the Modify Memory (MM) command, the MWR command is for interactive use only; you must press the Return key to display the contents of the next workspace register or to enter a new value for a workspace register. Also, press the Command key to terminate the MWR command and place SCI in command mode.

If the workspace registers to be modified are for a system task, you must have a user ID with a privilege level of two or higher.

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

REGISTER NUMBER:
An integer value from 0 through 15 that is the beginning workspace register to be displayed.

MWR (MODIFY WORKSPACE REGISTERS)

Example:

In the following example, the MWR command displays the workspace registers, beginning with register 2, for a task with a run ID of > A5. Registers 2 through 15 are displayed, and registers 2 and 3 are modified.

```
[] MWR

MODIFY WORKSPACE REGISTERS
               RUN ID: >A5
      REGISTER NUMBER: 2

RUN ID: A5 STATE=06 WP=FFBE PC=1026 <PC>=1024 ST=8000 LA P M

             R2: 0200  >FFFF
             R3: 0300  >3FFF
             R4: 062F
             R5: 8010
             R6: 0800
             R7: 49A8
             R8: 0000
             R9: 8000
            R10: 0000
            R11: 0000
            R12: 0000
            R13: 6F06
            R14: 6FEC
            R15: C5CF
```

Notes:

In the above example, the terminal is in VDT mode. In TTY mode, the MWR command does not display all registers at once and does not display the task run ID, state, or internal registers. In VDT mode, pressing the Command key causes the debug panel to be displayed before SCI is reactivated. The explanation of the Show Panel (SP) command provides an example of the debug panel.

Related Commands:

SWR (Show Workspace Registers)

PB (PROCEED FROM BREAKPOINT)

4.162 PB (PROCEED FROM BREAKPOINT)

Prompts:

```
PROCEED FROM BREAKPOINT
                    RUN ID:  integer                (*)
DESTINATION ADDRESS(ES):  [full exp list]
```

Purpose:

The PB command activates a task that is halted at a breakpoint, leaving the breakpoint active. If the task is not currently halted at a breakpoint when you execute PB, you are so notified by a warning message, and the task remains in its current state.

Optionally, PB sets one or more additional breakpoints in the task. Breakpoints are set according to the destination address values you supply. If you do not supply a value, no additional breakpoints are set.

Prompt Details:

RUN ID:
A valid run ID in your job. You can obtain current run IDs by executing the Show Task Status (STS) command.

DESTINATION ADDRESS(ES):
The integer value(s) of the address(es) within the task where the new breakpoints are to occur. Separate addresses by a comma. The default value is no new breakpoints.

Example:

In the following example, the PB command assigns new breakpoints within a task with a run ID of >9A.

```
[] PB

PROCEED FROM BREAKPOINT
                    RUN ID: >9A
  DESTINATION ADDRESS(ES): >A0,>10,>A14,>2B
```

Assumptions:

The task is at a breakpoint when you execute the PB command.

Related Commands:

AB	(Assign Breakpoint)
DB	(Delete Breakpoints)
DPB	(Delete and Proceed from Breakpoint)
LB	(List Breakpoints)

PBP (PROCEED FROM BREAKPOINT — PASCAL)

4.163 PBP (PROCEED FROM BREAKPOINT — PASCAL)

The PBP command assigns a breakpoint and resumes execution of a Pascal task. Refer to the *DNOS TI Pascal Programmer's Guide* for further information on the PBP command and the following prompts:

```
PROCEED FROM BREAKPOINT — PASCAL
                    RUN ID:  integer exp                (*)
       DESTINATION ROUTINE:  [character(s)]
WHERE(ENTRY/RETURN/BOTH):  [{ENTRY/RETURN/BOTH}]
```

PF (PRINT FILE)

4.164 PF (PRINT FILE)

Prompts:

PRINT FILE

FILE PATHNAME(S):	[site:]filename@ . . . [site:]filename@	(*)
ANSI FORMAT?:	YES/NO	(NO)
LISTING DEVICE OR CLASS:	[site:]alphanumeric@	(*)
DELETE AFTER PRINTING?:	[YES/NO]	(NO)
NUMBER OF LINES/PAGE:	[integer]	(62)
FORM:	[alphanumeric]	(STANDARD)
NUMBER OF COPIES:	integer	(1)
BANNER SHEET?:	YES/NO	(NO)

Purpose:

The PF command places files in a waiting line (output queue) from which their contents are printed at a printing device.

When printing a file in response to a PF command, the output is single spaced unless the file contains carriage control characters. Files containing carriage control characters (except for ANSI format control characters) are automatically printed in the format specified by these control characters. Files containing ANSI format carriage control characters are single spaced when printed unless you select ANSI format when you enter the PF command.

ANSI carriage control characters in the first position of each record have the following meanings:

blank	—	Single space before printing
0	—	Double space before printing
1	—	Top of page before printing
+	—	Suppress line space (overprint)
#	—	Elongated text (OE output to printer)

If PF output cannot be performed (for example, because the device is offline, an error message is written to the system log.

Prompt Details:

FILE PATHNAME(S):
A filename(s) that identifies the file(s) with contents to be printed. The response can contain more than one pathname of files to be placed on the output queue and printed, but the pathnames must be separated by commas. Sequential or relative record files can be printed.

ANSI FORMAT?:
If you enter YES in response to this prompt, the file being printed contains ANSI carriage control characters. NO specifies that the file being output does not contain ANSI control characters. If you specify YES and the files do not contain ANSI control characters, the output may not be what you expected.

PF (PRINT FILE)

LISTING DEVICE OR CLASS:
A device name or class name of a device to which the contents of the file(s) are to be printed.

DELETE AFTER PRINTING?:
If you enter YES in response to this prompt, the file being printed is deleted after it is printed. NO specifies that it should not be deleted. The default value is NO.

NUMBER OF LINES/PAGE:
The number of lines (from 1 through 255) to be printed on one page. If 255 is specified, the spooler does not insert any page ejects. The default value is 62.

FORM:
The name of the form that should be mounted on the output device. If you request a form but it is not currently mounted, a message is sent to the system operator requesting a change of form. When the request is granted, the new form is used. The default form is STANDARD.

NUMBER OF COPIES:
The number of copies to be printed. The maximum number of copies is 255.

BANNER SHEET?:
If you enter YES, an identifier sheet is printed as the first page of the output. The default value is NO.

Example:

In the following example, the PF command prints the contents of the file VOL1.DIR2.FILE1 to a line printer:

```
[] PF

PRINT FILE
          FILE PATHNAME(S): VOL1.DIR2.FILE1
              ANSI FORMAT?: NO
  LISTING DEVICE OR CLASS: LP02
   DELETE AFTER PRINTING?: NO
     NUMBER OF LINES/PAGE: 62
                     FORM: STANDARD
          NUMBER OF COPIES: 1
             BANNER SHEET?: NO
```

PF (PRINT FILE)

Notes:

Characters with a value of less than > 20 (space) are control characters. These characters are not printable and can cause formatting problems in the printed output.

Any records larger than the largest record size handled by the print device are truncated.

Related Commands:

SF	(Show File)
SOS	(Show Output Status)
MSD	(Modify Spooler Device)

4.165 PGS (PATCH GENERATED SYSTEM)

Prompts:

```
PATCH GENERATED SYSTEM
                DATA DISK/VOLUME:  [site:]{devicename/alphanumeric}  (*)
              TARGET DISK/VOLUME:  [site:]{devicename/alphanumeric}  (*)
                     SYSTEM NAME:  alphanumeric                      (*)
                   PATCH KERNEL?:  YES/NO                            (YES)
                  PATCH UTILITY?:  YES/NO                            (NO)
PATCH COMM(NONE,SOME,ALL):        {N/NONE/S/SOME/A/ALL}             (NONE)
```

If you enter YES to the PATCH UTILITY? prompt, the following prompts are displayed:

```
PATCH GENERATED SYSTEM
          UTILITY PROGRAM FILE:  alphanumeric  (S$UTIL)
                  IPL FILE NAME:  alphanumeric  (S$IPL)
COMMAND PROCEDURE DIRECTORY:      alphanumeric  (S$CMDS)
```

If you enter SOME in response to the PATCH COMM(NONE,SOME,ALL) prompt, the following prompt is displayed:

```
PATCH GENERATED SYSTEM
    COMM PROTOCOL NAME(S):  character(s) . . . character(s)
```

Purpose:

After the successful completion of either the Assemble and Link Generated System (ALGS) command or the Execute System Generation Utility (XSGU) command with the assemble and link option, the new operating system resides in the kernel program file. The name of this file was given in response to the SYSTEM NAME prompt. The system image must be patched before an initial program load (IPL) can be performed. The PGS command is used to patch the new system.

The PGS process builds a batch stream containing commands to execute the appropriate batch streams. The batch stream is placed in the .SSGU.systemname.PGS file on the target disk and is executed by PGS. The batch SCI listing is written to the file .SSGU.systemname.PATCHLST.PGS on the target disk.

After termination of PGS, the new system image is ready to be tested and/or installed as the executing operating system.

PGS (PATCH GENERATED SYSTEM)

Prompt Details:

DATA DISK/VOLUME:
The device or volume name that contains the system linkable object directories and patch files. The initial value is the same as the response to the last DATA DISK/VOLUME prompt of either the ALGS, XSGU, or PGS command.

TARGET DISK/VOLUME:
The device or volume name that contains the newly generated system. The initial value is the same as the response to the last TARGET DISK/VOLUME prompt of either the ALGS, XSGU, or PGS command.

SYSTEM NAME:
A one- to eight-character alphanumeric string that is the name of the generated system. The initial value is the same as the response to the last SYSTEM NAME prompt of either the ALGS, XSGU, or PGS command.

PATCH KERNEL?
If YES is the response, the kernel patches are applied to the specified system name. Respond NO if you do not want to apply kernel patches.

PATCH UTILITY?:
A YES response causes additional prompts to be displayed. Respond NO if you do not want to apply utility patches.

PATCH COMM(NONE,SOME,ALL):
A response of ALL causes all communications patches for the protocols you have generated to be applied to the generated system. Respond NONE if you do not want to apply communications patches. A response of SOME causes additional prompts to be displayed, and only the selected communications patches are applied.

UTILITY PROGRAM FILE:
A one- to eight-character alphanumeric string that is the name of the utility program file to be patched. This is usually S$UTIL. Enter the exact characters of the last component of the file name. You cannot use logical names or synonyms. The file must reside in the VCATALOG directory on the target disk.

IPL FILE NAME:
A one- to eight-character alphanumeric string that is the name of the system loader file to be patched. This is usually S$IPL. Enter the exact characters of the last component of the file name. You cannot use logical names or synonyms. The file must reside in the VCATALOG directory on the target disk.

COMMAND PROCEDURE DIRECTORY:
A one- to eight-character alphanumeric string that is the name of the command procedure directory to be patched. This is usually S$CMDS. Enter the exact characters of the last component of the directory name. You cannot use logical names or synonyms. The directory must reside in the VCATALOG directory on the target disk.

PGS (PATCH GENERATED SYSTEM)

COMM PROTOCOL NAME(S):
A communications protocol name(s) that identifies the communications protocol(s) patches to be applied. The valid responses are 3780, 2780, 3270, RTS, SDLC, LAP, COMA, and CMNS. More than one protocol name can be entered. Separate each protocol name with a comma.

Example:

In the following example, a new system has been generated. The name specified for the system is SYS1. The PGS command patches the system image, utilities, and 3780 communications.

```
[] PGS

PATCH GENERATED SYSTEM
              DATA DISK/VOLUME: VOL1
            TARGET DISK/VOLUME: VOL1
                   SYSTEM NAME: SYS1
                 PATCH KERNEL?: YES
               PATCH UTILITY?: YES
   PATCH COMM(NONE,SOME,ALL): SOME

PATCH GENERATED SYSTEM
         UTILITY PROGRAM FILE: S$UTIL
                 IPL FILE NAME: S$IPL
 COMMAND PROCEDURE DIRECTORY: S$CMDS

PATCH GENERATED SYSTEM
        COMM PROTOCOL NAME(S): 3780
```

Notes:

The PGS listing files will be placed in the .SSGU.systemname.PATCHLST directory on the target disk. The file names of the various listings are as follows:

KERNEL	— Kernel patch listing
< utility program file>	— Utility patch listing
COMPATCH	— COMPATCH batch stream listing
COMM	— COMM batch stream listing
DNPCSWS	— COMSWS patch listing
CMON< protocol name>	— Common communications protocol DSR patch listing
DNP< protocol name>	— Communications protocol DSR patch listing
PGS	— PGS batch listing

Related Commands:

ALGS	(Assemble and Link Generated System)
IGS	(Install Generated System)
TGS	(Test Generated System)

Q (QUIT SCI)

4.166 Q (QUIT SCI)

Purpose:

The Q command terminates interactive SCI at a terminal and saves the current set of synonyms and logical names.

If the Text Editor or Debugger is active at the terminal, the Q command automatically prompts for the Quit Edit (QE) or Quit Debug Mode (QD) command responses. The following message is displayed following the entry of responses to either the QE or QD command:

QUIT PROCESSING INCOMPLETE: RE-ENTER Q COMMAND

To terminate SCI at the terminal, press the Return key and reenter the Q command.

If a background task is active at the terminal when you enter the Q command the following message is displayed:

U SCI-0029 CANNOT QUIT WITH BACKGROUND TASK PENDING

You must allow the background task to complete. Otherwise, it can be terminated using the Kill Background Task (KBT) command. You must then reenter the Q command following background task completion in order to terminate SCI at the terminal. Should any of the tasks EDITOR, DEBUGGER, SCU, XOI, or ANALZ be pending, you will have to terminate the sessions before entering the Q command and deactivating SCI.

Permanent copies of the synonyms and logical names are kept in files on the system disk. When you log off the terminal, the files are updated. Since several users may log on with the same user ID, each user begins with the current set of synonyms and logical names from the current copy of the files. When each user logs off, his current set of synonyms and logical names is saved to the files. Thus, multiple use of the same user ID requires cooperation by the users when assigning and deleting synonyms and logical names.

Each time you execute the Q command SCI automatically calls and executes the Clear Synonyms (Q$SYN) command. The Q$SYN command clears synonyms used by other command procedures and other synonyms you added. A Show File (SF) command on the file .S$CMDS.Q$SYN shows the synonyms that are deleted.

Example:

In the following example, the Q command terminates SCI at the terminal where the command was executed.

```
[] Q
```

Related Commands:

Q$SYN (Clear Synonyms)

4.167 QD (QUIT DEBUG MODE)

Prompts:

QUIT DEBUG MODE
KILL TASK?: YES/NO (YES)

Purpose:

The QD command takes a task out of debug (controlled) mode. You have the option of killing the task at this point. If you choose not to kill the task, it is left unconditionally suspended, but you can still issue any of the general SCI commands. The Resume Task (RT) or Proceed from Breakpoint (PB) commands (depending on whether the task is at a breakpoint) can be used to activate the task.

The RT command is discussed later in this section.

Prompt Details:

KILL TASK?:
If you enter YES, the task currently in debug mode is killed. The task then executes its end-action routine. If you enter NO, the task currently in debug mode is unconditionally suspended.

Example:

In the following example, the QD command takes the currently executing task out of controlled mode and kills task execution.

```
[] QD

QUIT DEBUG MODE
        KILL TASK?:  YES
```

Related Commands:

XD (Execute in Debug Mode)

QE (QUIT EDIT)

4.168 QE (QUIT EDIT)

After you activate the Text Editor, the QE command terminates Text Editor processing. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the QE command and the following prompts:

```
QUIT EDIT
                          ABORT?:  YES/NO              (NO)
```

If N or NO (No is the initial value supplied) is entered in response to the ABORT? prompt, the following is displayed:

```
QUIT EDIT
    OUTPUT FILE ACCESS NAME:  [site:]filename@       (*)
                   REPLACE?:  YES/NO                 (NO)
        MOD LIST ACCESS NAME:  [site:][pathname]@
```

QOI (QUIT OPERATOR INTERFACE)

4.169 QOI (QUIT OPERATOR INTERFACE)

Purpose:

The QOI command terminates the operator interface task that was bid by the Execute Operator Interface (XOI) command. You can only execute this command from the operator console (the terminal where the XOI command was entered).

Example:

In the following example, the QOI command terminates the operator interface task:

```
[] QOI
```

Notes:

The system operator can receive one more request message before the command terminates if the message was sent before the QOI command was processed. This message may already have been killed using the Kill Operator Interface Request (KOR) command. If the operator has entered the Receive Operator Messages (ROM) command, the QOI command also acts as a Kill Operator Messages (KOM) command.

Related Commands:

COM	(Create Operator Message)
KOM	(Kill Operator Messages)
KOR	(Kill Operator Interface Request)
LOM	List Operator Messages)
ROM	(Receive Operator Messages)
ROR	(Respond to Operator Interface Request)
XOI	(Execute Operator Interface)

QSCU (QUIT CONFIGURATION UTILITY SESSION)

4.170 QSCU (QUIT CONFIGURATION UTILITY SESSION)

Prompts:

```
QUIT CONFIGURATION UTILITY SESSION
                    ABORT?:  YES/NO          (NO)
```

Purpose:

The QSCU command terminates the Execute System Configuration Utility (XSCU) command.

Prompt Details:

ABORT?:
: If you enter YES in response to this prompt, none of the modifications made during the SCU session are applied to the specified disk image. If you enter NO, the modifications are applied.

Example:

In the following example, the QSCU terminates the XSCU command and applies all modifications made during the XSCU session.

```
[] QSCU

QUIT CONFIGURATION UTILITY SESSION
            ABORT?: NO
```

Related Commands:

XSCU (Execute System Configuration Utility)

4.171 Q$SYN (CLEAR SYNONYMS)

Purpose:

The Q$SYN command clears synonyms used by several supplied command procedures and any other synonyms that you may have added to the procedure. Q$SYN is always called by SCI when you log off the terminal. However, you can invoke Q$SYN at any time after you log on to remove unnecessary synonyms from the currently defined set.

A Show File (SF) command on file .S$CMDS.Q$SYN shows the synonyms cleared by the Q$SYN command.

Example:

In the following example, the Q$SYN command clears synonyms for the terminal where the command executed.

```
[] Q$SYN
```

RAL (RELEASE ALL LOGICAL UNITS)

4.172 RAL (RELEASE ALL LOGICAL UNITS)

Prompts:

```
RELEASE ALL LOGICAL UNITS
                ARE YOU SURE?:  YES/NO
```

Purpose:

The RAL command releases all job-local LUNO's except LUNO 0.

Prompt Details:

ARE YOU SURE?:
You must enter YES or NO to confirm or deny that you want to release all job-local LUNOs (except LUNO 0).

Example:

In the following example, the RAL command releases all job-local LUNOs assigned to I/O resources from your job:

```
[] RAL

RELEASE ALL LOGICAL UNITS
        ARE YOU SURE?: YES
```

Related Commands:

AL	(Assign LUNO)
RL	(Release LUNO)

RCRU (READ CONTENTS OF SPECIFIED CRU ADDRESS)

4.173 RCRU (READ CONTENTS OF SPECIFIED CRU ADDRESS)

Prompts:

```
READ CONTENTS OF SPECIFIED CRU ADDRESS
                CRU ADDRESS:  integer
        OUTPUT ACCESS NAME:  [site:][pathname]@
```

Purpose:

The RCRU command reads the contents of the specified CRU register. This command transfers 16 bits from the specified CRU offset and writes four hexadecimal digits to the specified access name.

Prompt Details:

CRU ADDRESS:
The CRU address offset from which to read 16 bits of information. (Must be less than or equal to >1FE0.)

OUTPUT ACCESS NAME:
The device or file name to which the RCRU command writes the CRU data produced by the read operation. The default value is the terminal local file.

Example:

In the following example, the RCRU command reads 16 bits of information from address >0100 and outputs the information to the terminal.

```
[] RCRU

READ CONTENTS OF SPECIFIED CRU ADDRESS
             CRU ADDRESS: >0100
     OUTPUT ACCESS NAME:
```

The following information is displayed on the user's terminal:

```
CRU REGISTER VALUE: >0DA0
```

Related Commands:

WCRU (Write Value to Specified CRU Address)

RD (RESTORE DIRECTORY)

4.174 RD (RESTORE DIRECTORY)

Prompts:

```
RESTORE DIRECTORY
    SEQUENTIAL ACCESS NAME:  [[site:]pathname/[site:]device
                             name . . . [site:]device name]@      (*)
      DIRECTORY PATHNAME:    [pathname]@
    CONTROL ACCESS NAME:     [site:][pathname]@
    LISTING ACCESS NAME:     [site:][pathname]@
                OPTIONS:     [character(s) . . . character(s)]    (ADD)
  EXECUTION MODE(F,B):       {F/FOREGROUND/B/BACK-
                             GROUND} (FOREGROUND)
```

Depending on your responses, you might receive one or more additional prompts:

```
PATHNAME OF BACKUP FILE
                PATHNAME:  pathname@

          DATE OF OLDEST FILE
                    YEAR:  integer
                   MONTH:  integer
                     DAY:  integer
                    HOUR:  [integer]          (0)
                  MINUTE:  [integer]          (0)
```

Purpose:

The RD command restores a directory from a backup file. RD can handle any backup file created by the Backup Directory (BD) and the Backup Directory to Device (BDD) commands, including multiple volume files on magnetic tape or disk.

RD honors all of the options available with BD for specifying the files you want to include in the restore operation. Unless you tell it otherwise RD restores all the files (and aliases) in the backup file, including any subdirectories and files they contain. Your choice of the ADD or REPLACE option tells RD what to do when it finds that the directory already has a copy of the file being restored. Also like BD, you can use the Verify Backup (VB) command to compare the contents of the backup file and the restored directory.

RD allows you to use a *control file* containing directives that identify the directory, backup file, and files involved in the operation. For information on the control file directives, see the notes for BD.

Do not have other activity going on to the directory while it is being restored. Activities such as creating files can cause inconsistent file structures to be created if a restore operation is in progress.

RD (RESTORE DIRECTORY)

Prompt Details:

SEQUENTIAL ACCESS NAME:

The access name for the backup file. You can enter a file name, a device name, or a list of device names. If you enter the device name of a disk drive (or a list of device names DS01 through DS99), RD displays an additional prompt for the pathname of the backup file. When restoring from a backup file made by BDD or BD to device name, you need to enter a device name here. For details on multiple volume backups, refer to the special cases for RD. You can skip this prompt only if you specify the sequential access name with a MOVE directive in a control file.

DIRECTORY PATHNAME:

The pathname of the directory that you want to receive the contents of the backup file. If this directory does not already exist, RD automatically creates it with the same number or entries as the directory backed up with BD. Unless you limit the command with options or control file directives, this directory receives all files (and aliases) in the backup file, including any subdirectories and the files they contain. You can skip this prompt only if you specify the directory pathname with a MOVE directive in a control file.

CONTROL ACCESS NAME:

The access name of the control file. The control file contains directives that determine which files in the directory take part in the operation. If you enter a pathname of a sequential file, RD uses the directives it finds there. If you enter ME, RD expects you to enter control file directives from your station. For details on control files, refer to the notes for BD. If you respond to both the DIRECTORY PATHNAME and SEQUENTIAL ACCESS NAME prompts, you do not need to specify a control file.

LISTING ACCESS NAME:

The device name or file name where you want to receive listings of the backup operation. If you skip this prompt, RD assumes by default that you want the listings sent to the terminal local file of your station. If the listing file reports an error in restoring a file, the file is not included in the directory.

RD (RESTORE DIRECTORY)

OPTIONS:

List of options to control the restore operation. Each option is expressed as a pair of alternatives, for example ALIAS and NOALIAS. You can choose one or the other, but not both. If you do not choose either, RD uses a default. Options specified in the control file override options specified here. The options listed after ORIGINAL OPTIONS are the default options modified by the response to the OPTIONS prompt. Options specified by a control file are not included at this point on the listing.

Option	Purpose
ADD	(Default) If the last component of the file name of a file being restored matches the last component of the file name of a file in the directory, do not replace the file in the directory. Leave the original contents of the directory as is, but add any new files from the backup file.
REPLACE	If the file type and last component of the file name of a file being restored match the file type and last component of the file name of a file in the directory, replace the file in the directory with a copy of the file in the backup file. If the file name being restored does not match any of the file names in the directory, create a new entry in the directory for the file on the backup file, giving it a file name that consists of the pathname of the directory followed by the last component of the file name of the file being restored. Copy everything from the backup file into the directory, replacing any files in the directory whose file names conflict with the new additions.
ALIAS	(Default) Restore aliases.
NOALIAS	Do not restore aliases.
DATE	Restore only files updated later than a specified date. If you choose this option, RD displays additional prompts for the date and time. If the DATE option is used in a control file, additional date and time prompts must be answered. In batch or expert mode they can be answered directly. In interactive mode the DATE option must be included in the response to the OPTIONS prompt in order to answer the additional date and time prompts. If either the last update date or the creation date is after the date specified, the file is copied.
NODATE	(Default) Restore files regardless of their dates.
REWIND	Rewind the tape before performing the restore operation.
NOREWIND	(Default) Do not rewind the tape before performing the restore operation.

RD (RESTORE DIRECTORY)

UNLOAD	Rewind and unload the reel after the restore operation.
NOUNLOAD	(Default) Do not rewind and unload the reel after the restore operation.
CMP	(Default) When restoring key indexed files, compress them to the end of medium.
NOCMP	When restoring key indexed files, do not compress them.
SYSFILES	(Default) Include the following system files and directories in a restore operation on the system disk:

.SCI990	.S$SYSLIB
.S$CDT	.S$ISBTCH
.S$CMDS	.S$ISLIST
.S$EXPMSG	.S$LANG
.S$MSG	.S$PWCS
.SSGU	.S$SHARED
.S$SYSTEM	.S$SHIP
.S$IPL	.S$UTIL

NOSYSFILE	Do not include the system files in the restore operation.
FAST	Restore a program file all at once. If you choose this option, you need to make sure the disk that contains the directory where you restore the file has the same sector size as the disk that contained the directory backed up by BD.
NOFAST	(Default) Restore each task, procedure, program segment, and overlay image individually.

EXECUTION MODE(F,B):

Choice of whether to carry out the command in the foreground or background. If you enter F or FOREGROUND, the command executes in the foreground. If you enter B or BACKGROUND, the command executes in the background. Though the default is FOREGROUND, in a batch stream the RD command always executes in the background.

PATHNAME:

The pathname of the backup file. This prompt appears only when you give a device name of a disk unit (DS01 through DS99) in response to the SEQUENTIAL ACCESS NAME prompt. The backup file has to be a sequential file, but it can span several disks. For details, refer to the special case concerning multiple volume backups.

YEAR:

The two- or four-digit number for a year. This prompt appears only when you choose the DATE option. Your response becomes the year portion of the cutoff date used by the DATE option to determine which files to restore.

RD (RESTORE DIRECTORY)

MONTH:
The one- or two-digit number of a month. This prompt appears only when you choose the DATE option. Your response becomes the month portion of the cutoff date used by the DATE option to determine which files to restore.

DAY:
The one- or two-digit number of a day of the month. This prompt appears only when you choose the DATE option. Your response becomes the day portion of the cutoff date used by the DATE option to determine which files to restore.

HOUR:
The hour on the 24-hour clock. This prompt appears only when you choose the DATE option. Your response becomes the hour portion of the cutoff time used by the DATE option to determine which files to restore. If you skip this prompt, RD assumes 0 (the initial value).

MINUTE:
The minute of the hour. This prompt appears only when you choose the DATE option. Your response becomes the minute portion of the cutoff time used by the DATE option to determine which files to copy. If you skip this prompt, RD assumes 0 (the initial value).

Example:

This example restores and verifies the directory VOL2.DIR1 from the backup file BACKUP82.OCTOBER created in the example for BD. The RD command specifies the same cutoff date as before, 10/01/82.

```
[] RD

RESTORE DIRECTORY
   SEQUENTIAL ACCESS NAME: BACKUP82.OCTOBER
       DIRECTORY PATHNAME: VOL2.DIR1
      CONTROL ACCESS NAME:
      LISTING ACCESS NAME: LP
                  OPTIONS: DATE
      EXECUTION MODE(F,B): BACKGROUND

DATE OF OLDEST FILE
                     YEAR: 82
                    MONTH: 10
                      DAY: 1
                     HOUR: 0
                   MINUTE: 0
```

RD (RESTORE DIRECTORY)

The RD command produces the following listings:

```
RESTORE DIRECTORY        13:55:59 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       BACKUP82.OCTOBER
ORIGINAL DESTINATION: VOL2.DIR1
ORIGINAL OPTIONS:      ADD,ALIAS,DATE,NOREWIND,NOUNLOAD,NOFAST,SYSFILES,CMP
CONTROL FILE:
LIST FILE:              LP


  **        DIRECTORY  VOL2.DIR1
  **   FILE3
  **   FILE2
  **   PROGRAMS - PROGRAM FILE

  **   TASK        ID
  **   PROGA      >03
  **   PROGB      >0E
  **        END OF PROGRAM FILE

  **        END OF DIRECTORY  VOL2.DIR1


ELAPSED TIME = 0 MINUTES   32 SECONDS
SIZE OF INPUT = 1782 ADU'S

****** RESTORE DIRECTORY COMPLETED
```

The following VB command makes sure the restored directory matches the backup file:

```
[] VB

    VERIFY BACKUP
       SEQUENTIAL ACCESS NAME: BACKUP82.OCTOBER
            DIRECTORY PATHNAME: VOL2.DIR1
          CONTROL ACCESS NAME:
          LISTING ACCESS NAME:
                        OPTIONS: DATE
          EXECUTION MODE(F,B): BACKGROUND

    DATE OF OLDEST FILE
                           YEAR: 82
                          MONTH: 10
                            DAY: 1
                           HOUR: 0
                         MINUTE: 0
```

RD (RESTORE DIRECTORY)

The VB command produces the following listings:

```
VERIFY BACKUP            13:57:48 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       BACKUP82.OCTOBER
ORIGINAL DESTINATION: VOL2.DIR1
ORIGINAL OPTIONS:      ALIAS,DATE,NOREWIND,NOUNLOAD,SYSFILES,NOFAST
CONTROL FILE:
LIST FILE:             LP


  **       DIRECTORY  VOL2.DIR1
  **  FILE3
  **  FILE2
  **  PROGRAMS - PROGRAM FILE

  **  TASK       ID
  **  PROGA     >03
  **  PROGB     >0E
  **       END OF PROGRAM FILE

  **       END OF DIRECTORY  VOL2.DIR1


ELAPSED TIME = 0 MINUTES   24 SECONDS
SIZE OF INPUT = 1782 ADU'S

****** VERIFY BACKUP COMPLETED
```

Messages:

ERRORS ENCOUNTERED ON BACKUP MEDIUM-FILE NOT RESTORED
: A disk error occurred during the backup of this file. The file was flagged during the backup operation and cannot be restored (or verified).

ERRORS ENCOUNTERED ON BACKUP MEDIUM-REMAINING TASKS NOT RESTORED
: A disk error occurred during the backup of this program file. The preceding tasks were restored. However, any tasks in this program file that were not in the listing file cannot be restored.

Special Cases

Stacked Directories on Tape

If you include more than one MOVE directive in your control file with BD, you can store the backup copies of more than one directory in a single sequential file or on tape. If you do this, the subsequent RD command must use a control file with corresponding MOVE directives to unstack the directories. Several BD commands can be used to back up directories to a magnetic tape, and these can be restored and verified in turn.

In the following example, a BD command has stacked backup copies of the directory VOL2.DIR1, the file VOL2.DIR2.PROGRAMS, and all files in the directory VOL2.DIR3 except for VOL2.DIR3.HEADER into the backup file on MT01. To do this, the BD command used the control file VOL2.STACKBD with the following directives:

```
MOVE VOL2.DIR1
MOVE VOL2.DIR2.PROGRAMS
MOVE VOL2.DIR3
EXCL HEADER
END
```

To unstack the backup copies, the RD command requires the control file VOL2.STACKRD with the following directives:

```
MOVE ,VOL2.DIR1
MOVE ,VOL2.DIR2.PROGRAMS
MOVE ,VOL2.DIR3
EXCL HEADER
END
```

The following RD command unstacks the directories from the backup file, restoring their original contents:

```
[] RD

RESTORE DIRECTORY
   SEQUENTIAL ACCESS NAME: MT01
        DIRECTORY PATHNAME: VOL2.DIR1
      CONTROL ACCESS NAME: VOL2.STACKRD
      LISTING ACCESS NAME: LP
                  OPTIONS: REPLACE
      EXECUTION MODE(F,B): BACKGROUND
```

The presence of the exclude directive in the control file leads to a spurious warning for the file excluded. Similarly, the inadvertent omission of the exclude directive from the control file does not produce a warning. This leaves open the possibility of that file being incorrectly included in the restored directory without the appropriate warning.

RD (RESTORE DIRECTORY)

The restore operations return the following listings:

```
RESTORE DIRECTORY        14:03:04 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       MT01
ORIGINAL DESTINATION:
ORIGINAL OPTIONS:  REPLACE,ALIAS,NODATE,NOREWIND,NOUNLOAD,NOFAST,SYSFILES
CONTROL FILE:          VOL2.STACKRD
LIST FILE:             LP

MOVE ,VOL2.DIR1

  **       DIRECTORY  VOL2.DIR1
  **  FILE1      WAS REPLACED
  **  FILE3      WAS REPLACED
  **  FILE2      WAS REPLACED
  **  PROGRAMS - PROGRAM FILE

  **  TASK        ID
  **  PROGA      >03
  **  PROGB      >0E
  **       END OF PROGRAM FILE

  **       END OF DIRECTORY  VOL2.DIR1

MOVE ,VOL2.DIR2.PROGRAMS

  **       DIRECTORY  VOL2.DIR2
  **  PROGRAMS - PROGRAM FILE

  **  TASK        ID
  **  PROGA      >03
  **  PROGB      >0E
  **       END OF PROGRAM FILE

  **       END OF DIRECTORY  VOL2.DIR2

MOVE ,VOL2.DIR3
EXCL HEADER

  **       DIRECTORY  VOL2.DIR3
  **  FILEC      WAS REPLACED
  **  FILEA      WAS REPLACED
  **  FILEB      WAS REPLACED

  **       END OF DIRECTORY  VOL2.DIR3

  **  HEADER     INCLUDE/EXCLUDE FILE NOT FOUND
END

ELAPSED TIME = 3 MINUTES   16 SECONDS
SIZE OF INPUT = 4556 ADU'S

****** RESTORE DIRECTORY COMPLETED
```

You can use a VB command with a similar control file to make sure the backup files match the original directories.

Multiple Volume Backups

Backup files can span more than one disk or magnetic tape, as explained in the special cases for BD. When you restore the files to a directory and verify the restoration, you need to see that the RD or VB command reads the backup disks or tapes in the order they were written. A multi-volume restore operation cannot be executed from a batch stream.

The following commands demonstrate how to perform a multiple volume restore and verify operation. RD restores all files in VOL2.DIR1 except for VOL2.DIR1.PROGRAMS using a control file entered from the station. As in the example given in the special cases for BD, the backup file BACKUP82.OCTOBER fills three diskettes and again the only diskette drives available are DS03 and DS04. The operation begins with volume 1 of BACKUP82 mounted on DS03, volume 2 on DS04, and volume 3 held for later.

```
[] RD

RESTORE DIRECTORY
   SEQUENTIAL ACCESS NAME: DS03,DS04
        DIRECTORY PATHNAME: VOL2.DIR1
      CONTROL ACCESS NAME: ME
      LISTING ACCESS NAME: LP
                  OPTIONS:
      EXECUTION MODE(F,B): BACKGROUND

   PATHNAME OF BACKUP FILE
                 PATHNAME: BACKUP82.OCTOBER

EXCLUDE PROGRAMS
END
```

After it restores the files from the first two diskettes, RD displays the following message at the station:

```
END OF MEDIA  MOUNT VOLUME 3; TYPE $ TO QUIT, Y TO CONTINUE
```

RD (RESTORE DIRECTORY)

After it receives a Y response, indicating that the third volume of BACKUP82 is mounted in DS03, the RD command continues executing, eventually producing the following listings:

```
  RESTORE DIRECTORY        14:09:29 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       DS03,DS04
ORIGINAL DESTINATION:  VOL2.DIR1
ORIGINAL OPTIONS:      ADD,ALIAS,NODATE,NOREWIND,NOUNLOAD,NOFAST,SYSFILES,CMP
CONTROL FILE:          ST03
LIST FILE:             LP

EXCLUDE PROGRAMS

  **        DIRECTORY  VOL2.DIR1
  **   FILE1
  **   FILE3
  **   FILE2
  **        END OF DIRECTORY  VOL2.DIR1

END

ELAPSED TIME = 0 MINUTES   24 SECONDS
SIZE OF INPUT = 123 ADU'S

****** RESTORE DIRECTORY COMPLETED
```

The following VB command makes sure the contents of the multiple volume backup file match the restored directory. Since it is not necessary to exclude the file VOL2.DIR1.PROGRAMS from the verification, the VB command requires no control file.

```
[] VB

VERIFY BACKUP
   SEQUENTIAL ACCESS NAME: DS03,DS04
       DIRECTORY PATHNAME: VOL2.DIR1
      CONTROL ACCESS NAME:
      LISTING ACCESS NAME: LP
                  OPTIONS:
      EXECUTION MODE(F,B): BACKGROUND

   PATHNAME OF BACKUP FILE
                 PATHNAME: BACKUP82.OCTOBER
```

After it verifies the files from the first two diskettes, VB displays the following message at the station:

```
END OF MEDIA  MOUNT VOLUME 3; TYPE $$ TO QUIT, Y TO CONTINUE
```

RD (RESTORE DIRECTORY)

The VB command produces the following listings:

```
  VERIFY BACKUP            14:11:06 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       DS03,DS04
ORIGINAL DESTINATION:  VOL2.DIR1
ORIGINAL OPTIONS:      ALIAS,NODATE,NOREWIND,NOUNLOAD,SYSFILES,NOFAST
CONTROL FILE:
LIST FILE:             LP


  **        DIRECTORY  VOL2.DIR1
  **   FILE1
  **   FILE3
  **   FILE2
  **   PROGRAMS  FILE NOT FOUND IN DESTINATION
  **        END OF DIRECTORY  VOL2.DIR1


ELAPSED TIME = 0 MINUTES   11 SECONDS
SIZE OF INPUT = 123 ADU'S
1 WARNINGS ISSUED

****** VERIFY BACKUP COMPLETED
```

Notes:

When RD is used in a file security system, you must have read access rights to the file specified for SEQUENTIAL ACCESS NAME. If you are replacing any existing files in the directory specified for DIRECTORY PATHNAME, you must also have delete access rights to the files being replaced.

Related Commands:

BD (Backup Directory)
VB (Verify Backup)

RE (RECOVER EDIT)

4.175 RE (RECOVER EDIT)

The RE command allows you to recover a major portion of a text edit that was in progress when a system crash or power failure occurred, depending upon the nature of the system failure. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the RE command and the following prompts:

```
RECOVER EDIT
    OUTPUT FILE ACCESS NAME:  [site:]filename@
                   REPLACE?:  YES/NO                (NO)
        MOD LIST ACCESS NAME:  [site:][pathname]@
```

4.176 RGL (RELEASE GLOBAL LUNO)

Prompts:

```
RELEASE GLOBAL LUNO
            LUNO:  integer
```

Purpose:

The RGL command releases a global LUNO assignment. Global LUNOs actively performing I/O operations cannot be released.

Prompt Details:

LUNO:
The logical unit number to be released.

Example:

In the following example, the global LUNO >25 was previously assigned to the line printer, LP01. The global LUNO assignment is released through the RGL command as shown below:

```
[] RGL

RELEASE GLOBAL LUNO
            LUNO: >25
```

Related Commands:

AGL (Assign Global LUNO)

RJ (RESUME JOB)

4.177 RJ (RESUME JOB)

Prompts:

```
RESUME JOB
        JOB NAME OR ID:   {alphanumeric/integer}  (*)
```

Purpose:

The RJ command resumes execution of a job that was previously halted by the Halt Job (HJ) command.

Prompt Details:

JOB NAME OR ID:
The job name that you specify during job creation or the job ID returned by the Execute Job (XJ) command.

Example:

In the following example, a user resumes a job named REPORT, previously halted, by executing the RJ command as shown below:

```
[] RJ

RESUME JOB
        JOB NAME OR ID: REPORT
```

Assumptions:

The job being resumed is under the same user ID as the job in which the RJ command is entered, or the command is entered by the system operator.

Related Commands:

HJ	(Halt Job)
KJ	(Kill Job)
LJ	(List Jobs)
MJP	(Modify Job Priority)
SJS	(Show Job Status for < user ID>)
XBJ	(Execute Batch Job)
XJ	(Execute Job)

4.178 RL (RELEASE LUNO)

Prompts:

```
RELEASE LUNO
            LUNO:  integer
```

Purpose:

The RL command releases a job-local LUNO assignment from a device or file. You cannot release a LUNO in use.

Prompt Details:

LUNO:
The job-local LUNO to be released.

Example:

In the following example, job-local LUNO > 25, which is assigned to a file, is released through the RL command as shown below:

```
[] RL

RELEASE LUNO
            LUNO: >25
```

Related Commands:

AL	(Assign LUNO)
RAL	(Release All Logical Units)

RLN (RELEASE LOGICAL NAME)

4.179 RLN (RELEASE LOGICAL NAME)

Prompts:

```
RELEASE LOGICAL NAME
              LOGICAL NAME:  alphanumeric
              GLOBAL NAME?:  YES/NO                (NO)
```

Purpose:

The RLN command deletes the specified logical name. In the case of a logical name specified to represent a job-local temporary file, the file will be deleted after all LUNOs are released.

Prompt Details:

LOGICAL NAME:
A one- to eight-character name.

GLOBAL NAME?:
If the response is YES, this indicates that the logical name is available to all users. If the response is NO, the logical name is not available to all users.

Example:

In the following example, a logical name PRINT has been defined but is no longer needed. The RLN command releases the logical name definition as shown below:

```
[] RLN

RELEASE LOGICAL NAME
        LOGICAL NAME: PRINT
        GLOBAL NAME?: NO
```

Notes:

In order to remove the global logical name from the permanent list, a Snapshot Global Name Definitions (SGND) must follow the RLN command.

Related Commands:

ALN	(Assign Logical Name)
LLN	(List Logical Names)
SGND	(Snapshot Global Name Definitions)

RO (RESUME OUTPUT AT DEVICE)

4.180 RO (RESUME OUTPUT AT DEVICE)

Prompts:

```
RESUME OUTPUT AT DEVICE
        DEVICE OR CLASS NAME:  [site:]{devicename/characters(s)}@  (*)
    PAGE SKIP(-32767..32767):  [integer]                           (0)
```

Purpose:

The RO command resumes the queued output that has been temporarily halted. Before resuming the output, you can specify how many pages to back up or go forward from the point where output was halted.

Prompt Details:

DEVICE OR CLASS NAME:
The device or class name for which the queued output is to resume.

PAGE SKIP(-32767..32767):
An integer value specifying how many pages to skip backward or forward before output is resumed. The maximum number of pages to back up or go forward is -32767 and 32767, respectively. If a null response is entered, output resumes at the point where it was halted.

Example:

In the following example, output that was halted at a line printer LP01 is resumed through the RO command:

```
[] RO

RESUME OUTPUT AT DEVICE
       DEVICE OR CLASS NAME: LP01
  PAGE SKIP(-32767..32767): 0
```

Related Commands:

HO	(Halt Output at Device)
KO	(Kill Output at Device)
MO	(Modify Output at Device)
SOS	(Show Output Status)

ROM (RECEIVE OPERATOR MESSAGES)

4.181 ROM (RECEIVE OPERATOR MESSAGES)

Prompts:

```
RECEIVE OPERATOR MESSAGES
        MESSAGES (MY/ALL): {MY/ALL}
```

Purpose:

The ROM command causes your terminal to receive operator messages determined by your user ID. If there is no system operator, you can respond to the messages; otherwise, only the operator may respond.

You can continue to use the terminal for other purposes after entering the ROM command.

Prompt Details:

MESSAGES (MY/ALL):
If you enter MY as the response to the MESSAGES (MY/ALL) prompt, only messages originated by or for you are relayed. If you enter ALL in response to MESSAGES (MY/ALL), all operator messages are sent to your terminal.

Example:

In the following example, all operator messages are to be relayed to a user's terminal by executing the ROM command:

```
[] ROM

RECEIVE OPERATOR MESSAGES
     MESSAGES (MY/ALL): ALL
```

Special Cases:

If you issued the command from the operator console (the terminal where the Execute Operator Interface (XOI) command was entered), the MY response to the MESSAGES (MY/ALL) prompt has the same effect as entering ALL.

Notes:

If you enter the ROM command and another user enters the XOI command, you will be shown operator messages without request ID's. This indicates that you cannot respond to them.

Related Commands:

COM	(Create Operator Message)
KOM	(Kill Operator Messages)
KOR	(Kill Operator Interface Request)
LOM	(List Operator Messages)
QOI	(Quit Operator Interface)
ROR	(Respond to Operator Interface Request)
XOI	(Execute Operator Interface)

ROR (RESPOND TO OPERATOR INTERFACE REQUEST)

4.182 ROR (RESPOND TO OPERATOR INTERFACE REQUEST)

Prompts:

RESPOND TO OPERATOR INTERFACE REQUEST
REQUEST ID: integer

Purpose:

The ROR command responds to a request which the operator has fulfilled. The operator enters the request ID associated with the request to specify for which message the response is intended. This command is only valid interactively.

When you enter the request ID in response to the REQUEST ID prompt, the text of the request message and other prompts pertinent to the request are displayed at the terminal if a response is required. The original values specified by the requester become the default values for the prompts.

If the specified request does not require a response, a message (instead of the request message text and prompts) is displayed noting that the request has been acknowledged.

Pressing the F4 terminal function key is also equivalent to executing the ROR command if the Execute Operator Interface (XOI) command has been entered and the XOI task is active in the foreground.

Prompt Details:

REQUEST ID:
The request ID associated with the request which is to be responded to.

If you press the Command key when the second set of prompts is displayed, the ROR command terminates and leaves the request pending.

ROR (RESPOND TO OPERATOR INTERFACE REQUEST)

Example:

In the following example, a message and prompts have been created by a user requesting that tape number 444555 be mounted on tape drive MT02. The request has an ID of 123. The request displays on the operator's terminal as follows:

```
OR 123* FROM USER1234 AT 12:34-MOUNT TAPE 444555 ON MT02
```

The operator could complete the request and respond to any prompts which accompany the request by executing the ROR command:

```
[] ROR

RESPOND TO OPERATOR INTERFACE REQUEST
    REQUEST ID: 123

MOUNT TAPE 444555 ON DRIVE MT02

        TAPE: 444555
       DRIVE: MT02
```

If the operator request ID is followed by an asterisk, but the request does not include prompts for TAPE and DRIVE, the following message is displayed:

```
MOUNT TAPE 444555 ON DRIVE MT02

      GRANT REQUEST?:
```

Respond YES to return a positive reply to the requestor. Respond NO to leave the request pending.

If the operator request ID is not followed by an asterisk, the mount request message and the TAPE and DRIVE prompts are not displayed. However, the following message is displayed acknowledging the ROR request:

```
I UTILITY-0531 REQUEST 123 ACKNOWLEDGED
```

Assumptions:

You previously entered the XOI command or you previously entered the Receive Operator Messages (ROM) command, and no other user has entered the XOI command. Additionally, the request ID entered corresponds to a pending operator request, and no other user is in the process of responding to that request.

ROR (RESPOND TO OPERATOR INTERFACE REQUEST)

Notes:

If the operator request ID is followed by an asterisk (*), the associated task is suspended until you enter a response for the request. If the request ID is not followed by an asterisk, the associated task is not suspended. However, the operator must enter the KOR or ROR command to complete the request.

Operator requests are created by the Create Operator Message (COM) command, the spooler subsystem, and by programs that use the OI$ routines described in the *DNOS Systems Programmer's Guide.*

Related Commands:

COM	(Create Operator Message)
KOM	(Kill Operator Messages)
KOR	(Kill Operator Interface Request)
LOM	(List Operator Messages)
QOI	(Quit Operator Interface)
ROM	(Receive Operator Messages)
XOI	(Execute Operator Interface)

RS (REPLACE STRING)

4.183 RS (REPLACE STRING)

After you activate the Text Editor, the RS command replaces a specified number of occurrences of a string of characters with a new character string. Refer to the *DNOS Text Editor Reference Manual* for further information on the RS command and the following prompts:

```
REPLACE STRING
    NUMBER OF OCCURRENCES:  integer          (1)
             START COLUMN:  integer          (*)
               END COLUMN:  integer          (*)
                   STRING:  [character(s)]   (*)
                   CHANGE:  [character(s)]
```

4.184 RST (RESUME SIMULATED TASK)

Purpose:

The RST command resumes task simulation following a breakpoint, a simulated breakpoint, or simulation of a specified number of instructions (time out). The last entered values for the FOR and TO prompts of the Simulate Task (ST) command are used as the RST limits. Upon reaching a terminating condition (breakpoint, simulated breakpoint, time out, or the value specified for the TO prompt), a debug panel and termination reason are displayed. Press the F3 function key to continue simulation or the Command key to terminate the simulation and return SCI to the command mode.

Example:

In the following example, the RST command resumes a simulated task which has encountered a terminating condition, using the values entered for the FOR and TO prompts of the ST command.

```
[] RST

RESUME SIMULATED TASK
```

Special Cases:

If you have not executed the ST command previously, the default value of FOR is one. One instruction is executed before the task is halted.

Assumptions:

The task has been placed in controlled mode by the execution of the Execute in Debug Mode (XD) command, and has reached a terminating condition.

Messages:

Upon reaching a terminating condition, the debug panel and a termination reason are displayed on your terminal.

Related Commands:

ST	(Simulate Task)
XD	(Execute in Debug Mode)

RT (RESUME TASK)

4.185 RT (RESUME TASK)

Prompts:

RESUME TASK

RUN ID: integer (*)

Purpose:

The RT command activates a task at the point at which it was suspended. The specified task must be unconditionally suspended when you execute this command or an error is indicated. You must use either the Delete Breakpoint (DB) and the RT command, Delete and Proceed from Breakpoint (DPB) or Proceed from Breakpoint (PB) command to restart a task halted at a breakpoint. However, use the RT command, rather than the Activate Task (AT) command, to reactivate a task halted by the Halt Task (HT) command.

Prompt Details:

RUN ID:
The response to this prompt must be a valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

Example:

In the following example, the RT command will reactivate a suspended task with a run ID of >7.

```
[] RT

RESUME TASK
            RUN ID: >7
```

Messages:

If the task to be activated is not unconditionally suspended, an error message is displayed on your terminal.

Related Commands:

AT	(Activate Task)
HT	(Halt Task)
XD	(Execute in Debug Mode)

RVI (RECOVER VOLUME INFORMATION)

4.186 RVI (RECOVER VOLUME INFORMATION)

Prompts:

```
RECOVER VOLUME INFORMATION
                UNIT NAME:  devicename@
```

Purpose:

The RVI command restores the volume information to track 0, sector 0, of the disk from a copy that was stored on track 1 when the disk was initialized.

If the volume information on track 0, sector 0, has been destroyed, you cannot install the disk and access its file. Also, if the track 1 copy of the volume information has been destroyed, you cannot recover the disk through the RVI command.

Execute the RVI command to a disk if a flashing crash code of >1, >6, or >B occurs while performing an initial program load (IPL) with that disk. If an error code of >10, >11, >12, >17, >1B, or >1C occurs while installing a disk with the Install Volume (IV) command, load the disk as a secondary disk and recover it by executing the RVI command.

NOTE

The RVI command can be executed on either an installed or an uninstalled volume. However, if RVI is executed on an uninstalled volume, the volume remains uninstalled after completion of the RVI command.

RVI flags all allocatable disk units (ADUs) on the disk as allocated. Therefore, it is necessary to copy the data files on the disk to another media after you perform RVI by using the Copy Directory (CD) or Backup Directory (BD) command. After you have copied all data files, reinitialize the recovered disk using the Initialize New Volume (INV) command in order to use that disk in the future. As a safeguard, do not reinitialize the recovered disk immediately, as it may be necessary to recopy files from that disk.

NOTE

After the RVI command has executed, you must install the disk using the Install Volume (IV) command to access any data files on that disk.

After you perform the RVI command, prior to copying the data files, you can execute the Check Disk for Consistency (CKD) command on the recovered disk. As a result, you know beforehand if there are any bad directories or files that are not copied when the CD or BD command is issued. Because RVI flags all ADUs as allocated, the CKD dump of the disk's bit maps contains lines showing the range of ADUs that are allocated but not used.

RVI (RECOVER VOLUME INFORMATION)

Prompt Details:

UNIT NAME:
The device name of the disk, other than the system disk, containing the disk to be restored.

Example:

In the following example, the volume information is restored and the disk is installed through the RVI and IV commands, respectively, as follows:

```
[] RVI

RECOVER VOLUME INFORMATION
          UNIT NAME:  DS04

[ ] IV

INSTALL VOLUME
          UNIT NAME:  DS04
        VOLUME NAME:  VOL1
```

Notes:

The RVI command cannot be performed on the system disk. If you must recover the system disk, mount that disk in a secondary disk drive.

Related Commands:

CKD (Check Disk for Consistency)

4.187 RWL (REWIND LUNO)

Prompts:

REWIND LUNO
LUNO: integer

Purpose:

The RWL command rewinds a sequential file or a rewindable device, such as a magnetic tape. Rewinding a file or device moves the current data-access position to the first record of that file or device. The operating system ignores entry of an RWL command if the device or file is not rewindable.

Prompt Details:

LUNO:
The LUNO assigned to the file or device that is to be rewound.

Example:

In the following example, LUNO > 27 has been assigned to a sequential file and the file must be rewound before any commands can be executed to access the file. The file is rewound through the RWL command as shown below:

```
[] RWL

 REWIND LUNO
              LUNO: >27
```

Notes:

The operations performed by RWL include an open rewind call and a close call to the file or device.

Related Commands:

BL (Backspace LUNO)
FL (Forward Space LUNO)

SAD (SHOW ABSOLUTE DISK)

4.188 SAD (SHOW ABSOLUTE DISK)

Prompts:

```
SHOW ABSOLUTE DISK
                    DISK UNIT:  devicename@          (*)
                         TRACK:  integer exp         (*)
                        SECTOR:  integer exp         (*)
                    FIRST WORD:  integer exp         (*)
              NUMBER OF WORDS:  [integer exp]
          OUTPUT ACCESS NAME:  [site:][pathname]@    (*)
```

Purpose:

The SAD command prints the contents of a specified absolute address on a disk. The contents of 16 bytes are printed per line, with the address of the first byte printed as the first entry on the line. The contents of each pair of bytes are shown as four hexadecimal digits. At the right end of the line, the contents are printed as ASCII characters. The bytes that contain values that correspond to printable ASCII characters are translated and printed as ASCII characters; nonprinting ASCII characters are printed as periods.

Prompt Details:

DISK UNIT:
The device name assigned to the disk during either system generation or system configuration.

TRACK:
The integer value that is the starting track address from which to begin printing the contents of the disk.

SECTOR:
The integer value that is the starting sector address, within the specified disk track, from which to begin printing the contents of the disk.

FIRST WORD:
The integer value that is the word offset, within the specified disk sector, from which to begin printing the contents of the disk.

NUMBER OF WORDS:
The integer value that is the number of words of the specified sector to print. The default value is the disk sector size.

OUTPUT ACCESS NAME:
The device name or file name of a device or file where the contents of the specified absolute disk address are to be printed. The default value is the terminal local file.

SAD (SHOW ABSOLUTE DISK)

Example:

In the following example, the SAD command prints the contents of five words on the disk in disk device DS02, beginning at track 0, sector 0, word 0. The results are printed to the terminal local file where the SAD command is executed.

```
[] SAD

SHOW ABSOLUTE DISK
               DISK UNIT: DS02
                   TRACK: 0
                  SECTOR: 0
              FIRST WORD: 0
         NUMBER OF WORDS: 5
     OUTPUT ACCESS NAME:

TRACK 0000 SECTOR 00  RECORD LENGTH 0120 BYTES (01 SECTORS).
0000  4D41 5931 3030 2020 2610                    MA Y1 00    &.
```

Related Comamnds:

MAD (Modify Absolute Disk)

SADU (SHOW ALLOCATABLE DISK UNIT)

4.189 SADU (SHOW ALLOCATABLE DISK UNIT)

Prompts:

```
SHOW ALLOCATABLE DISK UNIT
                DISK UNIT:  devicename@          (*)
              ADU NUMBER:   integer exp          (*)
           SECTOR OFFSET:   integer exp          (*)
              FIRST WORD:   integer exp          (*)
        NUMBER OF WORDS:    [integer exp]
      OUTPUT ACCESS NAME:   [site:][pathname]@   (*)
```

Purpose:

The SADU command outputs the contents of all or a portion of a sector specified by ADU number, sector offset within the ADU, and word offset within the sector.

All disks on the operating system are addressed in allocatable disk units (ADUs). The maximum number of ADUs on a disk is 65,535. Therefore, if a disk contains more than 65,535 sectors, multiple sectors are used to form one ADU.

Prompt Details:

DISK UNIT:
The device name assigned to the disk during system generation or system configuration.

ADU NUMBER:
The integer value that is the ADU with contents to be listed.

SECTOR OFFSET:
The integer value that is the sector of the ADU with contents to be listed.

FIRST WORD:
The integer value that is the word offset, within the specified sector, from which to begin listing the contents of the ADU.

NUMBER OF WORDS:
The integer value that is the number of words of the specified sector to list. The default value is the disk ADU size.

OUTPUT ACCESS NAME:
The device or file name where the contents of the specified ADU are to be listed. The default value is the terminal local file.

SADU (SHOW ALLOCATABLE DISK UNIT)

Example:

In the following example, the SADU command lists the contents of four words for ADU number 143 on the disk in disk device DS02, beginning at sector zero, word zero. The results are listed on the terminal local file where the SADU command is executed.

```
[] SADU

SHOW ALLOCATABLE DISK UNIT
                DISK UNIT: DS02
              ADU NUMBER: 143
            SECTOR OFFSET: 0
               FIRST WORD: 0
           NUMBER OF WORDS: 4
      OUTPUT ACCESS NAME:

ADU 008F  SECTOR 00  RECORD LENGTH 0120 BYTES (01 SECTORS).
0000  1102 0221 070D DDC1                          .. .! .. ..
```

Related Commands:

MADU (Modify Allocatable Disk Unit)

SBS (SHOW BACKGROUND STATUS)

4.190 SBS (SHOW BACKGROUND STATUS)

Purpose:

The SBS command views the status of a program which you activated from your terminal and is currently executing in background mode. Status messages, which reflect task state codes, are produced by the SBS command when the background is active. The format of the messages is as follows:

```
I STATUS - XXXX message text
```

where:

I indicates an informative message and XXXX is a message number associated with the message text.

See the section Status Messages and Codes in the *DNOS Messages and Codes Reference Manual* for details about the status.

Use the SBS command to display the background terminal local file (TLF). The TLF is displayed by SBS when the background task has terminated and the TLF has been written in by the background task. (When SCI is in batch mode, the background TLF is written to the listing file or device when each background command completes.)

The background TLF is opened with an open extend opcode so that output written to the TLF is not lost if you forget to do an SBS between two background task executions. The WAIT command automatically displays the background TLF if it exists when the background task completes.

A nondisplayed prompt, STATE SYNONYM, is contained in the SBS command procedure. If you specify a name for this prompt in expert mode or within a procedure the task state of the background task is assigned to this name as a synonym value. If there is no background activity in progress, the name is assigned the value of >FF. Use the STATE SYNONYM prompt in command procedures to check if there is any background activity in progress. (If a background task is started while background activity is in progress at a station, SCI immediately terminates the current procedure and other command procedures active at the user's station and displays a message indicating that a background task is in progress.) The following example specifies a value for the STATE SYNONYM prompt when entering the SBS command in expert mode:

```
[ ] SBS STATE SYNONYM = A
```

The name of A will be assigned the hexadecimal value of the task state of the background task (for example, >05 if the state of the background task is a time delay).

SBS (SHOW BACKGROUND STATUS)

Example:

In the following example, the SBS command checks the status of background SCI.

```
[] SBS

SHOW BACKGROUND STATUS
I UTILITY-0190 THERE IS NO BACKGROUND ACTIVITY AT THIS TERMINAL
```

Messages:

Status messages are displayed at the terminal where the SBS command is executed, reflecting the state of a task or indicating that there is background activity at that terminal.

Related Commands:

KBT	(Kill Background Task)
WAIT	(Wait for Background Task to Complete)
XT	(Execute Task)

SCAG (SET CREATION ACCESS GROUP)

4.191 SCAG (SET CREATION ACCESS GROUP)

Prompts:

```
SET CREATION ACCESS GROUP
                    PASSCODE:   – character(s)
          ACCESS GROUP NAME:    alphanumeric
```

Purpose:

The SCAG command permits you to designate an access group to be given full access control to any files you create. Your user ID must be a member of the specified access group. This command takes effect the next time your user ID is used for logging on or for starting another job. If this command has never been used for your user ID, the creation access group for your user ID is PUBLIC.

Prompt Details:

PASSCODE:
Enter your assigned log-on passcode.

ACCESS GROUP NAME:
Specify the name of the access group to be designated as your file creation access group.

Example:

In the following example, a user with the passcode P89J55 is a member of the access group named CLERKS. The SCAG command is used to designate the access group named CLERKS to automatically be granted full access control to any files this user may create. (The passcode is not echoed to the user's terminal.)

```
[] SCAG

SET CREATION ACCESS GROUP
                    PASSCODE:
          ACCESS GROUP NAME: CLERKS
```

Related Commands:

CAG (Create Access Group)

4.192 SCC (SHOW COUNTRY CODE)

Prompts:

SHOW COUNTRY CODE
LISTING ACCESS NAME: [site:][pathname]@

Purpose:

The SCC command lists the country code selected when you generate the operating system. Refer to the Modify Country Code (MCC) command for a list of valid country codes.

Prompt Details:

LISTING ACCESS NAME:
The device or file name to list the current country code. If you enter a null response, the country code is listed to the terminal local file.

Example:

In the following example, the SCC command lists the country code USA (United States of America) to the terminal local file.

```
[] SCC

SHOW COUNTRY CODE
    LISTING ACCESS NAME:

COUNTRY = USA
```

Notes:

SCC displays the country code of the running system. If the Execute System Configuration Utility (XSCU) command is executed, SCC displays the country code of the disk image specified in the XSCU command.

Related Commands:

MCC (Modify Country Code)
XSCU (Execute System Configuration Utility)

SCDT (SHOW COMMAND DEFINITION TABLE)

4.193 SCDT (SHOW COMMAND DEFINITION TABLE)

Prompts:

```
SHOW COMMAND DEFINITION TABLE
                VOLUME NAME:  [site:][alphanumeric]@
                SYSTEM NAME:  alphanumeric
DEVICE TYPE/CDT NUMBER:  {911/940/931/733/TPD/SD/VT/integer}(911)
          LISTING ACCESS NAME:  [site:][pathname]@
```

Purpose:

The SCDT command displays the Command Definition Entries (CDEs) listed in the Command Definition Table (CDT). These entries allow tasks to be bid through a specific key sequence; that is, you press the Attention key followed by the bid character associated with the task.

Prompt Details:

VOLUME NAME:
The device name or disk name on which the system resides that contains the CDT to be displayed. The default is the system disk.

SYSTEM NAME:
The name of the system image containing the CDT that is to be displayed.

DEVICE TYPE/CDT NUMBER:
The device type or CDT number indicating the CDT that is to be displayed. Each type of keyboard device is associated with a specific default CDT. You can enter a hexadecimal number (0 through 24) associated with a particular CDT or enter one of the following device types:

911	
940	
931	
733	
TPD	Teleprinter devices
SD	Special devices defined by the user
VT	Virtual terminal

If you enter a device type as the response, the CDT associated with the device type is used.

LISTING ACCESS NAME:
The pathname of the file or device to which the display is written. A file is created if it does not previously exist. The default is the terminal local file.

SCDT (SHOW COMMAND DEFINITION TABLE)

Example:

In the following example, CDT number > A is displayed to the terminal local file. (CDT > A is the default for the 911.)

```
[] SCDT

SHOW COMMAND DEFINITION TABLE
             VOLUME NAME: HISDISK
             SYSTEM NAME: SYSTEM1
 DEVICE TYPE/CDT NUMBER: 911
    LISTING ACCESS NAME:

                CDES FOR COMMAND DEFINITION TABLE NUMBER:  >A

2100 FF0A FF01 0000 0000 2020 2020 2020 2020      !. .. .. .. ..
1880 FF0A FF16 0000 0000 2020 2020 2020 2020      .. .. .. .. ..
4E30 FF0A FF91 0600 0100 4E45 5457 4F52 4B20      NO .. .. .. .. NE TW OR K
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
0000 0000 0000 0000 0000 0000 0000 0000 0000      .. .. .. .. .. .. .. .. ..
```

Related Commands:

MCDT (Modify Command Definition Table)

SCS (SHOW CHANNEL STATUS)

4.194 SCS (SHOW CHANNEL STATUS)

Prompts:

SHOW CHANNEL STATUS
CHANNEL PATHNAME: [site:]pathname@

Purpose:

The SCS command displays information about a channel currently in use and available to your job, including channel owner, type of channel, scope of channel, maximum message length, shared or not shared access, number of current assigns and number of current opens. For task local channels, the display also indicates the number of copies of that channel that are currently in use by your job. You can examine only global channels and channels in use by your job.

Upon a successful completion of the SCS command, information is displayed on your terminal.

Upon an unsuccessful completion of this command, a message describing the error is returned to your terminal.

Prompt Details:

CHANNEL PATHNAME:
The pathname of the channel to be displayed.

SCS (SHOW CHANNEL STATUS)

Example:

In the following example, the user wants to display the status information associated with a task local channel named .KC0017.REPORT. The information can be displayed by executing the SCS command as follows:

```
[] SCS

SHOW CHANNEL STATUS
     CHANNEL PATHNAME: .KC0017.REPORT

CHANNEL STATUS REPORT FOR .S$OPER

          CHANNEL PATHNAME: .KC0017.REPORT
  OWNER TASK PROGRAM FILE: .KC0017.COMPILE
             OWNER TASK ID: 03B
              CHANNEL TYPE: MASTER/SLAVE
             RESOURCE TYPE: CHAN
           PROCESS ASSIGNS: YES
             CHANNEL SCOPE: TASK
   MAXIMUM MESSAGE LENGTH: 100
    SHARED CHANNEL ACCESS: NO
CURRENT NUMBER OF ASSIGNS: 4
  CURRENT NUMBER OF OPENS: 2
NUMBER IN USE BY THIS JOB: 4
```

Descriptions of these fields can be found with the Create IPC Channel (CIC) command.

Assumption:

The channel specified by the pathname is in use. If it is not in use, you can use the List Directory (LD) command to examine channel characteristics by specifying the directory in which the channel resides.

Related Commands:

CIC	(Create IPC Channel)
DIC	(Delete IPC Channel)
LD	(List Directory)
SIS	(Show I/O Status)

SD (SCAN DISK)

4.195 SD (SCAN DISK)

Prompts:

```
SCAN DISK
                    DRIVE NAME:  devicename@
            LISTING FILE NAME:   [site:][filename]@
                      MONITOR:   YES/NO                (NO)
```

Purpose:

The SD command reads an entire disk, checking for sectors which cannot be read by the disk drive. Hard errors (scratches or other physical impairments) and soft errors (data errors) are detected and presented in a report. The SD processor scans the disk, then produces a report of the locations at which an error status was returned. The report lists the error locations in terms of track and sector of the disk and the allocation units (ADUs) that the software uses. If a file uses the error location, the report also gives the pathname of that file. The report may also be useful in determining if any files need to be reconstructed. Bad track information on the disk is not changed, and dual allocation problems are not analyzed. You can invoke the SD command for a disk whether or not the volume is installed.

Prompt Details:

DRIVE NAME:
Enter the device name of the disk unit in which the disk you want to scan is mounted.

LISTING FILE NAME:
Enter the access name of the file to which the bad sector information is written. The default is your terminal local file.

MONITOR:
If Scan Disk is executed in foreground mode, a YES for this prompt allows you to monitor the progress of SD on your terminal.

Example:

The following example shows a disk scan performed on the volume mounted in disk unit DS02. The output is written to the file .REPORT. The disk is a diskette that has four bad sectors.

```
[] SD

SCAN DISK
                DRIVE NAME: DS02
          LISTING FILE NAME: .REPORT
                    MONITOR: NO

SCAN DISK      VOLUME NAME:FLOPPY          05/11/83     11:30:10

 BAD TRACK SUMMARY

  PHYSICAL DATA                    ALLOCATION  DATA

   SECTOR    HEAD      CYL        ADU  SECTOR    FILE NAME
        0       1       24       1907       0    DS02.FILE1
        0       1       26       2404       0    DS02.FILE1
        0       1       27       2453       0    DS02.PROGA
        0       1       29       2609       0    NO FILE ASSOCIATED
```

Notes:

Errors that occur during the use of SD may appear as FORTRAN – 78 error messages. If you do not have FORTRAN messages on your system you may see messages in abbreviated format as follows:

FORT 78RT — INTERNAL CODE > 0038

FORT 78RT — INTERNAL CODE > 0040

The first message indicates that a Fortran STOP statement was executed and the second indicates normal completion. Consult a *DNOS FORTRAN Programmer's Guide* for details.

SD (SCAN DISK)

The AUTOCORRECT prompt is a hidden prompt that you can use in either the batch or the expert mode. If the errors are soft errors, you may select the AUTOCORRECT feature to reformat the tracks which contain errors. If you specify YES for this prompt, the SD command reads and rewrites tracks in an attempt to clear the error condition. The default value is NO.

WARNING

This feature is potentially dangerous. Disk reformating under autocorrect can introduce more errors rather than correcting all errors. Use this feature with extreme caution or further loss of data may result. Also, this feature does not guarantee that data is correct in sectors in which an error has been cleared. All files should be processed to detect any structure or data errors, and the file should be rebuilt.

Related Commands:

RVI (Recover Volume Information)

4.196 SDT (SHOW DATE AND TIME)

Purpose:

The SDT command displays the current date and time to the terminal where you execute the SDT command. If you are in batch mode, the SDT writes the current date and time to the batch listing file.

Example:

In the following example, the SDT command displays the current date and time at the terminal where the SDT command is executed. The time is given according to the 24-hour clock with the hour in the range of 00 to 23.

```
[] SDT

13:34:10 TUESDAY, FEBRUARY 05, 1980.
```

Notes:

If the utility task (which is bid by the SDT command) is bid with a PARMS list, the first PARM in the list is treated as a synonym and the time and date text string is assigned to it. In this case, the time and date are not printed at the terminal.

The following example assigns the time and date text string to the first PARM in a PARMS list:

```
.BID TASK = 02,UTILITY,CODE = 11,PARMS = ($EUD$DAT)
```

Related Commands:

IDT (Initialize Date and Time)

SEM (SHOW EXPANDED MESSAGE)

4.197 SEM (SHOW EXPANDED MESSAGE)

Prompts:

```
SHOW EXPANDED MESSAGE
                MESSAGE CATEGORY:  alphanumeric
                      MESSAGE ID:  [alphanumeric/integer]
          INTERNAL ERROR CODE:     {integer/UNKNOWN}     (UNKNOWN)
```

Purpose:

The SEM command displays an expanded explanation of a system message. You can use SEM to display the message explanation and recommended user action for a message displayed at your terminal. To do so, specify the message category and message identifier contained in the second field of the brief message displayed on your terminal. The INTERNAL ERROR CODE should be left as UNKNOWN.

You can also use SEM to show the explanation and recommended action for an internal error code, such as an SVC error in an SVC call block. To do so, specify the message category, but omit the message ID. Specify the internal error code as a hexadecimal value.

To use the SEM command, the .S$EXPMSG files containing the expanded message must be on the system.

Prompt Details:

MESSAGE CATEGORY:
The characters to the left of the dash in the brief message displayed on your terminal.

MESSAGE:
The integer or character(s) to the right of the dash in the brief message displayed on your terminal. Enter leading zeros, if there are any.

INTERNAL ERROR CODE:
The hexadecimal error code returned to your program or displayed by DNOS. Specify the value with a leading zero or a leading >.

Example:

In the following example, suppose a user wants to print the contents of the file SYS1.KC0017.PRINT to a line printer. Upon execution of the Show File (SF) command, the message shown below is displayed on his terminal:

```
U SVC-0315 SYS1.KC0017.PRINT DOES NOT EXIST (SF;5)
```

SEM (SHOW EXPANDED MESSAGE)

The user could execute the SEM command as shown below to obtain an expanded explanation of the message:

```
[] SEM

SHOW EXPANDED MESSAGE
     MESSAGE CATEGORY: SVC
            MESSAGE ID: 0315
  INTERNAL ERROR CODE: UNKNOWN
```

The following explanation will be displayed on the user's terminal:

> Explanation:
> The specified device, file or channel does not exist. If the specified pathname includes the name of a remote site, this error may mean that the disk volume involved is not presently installed at the remote site or that the network software is not presently running.
>
> User Action:
> Make sure that any synonyms or logical name used in the pathname is properly defined. If a remote site is involved, make sure that the network software is running. If necessary, create the specified file or channel and try the operation again.

In this example, assume you have executed a program that issued an Executive Task SVC (SVC > 2B). The call block now has an error byte of > 28. To see what this error means you can use SEM as follows, where the internal error code is composed of the SVC code followed by the error code.

```
[] SEM

SHOW EXPANDED MESSAGE
     MESSAGE CATEGORY: SVC
            MESSAGE ID:
  INTERNAL ERROR CODE: >2B28
```

Both the short form and the explanation of the message will be shown.

Similarly, an internal error returned in a batch stream or from an SCI S$ routine can be used with SEM. If an S$ routine returned error code >910C, you could find the explanation as follows:

```
[] SEM

SHOW EXPANDED MESSAGE
     MESSAGE CATEGORY: SCI
            MESSAGE ID:
  INTERNAL ERROR CODE: >910C
```

Both the short form and the explanation of the message will be shown.

SF (SHOW FILE)

4.198 SF (SHOW FILE)

Prompts:

SHOW FILE
FILE PATHNAME: [site:][filename]@ (*)

Purpose:

The SF command displays the contents of a file at your VDT or prints the contents of a file at your data terminal. When in VDT mode, the first part of the file is displayed at the terminal. To display the rest of the file, press the appropriate function keys on the keyboard as described in Table 4-18. You cannot enter another command while the contents of a file are being displayed.

NOTE

In Table 4-18, the F3, F4, F6, Next Field, and Previous Field keys will not perform the described functions unless you specify VDT for the DEFAULT MODE for the terminal. Specify the DEFAULT MODE for the terminal via the Modify Terminal Status (MTS) command.

Prompt Details:

FILE PATHNAME:
The file name that identifies the disk file to be displayed.

Example:

In the following example, the contents of the file VOL1.DIRA.FILE1 will be displayed on the user's terminal by executing the SF command as follows:

```
[] SF

SHOW FILE
        FILE PATHNAME: VOL1.DIRA.FILE1
```

SF (SHOW FILE)

Notes:

The control keys available to the SF command are also used when output from an SCI command is directed to the terminal local file. For instance, if a null response to a command prompt causes output to appear at the VDT screen, the control keys function as they do for the SF command.

If you use the SF command on a secured file for which you do not have read access, you will receive an error message or return to the main menu. This occurs if you have some access rights, such as write access, to the file.

Related Commands:

PF (Print File)

Table 4-18. Show File Control Keys (VDT Mode Only)

Key	Description
F1	Causes the display of the next portion of a file (the amount displayed depends on the type of terminal). When the end-of-file is reached, pressing the F1 key has no further effect.
F2	Causes the display of a previous portion of a file. If the beginning of the file is displayed, the F2 key has no effect.
F3	Causes the display to scroll horizontally 10 columns to the left. Scrolling may continue through column 160. The bottom line of the VDT screen displays a scaling line, unless the display is positioned in column 1. The last character which can be shown is in character position 240 of the input record. The terminal must be in default VDT mode (see the MTS command) for this function to operate as described.
F4	Causes the display to scroll horizontally 10 columns to the right. The bottom line of the VDT screen displays a scaling line, unless the display is positioned in column 1. The terminal must be in default VDT mode.
F5	Performs no function while displaying a file. If you press this key while showing a file, an error message is displayed.
F6	Causes line numbers to be displayed in the leftmost columns of the VDT screen. Pressing the F6 key again causes the line numbers to be removed from the screen. The terminal must be in new and default VDT modes.
F7	Performs no function while displaying a file. If you press this key while showing a file, an error message is displayed.
F8	Performs no function while displaying a file. If you press this key while showing a file, an error message is displayed.

SF (SHOW FILE)

Table 4-18. Show File Control Keys (VDT Mode Only) (Continued)

Key	Description
Next Field	Causes the display to scroll horizontally 1 column to the right. The bottom line of the VDT screen displays a scaling line unless the display is positioned in column 1. The terminal must be in default VDT mode.
Previous Field	Causes the display to scroll horizontally 1 column to the left. Scrolling can continue through column 160. The bottom line of the VDT screen displays a scaling line unless the display is positioned in column 1.The terminal must be in default VDT mode.
Previous Line	Causes the entire display to scroll up one line, which displays the next line of the file at the bottom of the screen.
Next Line	Causes the entire display to scroll down one line, which displays the previous line of the file at the top of the screen. If the first line of the file is displayed at the top of the screen, the Next Line key has no effect.
+ Integer then Return	Entering a plus sign (+) and integer (decimal or hexadecimal) and then pressing the Return key positions the display of the file contents ahead of the number of lines specified by the integer.
Integer then Return	Entering an integer and pressing the Return key positions the display of the file contents to the line specified by the integer.
– Integer then Return	Entering a minus sign (–), an integer (decimal or hexadecimal), and then pressing the Return key positions the display of the file contents back the number of lines specified by the integer.
Command	Terminates the display of a file and allows entry of another command.

SGND (SNAPSHOT GLOBAL NAME DEFINITIONS)

4.199 SGND (SNAPSHOT GLOBAL NAME DEFINITIONS)

Purpose:

The SGND command places the current set of global logical names in a permanent file. When the set of global names in memory is altered as a result of an Assign Logical Name (ALN) command or a Release Logical Name (RLN) command, use this command to place the new set of names in the permanent file.

Notes:

The GLOBAL FILE PATHNAME prompt is not displayed when you use the SGND command in the interactive mode. However, you can specify this prompt in expert mode or batch mode. The response to GLOBAL FILE PATHNAME can be any file name.

Related Commands:

ALN	(Assign Logical Name)
LLN	(List Logical Names)
RLN	(Release Logical Name)

SIR (SHOW INTERNAL REGISTERS)

4.200 SIR (SHOW INTERNAL REGISTERS)

Prompts:

```
SHOW INTERNAL REGISTERS
                    RUN ID:  integer          (*)
```

Purpose:

The SIR command displays the contents of the internal registers (program counter (PC), workspace pointer (WP), and status register (ST)) of a task along with the task state. The displayed state is the state of the task before it is suspended to show the contents of the internal registers, while the remainder of the display reflects the values in effect after the task is suspended.

The character string representation of the status register follows the hexadecimal value and can include the following characters:

L	=	logical greater than
A	=	arithmetic greater than
E	=	equal
C	=	carry
O	=	overflow
P	=	parity
X	=	XOP in progress
S	=	privileged mode
M	=	map file

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

Example:

In the following example, the SIR command displays the internal registers and task state of a task with a run ID of >0E.

```
[] SIR


SHOW INTERNAL REGISTERS
           RUN ID: >0E

RUN ID ID=0E  STATE=06  WP=0082  PC=2FCF  <PC>=020A  ST=218F  E  M
```

SIR (SHOW INTERNAL REGISTERS)

Notes:

In the above example, the terminal is in TTY mode. In VDT mode the entire debug panel is displayed.

Related Commands:

MIR	(Modify Internal Registers)
SP	(Show Panel)

SIS (SHOW I/O STATUS)

4.201 SIS (SHOW I/O STATUS)

Prompts:

```
SHOW I/O STATUS
                        LUNO:  [integer]
                        TYPE:  {ALL/GLOBAL/TASK/JOB}          (JOB)
           JOB NAME OR ID:  [{alphanumeric/integer}]
```

The following prompt is displayed if the response to the TYPE prompt was TASK:

```
SHOW TASK LOCAL I/O STATUS
                   TASK RUN ID:  [integer]
```

Purpose:

You can view the input/output status of global, job-local, and task-local logical units by entering the SIS command. You can view the status of one LUNO or of all LUNOs of a particular type (global, job-local, or task-local). Table 4-19 describes the headings under which the status information can be displayed (not all headings apply to all LUNO types).

Prompt Details:

LUNO:
The logical unit number with status being requested. If you do not enter a LUNO, all LUNOs of the type specified are displayed.

TYPE:
A character string (ALL, GLOBAL, TASK, JOB) to identify the type of LUNO with status being requested. The string can be abbreviated by entering a character that uniquely identifies the string. For example, one may enter JOB for a job-local LUNO. The response of ALL lists all LUNOs assigned. The GLOBAL response lists global LUNOs. The TASK response lists task-local LUNOs.

JOB NAME OR ID:
Enter either the name of a job (specified at log-on) or the ID of a job (determined by the output of the List Jobs (LJ) command). If you do not specify a response to this prompt, the LUNO status of all jobs is displayed by user ID for the user who executed the SIS command.

TASK RUN ID:
The ID you enter must be the ID assigned by the system (run-time ID) when you placed the task in execution. If task-local LUNOs are to be displayed and you do not specify a task ID, the LUNO data will be displayed for all tasks in the specified job.

SIS (SHOW I/O STATUS)

Example:

In the following example, the SIS command displays the LUNO status for all jobs associated with the user who executed the SIS command:

```
[] SIS

SHOW I/O STATUS
                    LUNO:
                    TYPE: JOB
         JOB NAME OR ID:
```

The following status information is displayed on the user's terminal:

```
SCOPE OPN    TASK     JOB          RESOURCE
&LUNO ACC    NAME     NAME    TYPE & ACCESS NAME
----- --- ------- --------  ---- ------------------
J  2D              JC        LP:S LP01
J  38              JC        SEQ. .KC0017.MYFILE
J  3F              JC        911  ST01
```

Notes:

The system operator can execute the SIS command for jobs with any associated user ID and for all LUNOs in the system.

Related Commands:

SCS	(Show Channel Status)
SOS	(Show Output Status)

SIS (SHOW I/O STATUS)

Table 4-19. Information Displayed in Response to an SIS Command

Heading	Information Displayed
SCOPE &LUNO	The scope of the LUNO (task, job, shared or global) and the LUNO ID.
OPN ACC	The access privilege assigned to the LUNO when the job or task opened the LUNO. If the LUNO is not open, this field is blank. The following are valid access privileges that can be assigned to the LUNO: RO -- Read only EW -- Exclusive write SH -- Shared EA -- Exclusive all
TASK NAME	For task-local LUNOs, the name of the task which assigned the LUNO. For other LUNOs, the name of the task which has opened the LUNO, if any.
JOB NAME	For task and job-local LUNOs, the name of the job in which the task that assigned the LUNO resides. For global LUNOs, the name of the job in which the task that has opened the LUNO, if any, resides.
TYPE	The type of resource assigned to the LUNO. For example, LP:S designates a serial line printer, SEQ designates a sequential file, and 911 designates a 911 VDT.
ACCESS NAME	The access name of the resource assigned to a LUNO. For example, a file name or device name.

SJS (SHOW JOB STATUS FOR <user ID>)

4.202 SJS (SHOW JOB STATUS FOR < user ID>)

Prompts:

```
SHOW JOB STATUS FOR < user ID>
              JOB NAME OR ID:  [character(s)]          (*)
        LISTING ACCESS NAME:   [site:][pathname]@
```

Purpose:

The SJS command lists the status of any job currently executing under your user ID.

Prompt Details:

JOB NAME OR ID:
The job name specified or job ID returned during job creation. If you enter a null response, SCI outputs the status of all jobs with your user ID.

LISTING ACCESS NAME:
The pathname of the device or file to which SCI should list the job status information. The default value is your terminal local file.

Example:

In the following example, a user with the ID of KC0017 wants to view the status of a job named PRINT and output a listing of the job status to the terminal. Entering the SJS command as shown below displays this information:

```
[] SJS

SHOW JOB STATUS FOR KC0017
          JOB NAME OR ID: PRINT
    LISTING ACCESS NAME:
```

The following information would be displayed:

```
USER-ID   JOB-NAME    ID    PRI   STATE   JCASIZE   CURRENT   MAXIMUM
KC0017    PRINT      0022    3      2       4000     2548       3477
```

Refer to the Show Task Status (STS) command for details concerning information listed by SJS.

Messages:

SCI displays the following job information to the device or file entered in response to the LISTING ACCESS NAME prompt: user ID, job name, job ID, job priority, JCA size, and job state (that is, executing or suspended).

SJS (SHOW JOB STATUS FOR <user ID>)

Related Commands:

HJ	(Halt Job)
KJ	(Kill Job)
LJ	(List Jobs)
MJP	(Modify Job Priority)
RJ	(Resume Job)
STS	(Show Task Status for < user ID>)
XBJ	(Execute Batch Job)
XJ	(Execute Job)

4.203 SL (SHOW LINE)

After you activate the Text Editor, the SL command repositions the input file to display a designated line within the file. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the SL command and the following prompt:

```
SHOW LINE
    LINE:  {BEGINNING/END/line number}  (1)
```

SMM (SHOW MEMORY MAP)

4.204 SMM (SHOW MEMORY MAP)

Purpose:

The SMM command monitors memory usage by system and user tasks and estimates if more or less memory is needed for the operating system to handle these tasks. This information is displayed on the terminal, and is updated every two seconds. If any portion of the memory display is blank, that memory is currently not in use.

The F1, F2, and F3 function keys are used to show which blocks of memory are being used for system and user tasks. The use of these keys is defined as follows:

F1 Use the F1 function key to display the static memory being used by the system. When you press the F1 key, the blocks of static system memory in the display are replaced by the S character. Press the F1 key again to remove the S character(s) from the display.

F2 Use the F2 function key to display the memory segments being used by user tasks. When you press the F2 key, the blocks of user memory in the display are replaced by the U character. Press the F2 key again to remove the U character(s) from the display.

F3 Use the F3 function key to display the cache memory. When you press the F3 function key, the blocks of cache memory in the display are replaced by the C character. This software cache holds segments of user memory that are not currently executing. It allows faster access than retrieving the segments from disk the next time they are needed. Press the F3 key again to remove the C character(s) from the display.

Table 4-20 defines the fields displayed with the memory information.

SMM (SHOW MEMORY MAP)

Example:

In the following example, the SMM command displays memory usage information on the terminal local file.

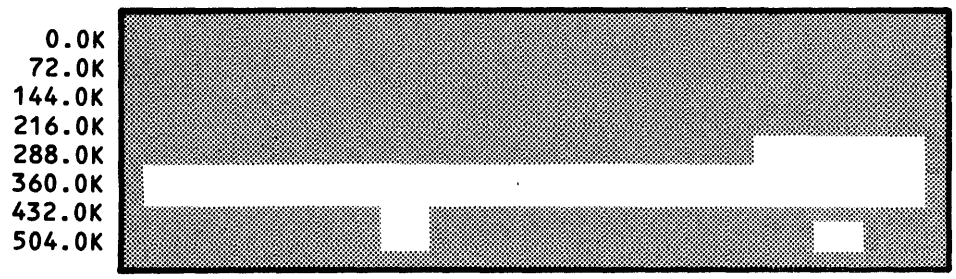
```
                        DNOS DYNAMIC MEMORY MAP

          MEMORY SIZE = 576.0 K              MEMORY RESIDENT OS = 162.3 K

          # OF SEGMENTS IN SOFTWARE CACHE = 70     TOTAL SIZE = 78.5 K

          # OF SEGMENTS SWAPPED OUT = 35           TOTAL SIZE = 194.0 K

# OF AVAILABLE MEMORY BLOCKS = 28   TOTAL SIZE = 145.0 K   LARGEST = 37.3 K
```




```
                    K = 1024 BYTES    X = 1152

2285321      F1-STATIC MEMORY   F2-USED SEGMENT MEMORY   F3-CACHE MEMORY
```

SMM (SHOW MEMORY MAP)

Press the F1, F2, and F3 function keys, and the S, U, and C characters appear in the display specifying system, user, and cache memory, respectively.

```
                        DNOS DYNAMIC MEMORY MAP

          MEMORY SIZE = 576.0 K              MEMORY RESIDENT OS = 162.3 K

          # OF SEGMENTS IN SOFTWARE CACHE = 70     TOTAL SIZE = 78.5 K

          # OF SEGMENTS SWAPPED OUT = 35           TOTAL SIZE = 194.0 K

# OF AVAILABLE MEMORY BLOCKS = 28   TOTAL SIZE = 145.0 K   LARGEST = 37.3 K

          0.0K SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
         72.0K SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
        144.0K SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
        216.0K SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
        288.0K SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
        360.0K
        432.0K UUUUUUUUUUUUUUUUU   CCCCCCCCCCUUUUUUUUUUUUUUUUUUUUUUU
        504.0K UUUUUUUUUUUUUUUUU   CCCCCCCCCCUUUUUUUUUUUUUUUU   UUUUU

                        K = 1024 BYTES   X = 1152

2285322          F1-STATIC MEMORY   F2-USED SEGMENT MEMORY   F3-CACHE MEMORY
```

Table 4-20. Fields of the SMM Display

Field	Description
MEMORY SIZE	Total amount of memory available to to the operating system
MEMORY RESIDENT OS	Amount of memory occupied by the memory resident portion of the operating system
# OF SEGMENTS IN SOFTWARE CACHE	Number of segments cached in memory for future use
TOTAL SIZE	Amount of memory occupied by the cache segments
#OF SEGMENTS SWAPPED OUT	Number of segments on roll file
TOTAL SIZE	Amount of memory that was occupied by rolled segments
# OF AVAILABLE MEMORY BLOCKS	Number of fragments (holes) of available user memory
TOTAL SIZE	Amount of available user memory
LARGEST	Largest fragment of available user memory

SMS (SHOW MEMORY STATUS)

4.205 SMS (SHOW MEMORY STATUS)

Prompts:

SHOW MEMORY STATUS
LISTING ACCESS NAME: [site:][pathname]@

Purpose:

The SMS command displays information on system memory usage. The operating system responds to the entry of the SMS command by displaying information about memory size, system size, and system table area sizes.

Prompt Details:

LISTING ACCESS NAME:
The device or file name to which the returned information is to be written. The default is the terminal local file.

Example:

In the following example, the SMS command returns the system memory usage information to the terminal local file.

```
[] SMS

SHOW MEMORY STATUS
    LISTING ACCESS NAME:


                          SHOW MEMORY STATUS

                    SYSTEM TABLE AREA = 6154 BYTES
CURRENT USAGE = 2880 BYTES              LARGEST AREA USED =3664 BYTES

                    USER JCA AREA = 2048 BYTES
CURRENT USAGE = 1042 BYTES              LARGEST AREA USED =1304 BYTES

                    SYSTEM JCA AREA = 4096 BYTES
CURRENT USAGE = 1042 BYTES              LARGEST AREA USED =1472 BYTES

               SEGMENT MANAGER AREA    1 = 2048 BYTES
CURRENT USAGE =  818 BYTES              LARGEST AREA USED = 868 BYTES

               FILE MANAGEMENT AREA    1 = 4096 BYTES
CURRENT USAGE =  518 BYTES              LARGEST AREA USED = 518 BYTES
```

SND (SNAPSHOP NAME DEFINITIONS)

4.206 SND (SNAPSHOT NAME DEFINITIONS)

Prompts:

```
SNAPSHOT NAME DEFINITIONS
    SYNONYM FILE PATHNAME:  [site:]pathname@
        PARMS LIST (IF ANY):  [character(s)]
```

Purpose:

The SND command takes a snapshot of all user-defined synonyms and/or logical names and writes them to the pathname specified in response to the SYNONYM FILE PATHNAME prompt. The parameters specified for the PARMS LIST prompt, if any, are also written to the file which contains the user-defined synonyms.

The SND command is useful if a job other than SCI is to be bid when you log on at a terminal and synonyms, logical names, or parameters are to be passed to that job.

The user can specify through the MTS command that a pathname for the synonym and logical names be supplied when logging on a terminal. This is done by responding YES to the ASK FOR NAME MANAGER FILES? prompt of the MTS command. The pathname specified at log-on would be the same as that specified for the SND command.

Prompt Details:

SYNONYM FILE PATHNAME:
The pathname of a file that will contain the set of synonyms and logical names to be used by a job.

PARMS LIST (IF ANY):
A user-defined character string that is to be used as parameters for a job. If you specify more than one parameter, separate them by commas. If the synonym file is to be used for the Execute Batch Job (XBJ) command, you must specify two parameters: an input access name and a listing access name. If the synonym file is to be used for the Execute Job (XJ) command, parameters must be supplied as needed by the initial task in that job. That is, if you bid the task within a command procedure, the PARMS LIST will be the same as the PARMS LIST specified by the .BID or .QBID primitive within the command procedure. Refer to the .BID primitive in the *DNOS Systems Programmer's Guide* for information concerning the PARMS LIST.

SND (SNAPSHOP NAME DEFINITIONS)

Example:

In the following example, a job other than SCI is to be bid when a user logs on at a terminal. The job keeps track of accounting information from the time the user logs on until the user logs off the terminal. The SND command can be used to pass user-defined synonyms and logical names to the job and also to pass the user ID and passcode as parameters to the initial task of the job.

```
[] SND

SNAPSHOT NAME DEFINITIONS
       SYNONYM FILE PATHNAME: .USER.SYN
         PARMS LIST (IF ANY): USER05,U05
```

Using the previous example, a batch job is to be bid with XBJ. The input access name and listing access name of the batch job is .USER.COM and .USER.LIST, respectively. The SND command can be used to pass this information to the initial task of the batch job.

```
[] SND

SNAPSHOT NAME DEFINITIONS
       SYNONYM FILE PATHNAME: .USER.SYN
         PARMS LIST (IF ANY): .USER.COM,.USER.LIST
```

Related Commands:

MTS	(Modify Terminal Status)
XJ	(Execute Job)
XBJ	(Execute Batch SCI Job)

4.207 SOS (SHOW OUTPUT STATUS)

Prompts:

```
SHOW OUTPUT STATUS
                    USER ID:  [alphanumeric]
DEVICE/CLASS NAME:  [site:][devicename/alphanumeric/ALL]@
```

Purpose:

The SOS command is used to show the following types of status of queued spooler output:

- The status of your files (as identified by your user ID) in the output queue.
- The status of all users' files in the output queue.
- The status of queued files waiting for output at a particular device or class.
- The status of queued files waiting for output at all devices. You cannot determine the status of a device used for output by using the SOS command (device errors are written to the system log).

Table 4-21 describes the information that is given under each heading of the SOS display.

Prompt Details:

USER ID:
The user ID associated with the files, in the specified output queue, whose status information is to be displayed. If you enter a null response for this prompt, the status of all users' files in the specified output queue is displayed.

DEVICE/CLASS NAME:
The name of the device that has the output queue containing the files whose status is to be displayed. If you enter a null response, the status of queued files waiting for output at all devices at your local site will be displayed. To see the same kind of display for a network site, you must enter < site> :ALL (not just the site name).

SOS (SHOW OUTPUT STATUS)

Example:

In the following example, the output status of files waiting on the output queue at a line printer LP01 is to be displayed. Only files associated with a particular user ID are listed. The class names for the device are listed with the device name. The responses to the SOS prompts result in the following display:

```
[] SOS

SHOW OUTPUT STATUS
              USER ID:  GEORGE
   DEVICE/CLASS NAME:  LP01

DEVICE =LP01          STATUS =EXCLUSIVE          FORM =STANDARD
CLASS NAMES  (LP          SLP         )
ST/     USER        FORM     SPOOL      LOGICAL NAME OR
PRI      ID                    ID          FILE NAME
A/4   GEORGE      STANDARD  S00004  .FILMGR.LIST.MYFILE
W/4   GEORGE      STANDARD  S00005  .FILMGR.SRC.MYFILE
W/4   GEORGE      STANDARD  S00006  .FILMGR.OBJ.MYFILE
                                      (<COPIES= 002>)
```

Notes:

If the output status of more than one device is to be listed, the class names will be displayed after all devices are displayed.

Related Commands:

HO	(Halt Output at Device)
KO	(Kill Output at Device)
MO	(Modify Output at Device)
RO	(Resume Output at Device)

Table 4-21. Description of Information Output by the SOS Command

Heading	Description
DEVICE	The device name at which file information is being or waiting to be output.
ST/PRI	The status (ST) of a file is Active (A) if it is currently outputting information at a device and Waiting (W) if it is waiting to output information at a device. PRI is the queue priority of the file. The range of the priority is 0 through 31, where 0 is highest priority.
USER ID	The user ID of a user whose file is waiting in the output queue to output information.
FORM	The printing form to be used to print the file.
SPOOL ID	The unique identifier that is assigned when a file is queued to print.
LOGICAL NAME OR FILE NAME	The logical or file name of the file that is currently outputting information or that is waiting to output information.
CLASS NAME	The name of a class of output devices on which the file might be printed. Class names are assigned by the Modify Spooler Device (MSD) command.

SP (SHOW PANEL)

4.208 SP (SHOW PANEL)

Prompts:

```
SHOW PANEL
                    RUN ID:  integer              (*)
          MEMORY ADDRESS:  [full exp]
```

Purpose:

The SP command displays the debug panel for a specified task. If the task is not unconditionally suspended, it is temporarily suspended while the panel is being formatted and displayed. The displayed task state is the state of the task before it is suspended. The debug panel consists of:

- Internal registers
- Workspace registers
- Breakpoints
- Memory display
- Task state

If the debug panel displayed is for a system task, you must have a privilege level of two or higher. For VDTs, the memory displayed is the number of lines available on the VDT screen. For hardcopy devices, four lines of memory are displayed.

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

MEMORY ADDRESS:
The integer value that is the starting memory address for the memory portion of the debug panel display. The default value is the current program counter (PC) address.

SP (SHOW PANEL)

Example:

In the following example, the SP command displays the debug panel for a task with a run ID of > 1A. The memory portion of the display begins with memory address > 56.

```
[] SP

SHOW PANEL
                RUN ID: >1A
        MEMORY ADDRESS: >56


RUN ID = 1A  STATE = 06    WP = 4106  PC = 0056  <PC> = 06A0  ST = 018F
                    W O R K S P A C E   R E G I S T E R S
4106    0000  0000  0000  0000  0000  0000  0000  0000  .. .. .. .. .. .. .. ..
4116    0000  0000  0000  0000  0000  0000  0000  0000  .. .. .. .. .. .. .. ..
                          B R E A K P O I N T S

                                  M E M O R Y
0056    06A0  1016  244D  4149  4E20  C820  02A8  4216  .. .. $M AI N  .  .. B.
0066    C120  4216  0A34  0420  2B32  0FA0  02AA  0FE4  .  B. .4 .  +2 .. .. ..
0076    41DE  0C0E  05A0  4216  8820  4216  02B2  12F0  A. .. .. B. .  B. .. ..
0086    C820  02A8  4216  C120  4216  0A34  0420  2B32  .  .. B. .  B. .4 .  +2
```

Notes:

The SP command affects the memory address displayed by the debugger for tasks in debug mode. When a task in debug mode (controlled state) is halted by a time out, a panel is displayed using the last memory address specified on a SP command. If a null response is given, the current PC address is used.

Related Commands:

SWR (Show Workspace Registers)

SPI (SHOW PROGRAM IMAGE)

4.209 SPI (SHOW PROGRAM IMAGE)

Prompts:

```
SHOW PROGRAM IMAGE
                    PROGRAM FILE:  [site:]filename@          (*)
          OUTPUT ACCESS NAME:      [site:][pathname]@        (*)
                    MODULE TYPE:   {T/P/O/S}                 (*)
          MODULE NAME OR ID:       {alphanumeric/integer}    (*)
                    ADDRESS:       integer                   (*)
                    LENGTH:        integer                   (040)
```

Purpose:

The SPI command displays the disk-resident memory image of a module (defined as a task, procedure, or overlay) for a specified program. It also displays a checksum of the image, which can be used for a crude object comparison.

Prompt Details:

PROGRAM FILE:
The file name of the program file on which the program (task, procedure, overlay, or segment) has been installed.

OUTPUT ACCESS NAME:
The device name or file name where the display of the memory image of the program is to be written. The default value is the terminal local file.

MODULE TYPE:
The type of program with a memory image to be displayed. The following characters are valid responses:

T = Task
P = Procedure
O = Overlay
S = Program Segment

MODULE NAME OR ID:
The characters or the associated ID that identifies the program on the specified program file.

ADDRESS:
The integer value that is the starting address of the memory image to be displayed.

LENGTH:
The integer value in the range 0 through 32767 that is the number in words of the memory image to be displayed.

SPI (SHOW PROGRAM IMAGE)

Example:

In the following example, the SPI command displays eight words of memory image for a procedure installed on a user's program file. The display is written to the terminal local file of the terminal where the SPI command was executed.

```
[] SPI

SHOW PROGRAM IMAGE
            PROGRAM FILE: DS02.PROG
      OUTPUT ACCESS NAME:
              MODULE TYPE: P
       MODULE NAME OR ID: PRINT
                  ADDRESS: 02000
                   LENGTH: 8

2000  1EFE 1306 9815 1EFA 1605 0585 0606 1220 .. .. .. .. .. .. .. ..
CHECKSUM = F8C9
```

Related Commands:

MPI (Modify Program Image)

SPS (SHOW PASCAL STACK)

4.210 SPS (SHOW PASCAL STACK)

The SPS command displays data from the stack structure of the currently executing process and other data pertaining to a specified Pascal task. Refer to the *DNOS TI Pascal Programmer's Guide* for further information on the SPS command and the following prompt:

```
SHOW PASCAL STACK
                    RUN ID:   integer exp
```

4.211 SRF (SHOW RELATIVE TO FILE)

Prompts:

```
SHOW RELATIVE TO FILE
                    PATHNAME:  [site:]filename@        (*)
               RECORD NUMBER:  integer                 (*)
                  FIRST WORD:  integer                 (*)
            NUMBER OF WORDS:   [integer]
          OUTPUT ACCESS NAME:  [site:][pathname]@      (*)
```

Purpose:

The SRF command displays any word or group of words within a file. It assumes that you have knowledge of the file structure and allows you to address any word within the file.

Prompt Details:

PATHNAME:
The name of the file with the data to be displayed.

RECORD NUMBER:
The integer value that is the physical record number within the file of the first word of data to be displayed.

FIRST WORD:
The integer value that is the byte offset within the physical record of the first word of data to be displayed. This value will be rounded down to an even value.

NUMBER OF WORDS:
The integer value that is the number of words of the record to be displayed. If the number of words specified is larger than the number of words left in the record, words from succeeding records will be displayed. The default is to display the number of words required to reach the end of the physical record.

OUTPUT ACCESS NAME:
The pathname of a device or file where the results of the SRF command are to be listed. The default is the terminal local file.

SRF (SHOW RELATIVE TO FILE)

Example:

In the following example, the SRF command lists the contents of six words for the first record of a user's file, beginning at word zero. The results are listed on the terminal local file.

```
[] SRF

SHOW RELATIVE TO FILE
                PATHNAME: .KC0017.XB
           RECORD NUMBER:  0
              FIRST WORD:  0
         NUMBER OF WORDS:  6
      OUTPUT ACCESS NAME:

FILE:  .KC0017.XB  RECORD: 000000
0000  0000  0018  1A0B  5842  2028  4558 .. .. .. XB    ( EX
CHECKSUM:6D3E   RECORD:000000
CUMULATIVE CHECKSUM:6D3E
```

Notes:

If the file can be thought of as a contiguous area of memory, it is possible to display data from the file by address relative to the beginning of the data. To do this, specify zero for the RECORD NUMBER and enter the address of the desired data for the FIRST WORD. For example, enter the following:

```
        PATHNAME: .YOURFILE
  RECORD NUMBER: 0
     FIRST WORD: 5000
NUMBER OF WORDS: 1
```

This technique will not work for words more than 65535 bytes from the beginning of the file. To access those words, it is necessary to determine the physical record length for the file. Then enter a RECORD NUMBER such that the expression:

Relative Address – (RECORD NUMBER * Physical Record Length)

has a value less than 65535. For example, to display the word at address 66666 of a file which has a physical record length of 256, enter the following:

```
        PATHNAME: .YOURFILE
  RECORD NUMBER: 20
     FIRST WORD: 66666-20*256
NUMBER OF WORDS: 1
```

Related Commands:

MRF (Modify Relative to File)

4.212 ST (SIMULATE TASK)

Prompts:

```
SIMULATE TASK
        FOR:  [full exp]  (*)
       FROM:  [full exp]
         TO:  [full exp]
```

Purpose:

The ST command provides controlled and traced execution of the instructions in a task. Controlled execution continues until the execution of a specified number of instructions has been simulated or until a specified address is placed in the program counter (PC) or the occurrence of a breakpoint or simulated breakpoint, whichever occurs first. Press the F3 function key to continue simulation.

Simulated execution continues without operator intervention and locks out further SCI commands. Following simulation of the instruction with an address specified by the response to the TO prompt, SCI displays the debug panel and halts simulation. You can regain SCI capabilities by pressing the Command key to return to command mode.

When the number of specified simulations is performed, SCI displays the following message and halts simulation:

```
TIME OUT
```

Single instruction execution is performed by specifying a one (1) in response to the FOR prompt.

You can use the ST or Resume Simulated Task (RST) command to reenter simulated execution.

Prompt Details:

FOR:

The integer expression that specifies the number of instruction simulations to be performed which must be less than or equal to 32,767. When the specified number of simulations has been performed, SCI displays the following message and halts simulation:

```
TIME OUT
```

If you enter a null response for this prompt, the value specified in a previous ST command is used; if you have not executed a previous ST command, a one is used.

FROM:

The integer expression that specifies the address of the first instruction to be simulated. If you enter a null response, simulation begins with the instruction at the address in the current PC of the controlled task.

ST (SIMULATE TASK)

TO:

The integer expression that specifies the address of the last instruction to be simulated. The integer expression entered can be less than that entered for the FROM command. If you enter a null response in response to this prompt, simulation continues until a breakpoint or simulated breakpoint is encountered or until you press the Command key, returning SCI to command mode.

Example:

In the following example, the ST command simulates 25 instructions of a program named IDTNAM. Simulation begins at location .BEGIN and halts either after simulating the specified number of instructions or at location .END. The use of program labels requires that the modules being debugged have a symbol table. If you do not use a symbol table, you must supply numerical addresses.

```
[] ST

SIMULATE TASK
             FOR: 25
            FROM: .IDTNAM.BEGIN
              TO: .IDTNAM.END
```

Assumptions:

The program to be simulated is placed in a controlled mode by the previous execution of the Execute in Debug Mode (XD) command.

Messages:

STOP AT TRAP NO. X

where X is the number of the simulated breakpoint set through the Assign Simulated Breakpoint (ASB) command.

Notes:

When a single instruction execution is finished, the next instruction cannot be data to be examined; it must be another instruction since it becomes a breakpoint command.

Related Commands:

RST	(Resume Simulated Task)
XD	(Execute in Debug Mode)

4.213 STI (SHOW TERMINAL INFORMATION)

Prompts:

```
TERMINAL INFORMATION..SITE:site..TERMINAL:stationname..USER:alphanumeric
     MOST RECENT TASK RUN ID:  [integer]                    (*)
MOST RECENT LUNO ASSIGNMENT:   [integer]                    (*)
          TEXT EDIT IN PROGRESS:  [YES/NO]                  (*)
          MOST RECENT EDIT FILE:  [pathname]@               (*)
          TASK BEING DEBUGGED:    [integer]                 (*)
       SCU SESSION IN PROGRESS:   [YES/NO]                  (*)
```

Purpose:

The STI command examines the status of a terminal.

The operating system responds to the entry of an STI command by displaying a command description message containing the station name of your terminal and your user ID.

You can change the following terminal status information:

- Most recent task run ID
- Most recent LUNO assignment

Prompt Details:

TERMINAL:
The station name of the terminal with the status information to be displayed. You cannot modify this value.

USER:
The user ID assigned at log-on time to your job.

MOST RECENT TASK RUN ID:
The run-time ID of the most recently activated task. You can modify this value.

MOST RECENT LUNO ASSIGNMENT:
The job-local LUNO most recently assigned by the user ID. You can modify this value.

TEXT EDIT IN PROGRESS:
If YES, a text edit is currently in progress. Otherwise, there is no text edit in progress. You cannot modify this value.

MOST RECENT EDIT FILE:
The pathname of the most recently edited file. You cannot modify this value.

TASK BEING DEBUGGED:
The task ID of a task currently being debugged. You cannot modify this value.

STI (SHOW TERMINAL INFORMATION)

SCU SESSION IN PROGRESS;
If YES, a System Configuration Utility (SCU) session is currently in progress. If NO, an SCU session is not executing.

Example:

In the following example, the STI command displays the status information for the terminal named ST01 at the site named AUSTIN. The user ID assigned to the terminal at log-on time was WHT016, the ID of the task most recently run from the terminal is >86, and a text edit operation is not in progress.

```
[]  STI

TERMINAL INFORMATION..SITE: AUSTIN..TERMINAL: ST01..USER: WHT016
    MOST RECENT TASK RUN ID: >86
MOST RECENT LUNO ASSIGNMENT:
      TEXT EDIT IN PROGRESS: N
      MOST RECENT EDIT FILE:
        TASK BEING DEBUGGED:
    SCU SESSION IN PROGRESS:
```

STS (SHOW TASK STATUS FOR <user ID>)

4.214 STS (SHOW TASK STATUS FOR < user ID>)

Prompts:

```
SHOW TASK STATUS FOR < user ID>
              JOB NAME OR ID:   [character(s)/integer]     (*)
        LISTING ACCESS NAME:    [site:][pathname]@
```

Purpose:

The STS command gives a description of the status of tasks in one specific job or all jobs running under the user ID. The user ID displayed along with the full name of the command is the user ID of the user executing the STS command. Table 4-22 describes the information displayed for jobs, and Table 4-23 describes the information displayed for tasks.

Prompt Details:

JOB NAME OR ID:
The job name specified or job ID returned during job creation. If you do not enter a response, SCI outputs the status of all tasks and jobs under your user ID.

LISTING ACCESS NAME:
The device or file name to which the operating system writes the status information produced by the STS command. The default value is the terminal local file.

Example:

In the following example, the STS command displays the status of a job with the name PRINT to the terminal local file. The user ID is SYSTEM00.

```
[] STS

SHOW TASK STATUS FOR SYSTEM00
          JOB NAME OR ID: PRINT
     LISTING ACCESS NAME:


USER-ID   JOB-NAME    ID    PRI   STATE   JCASIZE   CURRENT   MAXIMUM
SYSTEM00  PRINT      0001    20     2       2048      828       1042
      TASK      ID RUN-ID STN   PRI R-PRI STATE WP    PC   FLAG1 FLAG2  CPU SECS
      FILMGR    05   02          0   0     24   590E  C06E  F100  0000       0.6
      SCI990    01   01      2   4   164   17   8796  3DD8  7000  0000       3.2
      SJSSTS    30   08      2   4   150   00   E0BC  DCF6  F000  0800       0.2
```

Related Commands:

KJ	(Kill Job)
KT	(Kill Task)
SJS	(Show Job Status for < user ID>)

STS (SHOW TASK STATUS FOR <user ID>)

Table 4-22. Job Information Produced by an STS Command

Heading	Description
USER-ID	The user ID associated with the job.
JOB-NAME	The name of the job.
ID	The ID of the job.
PRI	The priority of the job, which ranges from 0 through 31.
STATE	The state of the job. The following codes describe the state of the job:

Hexadecimal Code	Meaning
01	Job is being created
02	Job is in an executable state
03	Job is halted
04	Job is terminating
05	JCA is being expanded

JCASIZE	The size in bytes of the job communications area.
CURRENT	The current job communications area (JCA) in bytes in use.
MAXIMUM	The amount of JCA in bytes that has been allocated since this job began executing.

STS (SHOW TASK STATUS FOR <user ID>)

Table 4-23. Task Information Produced by the STS Command

Heading	Description
TASK	The installed name of the task.
ID	The installed ID of the task associated with the job status to be displayed.
RUN-ID	The run-time ID assigned to the task when the task is activated.
STN	The station number with which the executing task is associated.
PRI	The priority of the task. The following codes identify the priority of the task:

Code	Meaning
0	Task is a system task (highest priority)
1	Second priority
2	Third priority
3	Lowest priority
4	A floating priority with level adjusted depending on the task. This is the recommended priority for nonprivileged interactive user tasks.
R1 – R127	Real-time priorities, with 1 being the highest.
B1 – B4	Background priorities 1 through 4.

R-PRI	The run-time priority assigned by the system for scheduling is in the range of 0 through 255.
STATE	The state of the job or task. If a task, a code that describes the state of the task as follows:

Hexadecimal Code	Meaning
00	Task awaiting time slice allotment or active
01	Task awaiting memory
02	Task's job is in nonexecutable state
03	Task load in progress
04	Task is terminated
05	Task in time delay
06	Task unconditionally suspended
09	Task suspended for I/O
0F	Task suspended for abort I/O
14	Task awaiting overlay loader services
17	Task awaiting coroutine activation
19	Task has exceeded initiate I/O threshold
1E	Task is waiting for access to door (usually locked directory)
1F	Waiting for Scheduled Bid Task SVC
20	Waiting for Install Volume SVC
21	Waiting for Log Message SVC
22	Waiting for Disk Manager SVC
24	Waiting for Queue Input

STS (SHOW TASK STATUS FOR <user ID>)

Table 4-23. Task Information Produced by the STS Command (Continued)

Heading	Description

Hexadecimal Code	Meaning
25	Waiting for Install Task SVC
26	Waiting for Install Procedure SVC
27	Waiting for Install Procedure SVC
28	Waiting for Delete Task SVC
29	Waiting for Delete Procedure SVC
2A	Waiting for Delete Overlay SVC
2B	Waiting for Bid Task SVC
2D	Waiting for Read/Write Task SVC
30	Waiting for System Table Area
31	Waiting for Map Program Name to ID SVC
34	Waiting for Unload Volume SVC
36	Waiting for any I/O SVC
37	Waiting for Assign Program File Space SVC
38	Waiting for Initialize New Volume SVC
3D	Task Suspended for Semaphore
40	Waiting for Segment Manager SVC
42	Waiting for Event Completion
43	Waiting for Name Manager SVC
48	Waiting for Job Manager SVC
4A	Waiting for Forced Roll SVC
4C	Waiting for Return Code Processor

WP — The workspace pointer register address which is relative to the task in memory.

PC — The currently executing task instruction address which is relative to the task address in memory.

FLAG1 — A task status block flag within which bits are set to indicate the characteristics of a task as follows:

Bit	Task Characteristic
0	System task
1	Privileged task
2	Current segment set in memory
3	Take end-action on error
4	I/O has been aborted for task
5	Task being aborted
6	Reserved
7	Queue server task
8	Activate task outstanding
9	Initial task bid
10	Software privilege
11 – 15	Reserved

Table 4-23. Task Information Produced by the STS Command (Continued)

Heading	Description
FLAG2	A task status block flag within which bits are set to indicate the characteristics of a task as follows:

Bit	Task Characteristic
0	Task being controlled
1	Stopped by scheduler
2	Stopped by breakpoint
3	Task to be halted
4	Restart parent on termination
5	RBID task
6	Re-Issue XOP
7 – 15	Reserved

CPU SECS	The CPU seconds that the task has been executing.

SV (SHOW VALUE)

4.215 SV (SHOW VALUE)

Prompts:

SHOW VALUE
EXPRESSION: full exp

Purpose:

The SV command displays the value of a specified expression. The hexadecimal, decimal, and ASCII representations of the value are given.

Prompt Details:

EXPRESSION:
The integer and/or character expression with a value to be displayed. If a task is being debugged and is a controlled task, the expression can be symbolic.

Example:

In the following example, the SV command displays the value of the specified expression.

```
[] SV

SHOW VALUE
          EXPRESSION: >FE*(>DD->3B)

HEX:  >0000A0BC    DECIMAL: 41148    ASCII:'....'
```

4.216 SVL (SAVE LINES)

After you activate the Text Editor, the SVL command copies a designated set of contiguous lines in the current file to another file. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the SVL command and the following prompts:

```
SAVE LINES
                        START LINE:  [BEGINNING/END/line number]
                          END LINE:  [BEGINNING/END/line number]
                 SAVE FILE PATHNAME: [site:]pathname@
OPTION(ADD/REPLACE/EXTEND):         {ADD/REPLACE/EXTEND}          (ADD)
```

SVS (SHOW VOLUME STATUS)

4.217 SVS (SHOW VOLUME STATUS)

Prompts:

```
SHOW VOLUME STATUS
                VOLUME NAME:  [alphanumeric]@
                 DRIVE NAME:  [devicename]@
        OUTPUT ACCESS NAME:   [site:][pathname]@
```

Purpose:

The SVS command displays the status of a disk volume. Table 4-24 describes the information displayed or listed by an SVS command.

Prompt Details:

VOLUME NAME:
The name of the volume with the status to be displayed. If you omit this response, you must specify the drive name. If you supply both drive name and volume name, the volume name is used.

DRIVE NAME:
The device name of the disk unit on which the volume is installed. If you omit the drive name, you must supply the volume name. If you supply both names, the volume name is used. If you specify ALL as the drive name, a volume status list of all disk volumes currently installed is produced.

OUTPUT ACCESS NAME:
The pathname of the device or file to which the status information produced by the SVS command should be written. The default is the terminal local file.

Example:

In the following example, the SVS command lists the status of a volume mounted in disk unit DS02 at the terminal where the SVS command is executed:

```
[] SVS

SHOW VOLUME STATUS
             VOLUME NAME:
              DRIVE NAME: DS02
      OUTPUT ACCESS NAME:

VOLUME NAME: HAWK2   ADUS: 16320   #BAD: 0   BYTES/ADU: 288
AVAILABLE: 4519  LARGEST AVAILABLE BLOCK: 4519  CONTROLLER ERRORS:0
PRIMARY SYSTEM IMAGE:  CC4    SECONDARY SYSTEM IMAGE:
NAME INSTALLED:  HAWK2   DEVICE NAME: DS02
```

Related Commands:

INV	(Initialize New Volume)
IV	(Install Volume)
UV	(Unload Volume)

Table 4-24. Information Produced by an SVS Command

Field	Description
VOLUME NAME:	Name of the volume that is in the disk drive.
ADUS:	Total number of spaces on the volume in Allocatable Disk Units (ADUs) for directories, subdirectories, and files.
#BAD:	Number of bad ADUs on the disk volume.
BYTES/ADU:	Number of bytes per ADU in the volume.
AVAILABLE:	Total number of ADUs still available for directories, subdirectories, and files.
LARGEST AVAILABLE BLOCK:	Size in ADUs of the largest contiguous block available for a file.
CONTROLLER ERRORS:	Number of unsuccessful attempts to rewrite or reread data on the volume since the initial program load (IPL). These errors are detected by the disk controller. This value should normally be zero. However, a value from one through ten during a week should not be cause for alarm.
PRIMARY SYSTEM IMAGE:	If the volume is a system disk volume, the name given is the file on which the primary system image resides.
SECONDARY SYSTEM IMAGE:	If the volume is a system disk volume, the name given is the file on which the secondary image resides.
NAME INSTALLED:	Name of the disk volume that is currently installed. This field is blank if no volume is installed. If the name in this field is not the same as the volume name given previously, then this volume was removed from the drive without being unloaded, using the Unload Volume (UV) command. The UV command must be entered for this name before another can be installed. If this disk is being used by INV, this field is $$$$$$$$ and the disk cannot be unloaded.
DEVICE NAME:	The name of the device which contains the disk volume.

SWR (SHOW WORKSPACE REGISTERS)

4.218 SWR (SHOW WORKSPACE REGISTERS)

Prompts:

```
SHOW WORKSPACE REGISTERS
                    RUN ID:  integer  (*)
```

Purpose:

The SWR command displays the current workspace of a task. If the task is not unconditionally suspended, it is temporarily suspended while the workspace is displayed.

If the terminal requesting the command is in VDT mode, the SWR command functions the same as the Show Panel (SP) command with a memory address of zero. If the workspace to be displayed is for a system task, you must have a privilege level of two or higher assigned to your user ID.

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

Example:

In the following example, the SWR command displays the workspace for a task with a run ID of >1A.

```
[] SWR

SHOW WORKSPACE REGISTERS
            RUN ID: >1A

RUN ID = 1A  STATE = 06    WP = 4106  PC = 0030  <PC> = C820  ST = 018F
                    W O R K S P A C E   R E G I S T E R S
4106    0000  0000  0000  0000  0000  0000  0000  0000  .. .. .. .. .. .. .. ..
4116    0000  0000  0000  0000  0000  0000  0000  0000  .. .. .. .. .. .. .. ..
                        B R E A K P O I N T S

                                  M E M O R Y
0000    06A0  1016  244D  4149  4E20  C820  02A8  4216  .. .. $M AI N  .  .. B.
0010    C120  4216  0A34  0420  2B32  0FA0  02AA  0FE4  .  B. .4 .  +2 .. .. ..
0020    41DE  0C0E  05A0  4216  8820  4216  02B2  12F0  A. .. .. B. .  B. .. ..
0030    C820  02A8  4216  C120  4216  0A34  0420  2B32  .  .. B. .  B. .4 .  +2
```

Related Commands:

MWR	(Modify Workspace Registers)
SP	(Show Panel)

TFFPC (TRANSFER FILE FROM PC)

4.219 TFFPC (TRANSFER FILE FROM PC)

Prompts:

```
TRANSFER FILE FROM PC
                    TI PC FILE NAME:   MS™-DOS filename@     (*)
                 TI PC FILE FORMAT:    {T/TEXT/B/BINARY}     (TEXT)
               TI PC STATION NAME:     [site:][stationname]  (*)
BUSINESS SYSTEM FILE NAME:             [site:] filename@     (*)
                        REPLACE?:      YES/NO                (NO)
```

Purpose:

The TFFPC command allows you to move a file from a TI Professional Computer (PC) to a Business System computer. The command can be specified from any terminal or batch stream. The file transfer task works with a TI PC connected to a communications port on the Business System computer. The TI PC must be running the TI PC 931 Emulator package. The file transfer will not work from a TI PC directly connected to a local area network.

Any sequential ASCII or binary file can be transferred from the PC to a Business System computer. The resulting Business System file can be either a sequential file or a relative record file. The Business System file will be a sequential file unless it has been precreated as a relative record file. Any PC ASCII character can be transmitted to the Business System computer.

A station that is issuing the command can be a different station than the PC from which the data is transferred. In this case, the PC specified as TI PC STATION NAME must not be logged on and must be using the TI PC 931 Emulator package.

Prompt Details:

TI PC FILE NAME:
The MS-DOS name of the PC file to be sent to the Business System computer.

TI PC FILE FORMAT:
The reply to this prompt identifies the format of the file on the PC. If you respond TEXT, the file is in an editable MS-DOS file, and the MS-DOS record separator (>0D0A) indicates the end of the record. For each MS-DOS record, a record is written on the Business System file. For international versions of PC using their national character set, national characters are translated from the PC 8-bit code to the ISO 7-bit code. If you respond BINARY, characters from the MS-DOS file are sent to the Business System by record of 80 bytes, and no translation is performed.

* MS is a registered trademark of Microsoft Corporation.

TFFPC (TRANSFER FILE FROM PC)

TI PC STATION NAME:
The station name of the TI PC from which the file is to be sent. This is always a station name on a Business System computer to which the PC is connected or remotely logged on. The default value is your station ID.

BUSINESS SYSTEM FILE NAME:
The pathname of the file to which the PC file is being transferred.

REPLACE?:
A YES or NO response, indicating whether or not the file specified for BUSINESS SYSTEM FILE NAME should be replaced if it already exists. If you respond NO, the specified file will not be replaced if it already exists on the Business System disk.

Example:

To transfer the file named B:EXAMPLE.ASM from the TI PC currently known as ST05 to the file named DATA2.EXAMPLE.FILE on the Business System to which the PC is connected, you could issue the following command.

```
[] TFFPC

TRANSFER FILE FROM PC
          TI PC FILE NAME: B:EXAMPLE.ASM
       TI PC FILE FORMAT: TEXT
      TI PC STATION NAME: ST05
BUSINESS SYSTEM FILE NAME: DATA2.EXAMPLE.FILE
                REPLACE?: NO
```

If you are using the TI PC connected to a Business System computer in a local area network, you can transfer a file from your PC to another Business System computer in the network. You can also transfer a file from another PC connected to a Business System computer to a file on a Business System computer. If you wanted to transfer B:EXAMPLE.ASM from the PC currently known as ST06 at the Business System computer with the site name of AUSTIN, you would issue the following command. The file would be moved to the Business System computer to which you are connected.

```
[] TFFPC

TRANSFER FILE FROM PC
          TI PC FILE NAME: B:EXAMPLE.ASM
       TI PC FILE FORMAT: TEXT
      TI PC STATION NAME: AUSTIN:ST06
BUSINESS SYSTEM FILE NAME: DATA2.EXAMPLE.FILE
                REPLACE?: NO
```

TFFPC (TRANSFER FILE FROM PC)

Notes:

When the TFFPC command is used, the command processor substitutes a sequence of eight blanks for any tab character it finds. To disable this option and receive the tab character, specify the SUBSTITUTE = NO prompt in expert mode when issuing the TFFPC command.

The TFFPC command can be used to send only MS-DOS files on the TI PC.

Do not confuse the MS-DOS disk drive name with the Business Systems sitename. Both are followed by colons.

Related Commands:

TFTPC (Transfer File to PC)

TFTPC (TRANSFER FILE TO PC)

4.220 TFTPC (TRANSFER FILE TO PC)

Prompts:

```
TRANSFER FILE TO PC
  BUSINESS SYSTEM FILE NAME:  [site:]filename@          (*)
          TI PC STATION NAME:  [site:][stationname]     (*)
              TI PC FILE NAME:  MS-DOS filename@        (*)
                     REPLACE?:  YES/NO                  (NO)
            TI PC FILE FORMAT:  {T/TEXT/B/BINARY}       (TEXT)
```

Purpose:

The TFTPC command allows you to move a file to a TI Professional Computer (PC) from a Business System computer. The command can be specified from any terminal or batch stream. The file transfer task works with a TI PC connected to a communications port on the Business System computer. The TI PC must be running the TI PC 931 Emulator package. The file transfer will not work from a TI PC directly connected to a local area network.

Any sequential or relative record ASCII or binary file can be transferred from a Business System computer to the PC. Any PC ASCII character can be transmitted.

A station that is issuing the command can be a different station than the PC to which the data is transferred. The PC specified as TI PC STATION NAME must not be logged on and must be using the TI PC 931 Emulator package.

Prompt Details:

BUSINESS SYSTEM FILE NAME:
The file name to be sent from the Business System computer to the TI PC.

TI PC STATION NAME:
The station name of the TI PC to which the file is to be sent. This is always a station name on a Business System computer to which the PC is connected or remotely logged on. The default value is your station ID.

TI PC FILE NAME:
The MS-DOS file name of the TI PC file into which the data is transferred.

REPLACE?:
A YES or NO response, indicating whether or not the file specified for TI PC FILE NAME should be replaced if it already exists. If you respond NO, the specified file will not be replaced if it already exists on the TI PC.

TFTPC (TRANSFER FILE TO PC)

TI PC FILE FORMAT:

The reply to this prompt defines the format of the file on the PC. If you respond TEXT, the file is in an editable MS-DOS file, and the MS-DOS record separator (>0D0A) is appended to the end of each record coming from the Business System computer. For International versions of PC using their national character set, national characters are translated from the ISO 7-bit code to the PC 8-bit code. If you respond BINARY, no record separator is appended to the records coming from Business System computer, and no translation is performed.

Example:

To transfer to the file named B:EXAMPLE.ASM on the TI PC currently known as ST05 from the file named DATA2.EXAMPLE.FILE on the Business System to which the PC is connected, you could issue the following command.

```
[] TFTPC

TRANSFER FILE TO PC
 BUSINESS SYSTEM FILE NAME: DATA2.EXAMPLE.FILE
         TI PC STATION NAME: ST05
            TI PC FILE NAME: B:EXAMPLE.ASM
                    REPLACE?: NO
          TI PC FILE FORMAT: TEXT
```

If you are using the TI PC connected to a Business System computer in a local area network, you can transfer a file to your PC from another Business System computer in the network. You can also transfer a file from another Business System computer to a file on another PC. If you wanted to transfer DATA2.EXAMPLE.FILE from your Business System computer to the PC known as ST06 at the Business System computer with the site name of AUSTIN, you would issue the following command.

```
[] TFTPC

TRANSFER FILE TO PC
 BUSINESS SYSTEM FILE NAME: DATA2.EXAMPLE.FILE
         TI PC STATION NAME: AUSTIN:ST06
            TI PC FILE NAME: B:EXAMPLE.ASM
                    REPLACE?: NO
          TI PC FILE FORMAT: TEXT
```

TFTPC (TRANSFER FILE TO PC)

Notes:

When the TFTPC command is used, the command processor substitutes eight blanks for a tab. To disable this option and send the tab character, specify the SUBSTITUTE = NO prompt in expert mode when issuing the TFTPC command.

The TPTPC command can be used to receive only MS-DOS files on the TI PC.

Do not confuse the MS-DOS disk drive name with the Business System site name. Both are followed by colons.

Related Commands:

TFFPC (Transfer File from PC)

TGS (TEST GENERATED SYSTEM)

4.221 TGS (TEST GENERATED SYSTEM)

Prompts:

```
TEST GENERATED SYSTEM
        TARGET DISK/VOLUME:  {devicename/volumename}@      (*)
            SYSTEM NAME:     alphanumeric                  (*)
```

Purpose:

Before the new operating system is installed as the primary system, test it using the TGS command. After testing is completed, the Install Generated System (IGS) command can be used to install the system.

The following procedure includes the steps to test and install a generated system:

1. Enter the TGS command to SCI, giving the parameters shown:

   ```
   [ ] TGS

   TEST GENERATED SYSTEM
           TARGET DISK/VOLUME:  devicename/volumename
               SYSTEM NAME:     alphanumeric
   ```

   where the response to the SYSTEM NAME prompt is the name assigned to the new system in response to the system generation OUTPUT CONFIGURATION prompt.

2. Perform an initial program load (IPL) on the new operating system. If the new system fails to operate, you can reload the old system using the same IPL process.

3. If the new system includes all of the devices specified during system generation, install it as the primary operating system by entering the IGS command:

   ```
   [ ] IGS

   INSTALL GENERATED SYSTEM
           TARGET DISK/VOLUME:  devicename/volumename
               SYSTEM NAME:     alphanumeric
   ```

4. If the new system is incorrect or unsatisfactory, perform an IPL on the original operating system software kit. The new system can be completely regenerated or changed using the system generation commands. Refer to the *DNOS System Generation Reference Manual* for a detailed description of generating or changing a new system.

TGS (TEST GENERATED SYSTEM)

Example:

In the following example, a new system has been generated and is contained in disk drive DS01. The name specified for the system is SYS1. The TGS command tests the new system with the following prompt responses:

```
[] TGS

TEST GENERATED SYSTEM
       TARGET DISK/VOLUME: DS01
              SYSTEM NAME: SYS1
```

If the system operates correctly after the IPL, the new system can be installed as the primary system using the IGS command as follows:

```
[] IGS

INSTALL GENERATED SYSTEM
       TARGET DISK/VOLUME: DS01
              SYSTEM NAME: SYS1
```

Related Commands:

ALGS	(Assemble and Link Generated System)
IGS	(Install Generated System)
PGS	(Patch Generated System)

4.222 UV (UNLOAD VOLUME)

Prompts:

UNLOAD VOLUME
VOLUME NAME: alphanumeric@ (*)

Purpose:

The UV command unloads disk volumes installed by the Install Volume (IV) command. You must remove an installed volume by entering a UV command to enable use of the disk unit by other users. You can use the Show Volume Status (SVS) command to find the volume name of the disk volume installed on a particular disk unit. After you unload the volume, a message is displayed indicating which disk unit was unloaded.

The UV command writes a message to the system log specifying the user ID, the station name, and volume name associated with the disk being unloaded.

Prompt Details:

VOLUME NAME:
The name of the installed volume that is being unloaded.

Example:

In the following example, the UV command unloads an installed volume named VOL1 which is loaded in disk unit DS03.

```
[] UV

UNLOAD VOLUME
        VOLUME NAME: VOL1
```

The following message is written to the terminal when the volume is successfully unloaded:

```
UNLOAD DISK DS03
```

Notes:

The DS10 and CD1400 Disk drives have removable and nonremovable platters turned by a single disk spindle. When the removable platter is being loaded or unloaded, you cannot access the nonremovable platter. Before changing the removable platter, ensure that neither platter is in use.

Related Commands:

CRV	(Check and Reset Volume)
INV	(Initialize New Volume)
IV	(Install Volume)

VB (VERIFY BACKUP)

4.223 VB (VERIFY BACKUP)

Prompts:

```
VERIFY BACKUP
    SEQUENTIAL ACCESS NAME:  [[site:]pathname/[site:]device name...
                             [site:]device name]@                    (*)
      DIRECTORY PATHNAME:    [pathname]@                             (*)
   CONTROL ACCESS NAME:      [site:][pathname]@
    LISTING ACCESS NAME:     [site:][pathname]@
                OPTIONS:     [character(s)...character(s)]
  EXECUTION MODE(F,B):       {F/FOREGROUND/B/BACKGROUND} (FOREGROUND)
```

Depending on your responses, you might receive one or more additional prompts.

```
PATHNAME OF BACKUP FILE
                PATHNAME:  pathname@

          DATE OF OLDEST FILE
                    YEAR:  integer
                   MONTH:  integer
                     DAY:  integer
                    HOUR:  [integer]                                 (0)
                  MINUTE:  [integer]                                 (0)
```

Purpose:

The VB command compares a directory and backup file to determine whether their contents match. VB checks for matches by comparing the files in the directory to the files in the backup file according to file name, type, and contents. You can use VB to verify the success of either a Backup Directory (BD) or Restore Directory (RD) command. VB can handle only backup files created by the BD command. To compare two directories, you can use the Verify Copy (VC) command.

The VB command supports the same options as BD for selecting files to include in the copy. Unless you specify otherwise, VB checks all the files (and aliases) in the directory, including any subdirectories and files they contain. VB does not check the system files .S$ROLLD.S$ROLLA, .S$DIAG, and .S$CRASH or the spooler directory .S$SDTQUE unless they are specifically included using an INCLUDE directive in a control file.

VB allows you to use a *control file* containing directives that identify the directory, backup file, and files involved in the operation. The OPTIONS, INCLUDE, and EXCLUDE directives in the VB (and RD) control file should match those in the BD control file. However, any MOVE directives should give the pathname of the backup file first, followed by the pathname of the directory—reversing the order from BD. For information on the control file directives, see the notes for BD.

VB (VERIFY BACKUP)

Prompt Details:

SEQUENTIAL ACCESS NAME:

The access name for the backup file. You can enter a sequential file pathname, a device name, or a list of device names. If you enter the device name of a disk drive (or a list of device names DS01 through DS99), VB displays an additional prompt for the pathname of the backup file. For details on multiple volume backups, refer to the special cases for BD and RD. You can skip this prompt only if you specify the sequential access name with a MOVE directive in a control file.

DIRECTORY PATHNAME:

The pathname of the directory backed up. Unless you limit the command with options or control file directives, the VB operation checks all files (and aliases) in this directory, including any subdirectories and files they contain. You can skip this prompt only if you specify the directory pathname with a MOVE directive in a control file.

CONTROL ACCESS NAME:

The access name of the control file. The control file consists of directives that determine which files in the directory participate in the operation. If you enter the pathname of a sequential file, VB uses the directives it finds there. If you enter ME, VB expects you to enter control file directives from your station. For details on control files, refer to the notes for BD. If you respond to both the SEQUENTIAL ACCESS NAME and DIRECTORY PATHNAME prompts, you do not need to specify a control file.

LISTING ACCESS NAME:

The device name or the file name where you want to receive listings of the backup operation. If you skip this prompt, VB assumes by default that you want the listings sent to the terminal local file of your station. If VB finds a file in the directory that does not match a file in the backup file, it notes the mismatch with a message in the listings next to the name of the file.

OPTIONS:

List of options to control the backup operation. Each option is expressed as a pair of alternatives, for example ALIAS and NOALIAS. You can choose one or the other, but not both. If you do not choose either, VB uses a default. Options specified in the control file override any options specified here. The options listed after ORIGINAL OPTIONS are the default options modified by the OPTIONS prompt. Options specified in a control file are not included here.

VB (VERIFY BACKUP)

Option	Purpose
ALIAS	(Default) Check aliases.
NOALIAS	Do not check aliases.
DATE	Check only files updated later than a specified date. If you choose this option, VB displays additional prompts for the date and time. If the DATE option is used in a control file, these additional date and time prompts must be answered. In batch or expert mode they can be answered directly. In interactive mode the DATE option must be included in the response to the OPTIONS prompt in order to answer the additional date and time prompts. If either the last update date or the creation date is later than the specified date, the file is checked.
NODATE	(Default) Check files regardless of their dates.
REWIND	Rewind the tape before performing the verify operation.
NOREWIND	(Default) Do not rewind the tape before performing the verify operation.
UNLOAD	Rewind and unload the reel after the verify operation.
NOUNLOAD	(Default) Do not rewind and unload the reel after the verify operation.
SYSFILES	(Default) Include the following system files and directories in a verify operation on the system disk: .SCI990 .S$SYSLIB .S$CDT .S$ISBTCH .S$CMDS .S$ISLIST .S$EXPMSG .S$LANG .S$MSG .S$PWCS .S$SGU$ .S$SHARED .S$SYSTEM .S$SHIP .S$IPL .S$UTIL
NOSYSFILE	Do not include the system files in the verify operation.
FAST	Verify a program file all at once. If you choose this option, you need to make sure the disk that contains the directory and the disk where you later restore it have the same sector size.
NOFAST	(Default) Verify each task, procedure, program segment, and overlay image individually.

VB (VERIFY BACKUP)

EXECUTION MODE(F,B):
Choice of whether to carry out the command in the foreground or background. If you enter F or FOREGROUND, the command executes in the foreground. If you enter B or BACKGROUND, the command executes in the background. Though the default is FOREGROUND, in a batch stream the VB command always executes in the background.

PATHNAME:
The pathname of the backup file. This prompt appears only when you give a device name of a disk unit (DS01 through DS99) in response to the sequential access name prompt. The backup file has to be a sequential file, but it can span several disks. For details, refer to the special cases concerning multiple volume backup for BD and RD.

YEAR:
The two- or four-digit number for a year. This prompt appears only when you choose the DATE option. Your response becomes the year portion of the cutoff date used by the DATE option to determine which files to verify.

MONTH:
The one- or two-digit number of a month. This prompt appears only when you choose the DATE option. Your response becomes the month portion of the cutoff date used by the DATE option to determine which files to verify.

DAY:
The one- or two-digit number of a day of the month. This prompt appears only when you choose the DATE option. Your response becomes the day portion of the cutoff date used by the DATE option to determine which files to verify.

HOUR:
The hour on the 24-hour clock. This prompt appears only when you choose the DATE option. Your response becomes the hour portion of the cutoff time used by the DATE option to determine which files to verify. If you skip this prompt, VB assumes 0 (the initial value).

MINUTE:
The minute of the hour. This prompt appears only when you choose the DATE option. Your response becomes the minute portion of the cutoff time used by the DATE option to determine which files to verify. If you skip this prompt, VB assumes 0 (the initial value).

Examples:

Since you only use VB after a prior BD or RD command, the VB examples are included with the examples for BD and RD.

VB (VERIFY BACKUP)

Messages:

PREVIOUS TASK NOT VERIFIABLE
A disk error occurred during the backup of this program file. The previous listed task was flagged as not verified.

< filename> WAS BACKED UP WITH ERRORS-CANNOT BE VERIFIED
A disk error occurred during the backup of the file specified. This file was not verified and cannot be restored.

Notes:

If a hardware error occurs during the backup operation, you receive a message in the listing file stating that the file cannot be verified. Subsequent VB (and RD) commands do not consider these files. Instead they list them as not verified due to disk errors during the backup process.

When VB is used in a file security system, you must have read access rights to all files that are to be verified.

Related Commands:

BD	(Backup Directory)
RD	(Restore Directory)
VC	(Verify Copy)

4.224 VC (VERIFY COPY)

Prompts:

```
VERIFY COPY
                MASTER PATHNAME:  [pathname]@                          (*)
                  COPY PATHNAME:  [pathname]@                          (*)
            CONTROL ACCESS NAME:  [site:][pathname]@
             LISTING ACCESS NAME: [site:][pathname]@
                         OPTIONS: [character(s)...character(s)]
          EXECUTION MODE(F,B):    {F/FOREGROUND/B/BACKGROUND} (FOREGROUND)
```

Depending on your response to the OPTION prompt, you might receive additional prompts:

```
DATE OF OLDEST FILE
                        YEAR:  integer
                       MONTH:  integer
                         DAY:  integer
                        HOUR:  [integer]                               (0)
                      MINUTE:  [integer]                               (0)

DESTINATION PHYSICAL RECORD LENGTH
PHYSICAL RECORD LENGTH:        integer                                 (768)
```

Purpose:

The VC command compares the files in two directories, typically a master directory and a copy made by the Copy Directory (CD) command. If you want to compare a directory with a backup file created by the Backup Directory (BD) command, you should use the Verify Backup (VB) command.

The VC command supports all the CD options that let you select the files included in the copy. In its listings, VC reports any files in the master directory that do not have the same name, type, and contents in the copy.

Unless you tell it otherwise, VC considers all files (and aliases) in the master directory, including any subdirectories and files they contain. VC does not verify the system files .S$ROLLD.S$ROLLA, .S$DIAG, and .S$CRASH or the spooler directory .S$SDTQUE unless they are specifically included using an INCLUDE directive in a control file.

Like CD, VC allows you to use a *control file* containing directives that identify the master directory, copy directory, and the files involved in the operation. For information about the control file directives, see the notes for BD.

VC (VERIFY COPY)

Prompt Details:

MASTER PATHNAME:
The pathname of the directory that contains the originals of the files copied. Unless you limit the command with options or control file directives, VC verifies all files (and aliases) in this directory—including any subdirectories and files they contain. You can skip this prompt only if you specify the master pathname with a MOVE directive in a control file.

COPY PATHNAME:
The pathname of the directory that received the copied files. You can skip this prompt only if you specify the copy pathname with a MOVE directive in a control file.

CONTROL ACCESS NAME:
The access name of the control file. If you enter a pathname of a sequential file, VC uses the directives it finds there. If you enter ME, VC expects you to enter control file directives from your station. For details on control files, refer to the notes for BD. If you respond to both the MASTER PATHNAME and COPY PATHNAME prompts, you do not need to specify a control file.

LISTING ACCESS NAME:
The device name or file name where you want to receive listings of the verify operation. If you skip this prompt, VC assumes by default that you want the listings sent to the terminal local file of your station. If the VC command finds an error in copying a file, it notes the mismatch in the listings.

OPTIONS:
List of options to control the verify operation. Each option is expressed as a pair of alternatives, for example ALIAS and NOALIAS. You can choose one or the other, but not both. If you do not choose either, VC uses a default. Options specified override any options specified here in the control file. The options listed after ORIGINAL OPTIONS are the default options modified by the response to the OPTIONS prompt. Options specified by a control file are not included here.

VC (VERIFY COPY)

Option	Purpose
ADD	Valid choice, but ignored by VC.
REPLACE	Valid choice, but ignored by VC.
ALIAS	(Default) Verify aliases.
NOALIAS	Do not verify aliases.
DATE	Verify only files updated later than a specified date. If you choose this option, VC displays additional prompts for the date and time. If the DATE option is used in a control file, these additional date and time prompts must be answered. In batch or expert mode they can be answered directly. In interactive mode the DATE option must be included in the response to the OPTIONS prompt in order to answer the additional date and time prompts. If either the last update date or the creation date is later than the specified date, the file is verified.
NODATE	(Default) Verify files regardless of their dates.
RPRL	Verify that relative record files in the copy directory have a specified physical record length. If you choose this option with CD, you must also choose it for the subsequent VC.
NORPRL	(Default) Verify that relative record files in the copy directory have the same physical record length as in the master directory.
SPRL	Verify that sequential files in the copy directory have a specified physical record length. If you choose this option with CD, you must also choose it for the subsequent VC.
NOSPRL	(Default) Verify that sequential files in the copy directory have the same physical record length as in the master directory.
SYSFILES	(Default) Include the following system files and directories in a verify operation on the system disk: .SCI990 .S$SYSLIB .S$CDT .S$ISBTCH .S$CMDS .S$ISLIST .S$EXPMSG .S$LANG .S$MSG .S$PWCS .S$SGU$ .S$SHARED .S$SYSTEM .S$SHIP .S$IPL .S$UTIL
NOSYSFILE	Do not include the system files in the verification.

VC (VERIFY COPY)

EXECUTION MODE(F,B):

Choice of whether to carry out the command in the foreground or background. If you enter F or FOREGROUND, the command executes in the foreground. If you enter B or BACKGROUND, the command executes in the background. Though the default is FOREGROUND, in a batch stream the VC command always executes in the background.

YEAR:

The two- or four-digit number for a year. This prompt appears only when you choose the DATE option. Your response becomes the year portion of the cutoff date used by the DATE option to determine which files to verify.

MONTH:

The one- or two-digit number of a month. This prompt appears only when you choose the DATE option. Your response becomes the month portion of the cutoff date used by the DATE option to determine which files to verify.

DAY:

The one- or two-digit number of a day of the month. This prompt appears only when you choose the DATE option. Your response becomes the day portion of the cutoff date used by the DATE option to determine which files to verify.

HOUR:

The hour on the 24-hour clock. This prompt appears only when you choose the DATE option. Your response becomes the hour portion of the cutoff time used by the DATE option to determine which files to verify. If you skip this prompt, VC assumes 0 (the initial value).

MINUTE:

The minute of the hour. This prompt appears only when you choose the DATE option. Your response becomes the minute portion of the cutoff time used by the DATE option to determine which files to verify. If you skip this prompt, VC assumes 0 (the initial value).

PHYSICAL RECORD LENGTH:

The physical record length in bytes. This prompt appears only when you choose the RPRL or SPRL options.

Examples:

Verifying the Results of a CD Command Without a Control File

The example for CD shows how to copy the input directory VOL2.DIR1 to the output directory VOL2.DIR2. The input directory consists of the files FILE1, FILE2, FILE3, and PROGRAMS (a program file that contains the modules PROGA and PROGB). The following VC command checks the result of the copy operation.

```
[] VC

   VERIFY COPY
            MASTER PATHNAME: VOL2.DIR1
              COPY PATHNAME: VOL2.DIR2
       CONTROL ACCESS NAME:
       LISTING ACCESS NAME: LP
                   OPTIONS: ADD
       EXECUTION MODE(F,B): FOREGROUND
```

VC produces the following listings:

```
  VERIFY COPY              14:28:17 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:       VOL2.DIR1
ORIGINAL DESTINATION: VOL2.DIR2
ORIGINAL OPTIONS:      ALIAS, NODATE, ADD, NOSPRL, NORPRL, SYSFILES
CONTROL FILE:
LIST FILE:             LP


  **        DIRECTORY  VOL2.DIR1
  **   FILE2
  **   FILE3
  **   PROGRAMS - PROGRAM FILE

  **   TASK        ID
  **   PROGA      >03
  **   PROGB      >0E

  **   PROCEDURE  ID

  **   OVERLAY    ID
  **        END OF PROGRAM FILE

  **   FILE1

ELAPSED TIME = 0 MINUTES   58 SECONDS
SIZE OF INPUT = 439 ADU'S

****** VERIFY COPY COMPLETED
```

VC (VERIFY COPY)

Verifying the Results of a CD Command Using a Control File

The example for CD uses the control file VOL2.CONTROL to copy files from the input directories VOL2.DIR1 and VOL2.DIR3 to the output directory VOL2.DIR2. The control file for a VC command to check the results of the copy operation uses the same control file as the prior CD command.

```
OPTION REP
MOVE VOL2.DIR1
EX FILE1,FILE2
OPTION ADD
MOVE VOL2.DIR3
IN FILEA
IN FILEB
END

[] VC

VERIFY COPY
        MASTER PATHNAME:
          COPY PATHNAME: VOL2.DIR2
    CONTROL ACCESS NAME: VOL2.CONTROL
    LISTING ACCESS NAME: LP
                OPTIONS: ADD
    EXECUTION MODE(F,B): FOREGROUND
```

VC (VERIFY COPY)

VC produces the following listings:

```
  VERIFY COPY              14:26:08 TUESDAY, OCTOBER 26, 1982.

ORIGINAL SOURCE:
ORIGINAL DESTINATION: VOL2.DIR2
ORIGINAL OPTIONS:     ALIAS,NODATE,ADD,NOSPRL,NORPRL,SYSFILES
CONTROL FILE:         VOL2.CONTROL
LIST FILE:            LP

OPTION REP
MOVE VOL2.DIR1
EX FILE1,FILE2

  **       DIRECTORY  VOL2.DIR1
  **  FILE3
  **  PROGRAMS - PROGRAM FILE

  **  TASK        ID
  **  PROGA      >03
  **  PROGB      >0E

  **  PROCEDURE  ID

  **  OVERLAY    ID
  **       END OF PROGRAM FILE

OPTION ADD
MOVE VOL2.DIR3
IN FILEA
IN FILEB

  **       DIRECTORY  VOL2.DIR3
  **  FILEB
  **  FILEA
END

ELAPSED TIME = 0 MINUTES   59 SECONDS
SIZE OF INPUT = 457 ADU'S

****** VERIFY COPY COMPLETED
```

Notes:

When VC is used in a file security system, you must have read access rights to all files that are to be verified.

Related Commands:

CD	(Copy Directory)
VB	(Verify Backup)

WAIT (WAIT FOR BACKGROUND TASK TO COMPLETE)

4.225 WAIT (WAIT FOR BACKGROUND TASK TO COMPLETE)

Purpose:

The WAIT command provides background reporting when no foreground activity is desired without constantly inputting the Show Background Status (SBS) command. Background command termination messages are ordinarily not displayed except after termination of foreground commands or when requested via the SBS command. The WAIT command displays the background terminal local file if it exists when the background task completes.

The WAIT command displays the following message until the background task terminates:

— WAITING FOR BACKGROUND TASK TO COMPLETE —

To abort the WAIT command and continue foreground processing, press the Command key. A message reporting the current state of the task is printed at the terminal.

Example:

In the following example, the WAIT command reports the background status on the terminal:

```
[] WAIT
```

The following message is then displayed on your terminal until the background task completes:

— WAITING FOR BACKGROUND TASK TO COMPLETE —

Notes:

When there is no background activity, the WAIT command may return to the main menu or it may display the message, THERE IS NO BACKGROUND ACTIVITY AT THIS TERMINAL.

Related Commands:

KBT	(Kill Background Task)
SBS	(Show Background Status)

WCRU (WRITE VALUE TO SPECIFIED CRU ADDRESS)

4.226 WCRU (WRITE VALUE TO SPECIFIED CRU ADDRESS)

Prompts:

WRITE VALUE TO SPECIFIED CRU ADDRESS
CRU ADDRESS: integer
NUMBER OF BITS: integer
VALUE TO BE WRITTEN: integer

Purpose:

The WCRU command writes a specified number of bits in the specified CRU address.

Prompt Details:

CRU ADDRESS:
The CRU address to which to write the specified number of bits. The CRU address must be less than or equal to >1FFE and not in the range of >1EE1 through >1FDF.

NUMBER OF BITS:
Number of bits to be written. If the value is 0, 16 bits are written; otherwise, this value must be less than or equal to 15.

VALUE TO BE WRITTEN:
An integer value to be written in the specified CRU address.

Example:

In the following example, the WCRU command writes 16 bits of information to a specified CRU address:

```
[] WCRU

WRITE VALUE TO SPECIFIED CRU ADDRESS
            CRU ADDRESS: >0100
         NUMBER OF BITS: 0
     VALUE TO BE WRITTEN: >4141
```

Related Commands:

RCRU (Read Contents of Specified CRU Address)

WEOF (WRITE EOF TO LUNO)

4.227 WEOF (WRITE EOF TO LUNO)

Prompts:

```
WRITE EOF TO LUNO
                    LUNO:   integer
```

Purpose:

The operating system accesses information on sequential files and devices on a record-by-record basis. You can write an EOF marker at the current data-access position by entering a WEOF command.

Prompt Details:

LUNO:
The LUNO assigned to the sequential device or file to which the EOF is to be written.

Example:

In the following example, LUNO >32 has been assigned to a magnetic tape unit. An EOF is written to that magnetic tape unit through the WEOF command as shown below:

```
[] WEOF

WRITE EOF TO LUNO
                    LUNO: >32
```

Notes:

The operations performed by WEOF include an open call and a close with write EOF call to the file or device.

XANAL (EXECUTE CRASH ANAYLSIS UTILITY)

4.228 XANAL (EXECUTE CRASH ANALYSIS UTILITY)

Prompts:

```
EXECUTE CRASH ANALYSIS UTILITY
        CONTROL ACCESS NAME:  [site:]pathname@        (ME)
        LISTING ACCESS NAME:  [site:][pathname]@      (ME)
ANALYZE RUNNING SYSTEM?:      YES/NO                  (NO)
            CRASH FILE NAME:  [site:]pathname@        (.S$CRASH)
```

Purpose:

The XANAL command executes the crash dump analyzer and creates a formatted printout of the system crash file. A knowledgeable system programmer can use this printout to detect and correct system logic errors.

When the operating system detects a system failure, it displays an error code on the front panel lights and idles the CPU. The first step that you must take in order to analyze the system crash is to dump memory to the predefined crash file on the disk by pressing HALT and then RUN on the front panel. The next steps in analyzing the crash are: perform the initial program load (IPL) on the front panel of the computer, bid SCI at a terminal, and execute the crash analyzer ANALZ by executing the XANAL command. Once XANAL is executing, you can select from several commands to obtain information about the crash. Table 4-25 lists the XANAL commands available to you and the information that can be obtained by their execution.

Refer to the *DNOS Systems Programmer's Guide* or the *DNOS System Design Document* for further information concerning the XANAL command and the system crash analyzer ANALZ.

Prompt Details:

CONTROL ACCESS NAME:
The pathname from which ANALZ reads the command stream.

LISTING ACCESS NAME:
The pathname to which ANALZ outputs a listing. A null response specifies ME.

ANALYZE RUNNING SYSTEM?:
If you specify YES, the system in memory is analyzed. If you specify NO, the system image in the crash file is analyzed.

CRASH FILE NAME:
The pathname of the crash file.

XANAL (EXECUTE CRASH ANAYLSIS UTILITY)

Example:

In the following example, the XANAL command executes the crash file analyzer to analyze the system image crash file of the system disk. The analyze commands listed in Table 4-25 are entered at your terminal with the results of the commands displayed at the terminal:

```
[] XANAL

EXECUTE CRASH ANALYSIS UTILITY
     CONTROL ACCESS NAME: ME
     LISTING ACCESS NAME: ME
 ANALYZE RUNNING SYSTEM?: NO
         CRASH FILE NAME: .S$CRASH
```

Table 4-25. XANAL Commands

Command	Action
ALL	Executes all of the commands
AQ	Displays contents of active queue
CCB	Dumps channel control blocks
DM	Dumps a specific area of memory
FCB	Dumps file control blocks
GI	Displays general information
JSB	Dumps job status blocks
LDT	Lists logical device tables
MM	Lists memory maps
OVB	Lists overhead beets
PBM	Lists partial bit maps
PDT	Dumps physical device tables
PQ	Displays all other system queues
Q or QU	Terminates execution of the XANAL command
ROB	Dumps resource ownership blocks
RPB	Displays resource privilege blocks
SGB	Dumps segment group blocks
SSB	Dumps segment status blocks
ST	Dumps secondary table areas
TA	Dumps task areas of tasks currently in memory
TR	Dumps requests for all tasks
TS	Displays task status
TSB	Dumps task status blocks
??	Lists all of the commands

4.229 XB (EXECUTE BATCH)

Prompts:

```
EXECUTE BATCH
          INPUT ACCESS NAME:   [site:]pathname@
      LISTING ACCESS NAME:     [site:]pathname@
```

Purpose:

The XB command activates SCI in background mode. SCI runs without interacting with the terminal until finished, at which time it sends a message to the terminal that initiated the XB command that the background task has completed. A batch command stream can be created by means of text editing or by the Create Batch Stream (CB) command.

Prompt Details:

INPUT ACCESS NAME:
The pathname from which SCI should read the batch command stream.

LISTING ACCESS NAME:
The pathname of a device or file to which SCI should write the listings of the batch command stream execution. This device or file must not be used by any command in the batch command stream.

Example:

In the following example, the XB command executes a batch stream from a file and outputs the results of the batch command stream to the file MY.LIST.

```
[] XB

EXECUTE BATCH
       INPUT ACCESS NAME: MY.BATCH
    LISTING ACCESS NAME: MY.LIST
```

Notes:

When the XB command begins executing, you can enter the WAIT command in response to the SCI prompt ([ ]) (if you do not want foreground activity), or press the Return or Command key from time to time (if SCI is in foreground command mode) to display the batch termination message when the batch stream execution completes.

You can also enter the Show Background Status (SBS) command through foreground SCI to determine the current status of background execution, or the Show File (SF) command to inspect the listing file.

Related Commands:

XBJ (Execute Batch SCI Job)

XBJ (EXECUTE BATCH SCI JOB)

4.230 XBJ (EXECUTE BATCH SCI JOB)

Prompts:

```
EXECUTE BATCH SCI JOB
                    SITE NAME:  [site]@
                    JOB NAME:   alphanumeric
    SYNONYM TABLE PATHNAME:     [site:][filename]@
```

If a site name is not supplied, the following prompt is displayed:

```
EXECUTE BATCH SCI JOB
        USE CURRENT USER ID?:   YES/NO                  (YES)
```

If a site name is supplied or if NO is the response to the USE CURRENT USER ID? prompt, the following prompts are displayed:

```
EXECUTE BATCH SCI JOB
                    USER ID:    alphanumeric
                   PASSCODE:    -[character(s)]
                 ACCOUNT ID:    [character(s)]
```

If your response to the SYNONYM TABLE PATHNAME prompt is null, the following prompts are displayed:

```
EXECUTE BATCH SCI JOB
              INPUT ACCESS NAME:    [site:]pathname@
            LISTING ACCESS NAME:    [site:]pathname@
```

Purpose:

The XBJ command creates a batch SCI job with operating parameters that can be different from those of the creating job.

Prompt Details:

SITE NAME:
A valid site name. Specify the site name of the computer at which the batch SCI job is to execute. If a null response is entered, the job is executed on the computer from which the XBJ command is entered.

JOB NAME:
A one- to eight-character string to reference the job.

USE CURRENT USER ID:
If you specify YES, the current user ID is used. If NO, you must specify a new user ID. This prompt is displayed only if a site name is not specified.

XBJ (EXECUTE BATCH SCI JOB)

USER ID:
The user ID to be associated with the new job. A response to this prompt is required if the response to the USE CURRENT USER ID? prompt was NO.

PASSCODE:
The passcode corresponding to the user ID of the new job.

ACCOUNT ID:
A 1- to 16-character string that is the account ID for the new job. An account ID is not specified for the new job if you respond with a null response.

SYNONYM TABLE PATHNAME:
The file name containing the set of synonyms and logical names to be used by the new job. You must create the file via the Snapshot Name Definition (SND) command and include the new job's input and listing access names in the SND command's parameter list. If you enter a null response, the INPUT ACCESS NAME and LISTING ACCESS NAME prompts are displayed at your terminal.

INPUT ACCESS NAME:
The pathname of a device or file where the job command stream resides.

LISTING ACCESS NAME:
The pathname of a device or file where the job execution results are to be listed.

Example:

In the following example, a batch job is created that creates a file, outputs data to the file, prints the file contents, then deletes the file. The command stream to perform these functions resides in a file named SYS1.KC0017.INPUT, and the logical names and synonyms of the creating job are passed to the new job. The XBJ command can be used to create and execute the batch job as shown below:

```
[] XBJ

EXECUTE BATCH SCI JOB
                    SITE NAME:
                     JOB NAME: BATCH
      SYNONYM TABLE PATHNAME:

EXECUTE BATCH SCI JOB
        USE CURRENT USER ID?: YES

EXECUTE BATCH SCI JOB
           INPUT ACCESS NAME: SYS1.KC0017.INPUT
         LISTING ACCESS NAME: LP01
```

XBJ (EXECUTE BATCH SCI JOB)

Messages:

US SCI-0051 BATCH SCI WAS UNABLE TO OPEN THE INPUT FILE
The input file for the Batch SCI Job does not exist. This message is displayed in the system log file and in your listing file.

US SCI-0050 BATCH SCI WAS UNABLE TO OPEN THE LISTING FILE
The listing file cannot be opened. This message is displayed in the system log file.

SCI will display the job ID generated by the Job Manager on your terminal. If the response to the SYNONYM TABLE PATHNAME prompt is DUMY, XBJ displays:

*** SYNONYM TABLE NAME REQUIRED ***

Normal termination messages are not recorded in the system log file or in your listing file.

Notes:

For the SYNONYM TABLE PATHNAME prompt, you must enter a null response or supply a file name containing synonyms and logical names produced by the SND command.

A job executed through the XBJ command is not associated with a terminal; therefore, SCI sets synonym $$ST to zero and synonym ME to a null value.

Four additional prompts, JCA SIZE, DISPLAY, PRIORITY, and BACKGROUND are not displayed in interactive mode but can be specified in interactive expert mode or batch mode. Your response to the JCA SIZE prompt controls the size of the job communication area (JCA) for the new job. The value of this prompt can be SMALL, MEDIUM, or LARGE. The default value is MEDIUM. The actual usage of the JCA by the new job is available through the accounting system at job termination.

The hidden prompt, DISPLAY, can be used to control whether or not the job ID is displayed on your terminal. If you respond NO to this prompt, the job ID is not displayed. The default is YES.

The hidden prompt, PRIORITY, can be used to control the priority of the job. The priority values range from 1 to 31, with 1 being the highest priority. The default value is 30.

The hidden prompt, BACKGROUND, can be used to specify that the initial task in the job, SCI, not be a background task. The acceptable responses are YES or NO. A NO response specifies that the task not be bid as a background task and, therefore, receive a higher run priority. YES is the default response.

XBJ (EXECUTE BATCH SCI JOB)

Related Commands:

HJ	(Halt Job)
KJ	(Kill Job)
LJ	(List Jobs)
MJP	(Modify Job Priority)
RJ	(Resume Job)
SJS	(Show Job Status for < user ID>)
SND	(Snapshot Name Definitions)
XJ	(Execute Job)

XD (EXECUTE IN DEBUG MODE)

4.231 XD (EXECUTE IN DEBUG MODE)

Prompts:

```
EXECUTE IN DEBUG MODE
                        RUN ID:  [integer]          (*)
SYMBOL TABLE OBJECT FILE:  [filename]@              (*)
          990/12 OBJECT CODE?:  YES/NO              (YES)
```

Purpose:

The XD command places a specified task into controlled mode. The run-time ID is optional but cannot be the ID of a system task. If you do not specify a run-time ID, an automatic call is made to the Execute and Halt Task (XHT) command to place the task into execution.

The symbol table object file is optional and its presence determines whether symbolic expressions are allowed on any of the subsequent debug commands. If you specify a symbol table to the Link Editor (SYMT option has been selected) and if the controlled task symbol table object file is specified, then symbolic expressions involving symbols in the object code symbol table can be used in commands that allow full expressions.

The debugger can be used to verify that code developed on a 990/12 will execute successfully on a 990/10; that is, it verifies that the code contains no special 990/12 instructions.

Only one task for each station can be in debug mode at a given time.

Detailed examples of using the debugger can be found in the *DNOS Assembly Language Programmer's Guide*.

Prompt Details:

RUN ID:
A valid run ID in your job. Obtain current run IDs by executing the Show Task Status (STS) command.

If you specify a null response, the prompts for the XHT command are displayed. Refer to the XHT command for information concerning responses to these prompts.

SYMBOL TABLE OBJECT FILE:
The file name specified to the Link Editor if the SYMT option has been selected. By specifying this file name in response to the prompt, you are allowed to use symbolic expressions which involve symbols in the object code symbol table on any debug command prompt which calls for a character(s) response. If you enter a null response, no symbol table file is used and symbolic expressions are not allowed.

990/12 OBJECT CODE?:
If you enter YES in response to this prompt, the debugger allows 990/12 special instructions if you are executing on a 990/12 computer. If you enter NO, the debugger returns an illegal opcode warning for 990/12 special instructions if you are executing on either a 990/10 or a 990/12 computer.

XD (EXECUTE IN DEBUG MODE)

Example:

In the following example, the XD command places the task with run ID >9A into controlled mode. An object code symbol table is specified, and the object code simulated is for a 990/10 computer.

```
[] XD

  EXECUTE IN DEBUG MODE
                          RUN ID:   >9A
    SYMBOL TABLE OBJECT FILE:   .OBJ.PROG
          990/12 OBJECT CODE:   NO
```

XE (EXECUTE TEXT EDITOR)

4.232 XE (EXECUTE TEXT EDITOR)

The XE command activates the DNOS Text Editor. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the XE command and the following prompts:

```
EXECUTE TEXT EDITOR
          FILE ACCESS NAME:  [site:][filename]@     (*)
            EXCLUSIVE EDIT?: YES/NO                 (YES)
               LINE LENGTH:  integer                (80)
```

XES (EXECUTE TEXT EDITOR WITH SCALING)

4.233 XES (EXECUTE TEXT EDITOR WITH SCALING)

The XES command activates the Text Editor at a VDT that is in VDT mode and displays a calibrated scale indicating column numbers, on line 24 of the screen display. Refer to the *DNOS Text Editor Reference Manual* for a detailed description of the XES command and the following prompts:

```
EXECUTE TEXT EDITOR WITH SCALING
        FILE ACCESS NAME:  [site:][filename]@      (*)
          EXCLUSIVE EDIT?: YES/NO                  (YES)
             LINE LENGTH:  integer                 (80)
```

XHT (EXECUTE AND HALT TASK)

4.234 XHT (EXECUTE AND HALT TASK)

Prompts:

```
EXECUTE AND HALT TASK
        PROGRAM FILE OR LUNO:  {filename/integer}@      (*)
            TASK NAME OR ID:   {alphanumeric/integer}   (*)
                      PARM1:   integer                  (0)
                      PARM2:   integer                  (0)
                 STATION ID:   {integer/ME}             (*)
```

Purpose:

The XHT command places a task in memory in a suspended state so it can be debugged. Typically, you place the task to be debugged in memory using XHT, establish the debug environment (including breakpoints), and then activate the task using the Resume Task (RT) command.

Prompt Details:

PROGRAM FILE OR LUNO:
The file name of, or the LUNO assigned to, the program file on which the task has been installed. If you specify a LUNO in response to this prompt, it must be assigned prior to the execution of the XHT command. If you specify zero, the .S$SHARED program file is used. If you specify > FF, the utility program file is assumed.

TASK NAME OR ID:
The name or the associated installed ID of the task to be halted.

PARM1:
An integer value to be passed to the task being halted, determined by the programmer who wrote the task.

PARM2:
A second integer value to be passed to the task being halted, determined by the programmer who wrote the task.

STATION ID:
The station ID (for example, 1, 2) with which the task is to be associated or the two-character pseudo device name of ME. If you enter > FF, the task is not associated with any station. If you enter ME or 0, the task is associated with the station at which the command is entered.

XHT (EXECUTE AND HALT TASK)

Example:

In the following example, the XHT command executes a task on a user"s program file, then the task is placed in memory to be debugged. The task is associated with the terminal where the XHT command is executed.

```
[] XHT

EXECUTE AND HALT TASK
     PROGRAM FILE OR LUNO: .KC0017.PROG
          TASK NAME OR ID: PRINT
                     PARM1: 0
                     PARM2: 0
                STATION ID: ME
```

Notes:

The hidden prompt, DISPLAY, is not displayed in interactive mode but can be specified in interactive expert mode or batch mode. The DISPLAY prompt can be used to control whether or not the task ID is displayed on your terminal. IF you respond NO to this prompt, the task ID is not displayed. The default is YES.

Related Commands:

RT	(Resume Task)
XT	(Execute Task)
XTS	(Execute Task and Suspend SCI)

XJ (EXECUTE JOB)

4.235 XJ (EXECUTE JOB)

Prompts:

```
EXECUTE JOB
                        SITE NAME:  [site]@
                        JOB NAME:  alphanumeric
              PROGRAM FILE PATHNAME:  [site:][filename]@
                 TASK ID OR NAME:  integer/alphanumeric
                           PARM1:  integer                    (0)
                           PARM2:  integer                    (0)
                      STATION ID:  [integer]
          SYNONYM TABLE PATHNAME:  [site:][filename]@
                        PRIORITY:  integer                    (20)
                        JCA SIZE:  {SMALL/MEDIUM/LARGE}       (MEDIUM)
```

If a site name is not specified or if your local site name is specified, the following prompt is displayed:

```
EXECUTE JOB
        USE CURRENT USER ID?:  YES/NO                         (YES)
```

If a site name other than your local site name is supplied or if NO is the response to the USE CURRENT USER ID? prompt, the following prompts are displayed:

```
EXECUTE JOB
                          USER ID:  alphanumeric
                        PASSCODE:  -[character(s)]
                      ACCOUNT ID:  [character(s)]
```

Purpose:

The XJ command interactively creates a job with operating parameters that can be different from those of the creating job.

Prompt Details:

SITE NAME:
A valid site name. Specify the site name of the computer at which the batch SCI job is to execute. If a null response is entered, the job is executed on the computer from which the XJ command is entered.

JOB NAME:
A one- to eight-character, user-defined string assigned by the user to reference the new job.

XJ (EXECUTE JOB)

PROGRAM FILE PATHNAME:
The program file containing the initial task. The program file must be located at the site at which the job is going to execute. The default program file is .S$SHARED at the specified site.

TASK ID OR NAME:
The installed ID or task name of the initial task to execute after the job is created.

PARM1:
A decimal or hexadecimal number in the range of -32768 through 65535 passed initial task. How the number is used depends on how the job is written.

PARM2:
A decimal or hexadecimal number in the range of -32768 through 65535 passed initial task. How the number is used depends on how the job is written.

STATION ID:
The two-digit ID number of the station associated with the initial task. The value of > FF or null specifies that the job is not associated with any terminal.

SYNONYM TABLE PATHNAME:
The pathname of a file containing the set of synonyms and logical names to be used by the new job. A null response passes a snapshot copy of the creating job's synonyms and logical names.

If you enter DUMY, no synonyms or logical names are passed. Any other response is assumed to be the pathname of a file established by the Snapshot Name Definitions (SND) command holding synonyms, logical names, and parameters for the new job.

PRIORITY:
The priority attached to the new job. Priorities range from 1 through 31, with 1 being the highest priority.

JCA SIZE:
The size of the job communication area (JCA) for the new job. The actual usage of the JCA by the new job is available through the accounting system at job termination.

USE CURRENT USER ID?:
If you specify YES, the current user ID is used. If you specify NO, you must specify a new user ID.

USER ID:
The user ID to be associated with the new job.

PASSCODE:
The passcode corresponding to the user ID for the new job.

ACCOUNT ID:
The account ID for the new job. A null response specifies that no account ID is specified with the job.

XJ (EXECUTE JOB)

Example:

In the following example, a job is created to pass parameters to an initial task to calculate the amount of income tax a person should have paid for the preceding year. The initial task performs the calculation, and the newly created job passes the gross annual income and the number of claimed dependents as parameters. The job name uses the employee number to distinguish one calculation from another. The job is created by executing the XJ command as shown below:

```
[] XJ

EXECUTE JOB
                    SITE NAME:
                     JOB NAME: E802378
       PROGRAM FILE PATHNAME: SYS2.USER.PROGFIL
             TASK ID OR NAME: >3B
                        PARM1: 25000
                        PARM2: 3
                  STATION ID:
      SYNONYM TABLE PATHNAME: DS02.USER.SYN
                    PRIORITY: 20
                    JCA SIZE: MEDIUM

EXECUTE JOB
        USE CURRENT USER ID?: YES
```

Messages:

Upon successful completion of the XJ command, SCI displays the job ID generated by the Job Manager.

Notes:

The USE CURRENT USER ID? prompt is displayed on a new screen at the terminal. The set of prompts containing USER ID, PASSCODE, and ACCOUNT ID is also displayed on a new screen at the terminal.

One additional prompt, DISPLAY, is not displayed in interactive mode but can be specified in interactive expert mode or batch mode. The hidden prompt, DISPLAY, can be used to control whether or not the job ID is displayed on your terminal. If you respond NO to this prompt, the job ID is not displayed. The default is YES.

Related Commands:

HJ	(Halt Job)
KJ	(Kill Job)
LJ	(List Jobs)
MJP	(Modify Job Priority)
RJ	(Resume Job)
SJS	(Show Job Status for < user ID>)
XBJ	(Execute Batch SCI Job)

4.236 XJM (EXECUTE JOB MONITOR)

Purpose:

The XJM command displays information about jobs executing on the system. The display is updated every two seconds, until you press the Command key to terminate the display.

Initially, XJM displays the information about the job associated with the ID of the user who logged on the VDT. If you are the system operator, pressing the F1 key steps through the jobs on the system, one job at a time. When all the jobs have been displayed, pressing the F1 key again displays the first job. The F1 key is ignored if you are not the system operator. The Print key can be used to copy the current screen to the logical name SCREEN. The logical name SCREEN can only be assigned to files.

Example:

In the following example, the XJM command displays job monitor information on the terminal where the command is executed. The first and third lines in the display are headings; the other lines are information about the job. Table 4-26 explains the headings.

```
[] XJM

USER-ID   JOBNAME JOB-ID  ACCOUNT   ID     JCA JPRI PRIV CPU-SECS SYNONYM CPU-LOAD
SYSTEM    SMITH     0011                   2048 0C   7      193.6    8704      .4%
TASKNAME  ID RID PR ST  WP   PC   CPU-SECS IO-TRANSFER  SVC-COUNT MEMORY CPU-LOAD
SCI990    01  7F 7C 17 7A86 7892      6.3        66142       1933  40032      .0%
XJM       74  CB 8C 00 C006 CF4E       .1        11200         67   7264      .4%
FILEMGR   05  02 00 24 90A4 C04C     26.9      2947990       5514   2080      .0%
SCI990    01  01 7C 09 9960 0BFA     47.4       545633      13848  40032      .0%
```

Notes:

The XJM command only works on VDTs.

Since the XJM display is updated in cycles, the data may not reflect the current state of a job or task. Task names for tasks terminating may not be valid and the memory may be reallocated; this can occur with tasks in state 4 (being terminated).

XSM (EXECUTE JOB MONITOR)

Table 4-26. Field Description for XJM Display

Field	Description
USER-ID	User ID of the job.
JOBNAME	Name of the job.
JOB-ID	ID of the job.
ACCOUNT ID	Account ID for the displayed job.
JCA	The size in bytes of the job communication area.
JPRI	The job priority.
PRIV	The SCI privilege level of the user ID.
CPU-SECS:	
First Line	The execution time of all the tasks that have run to completion in this job.
Third Line	The total execution time used by the displayed task since it was bid.
SYNONYMS	The number of bytes required to hold the synonyms and logical names for this job.
CPU-LOAD:	
First Line	Percent of time spent executing tasks in this job for the last cycle (cycle = 2 seconds).
Third Line	Percent of time executing the specified task during the last cycle (cycle = 2 seconds).
TASKNAME	The name of the task.
ID	The installed ID of the displayed task.
RID	The run ID of the displayed task.
PR	The run-time priority of the displayed task.
ST	The task state of the displayed task.
WP	The last scheduled workspace pointer of the displayed task.
PC	The program counter of the displayed task.

Table 4-26. Field Description for XJM Display (Continued)

Field	Description
IO-TRANSFER	The number of bytes of I/O the task has sent or received since the task was bid.
SVC-COUNT	The number of SVCs performed since the task was bid.
MEMORY	The number of bytes currently being used by the task. This total includes memory in shared procedures that may also be used by other tasks. The number reflects only that memory being used that is non-memory-resident.

Note:

When a value has too many digits to be displayed in a field, asterisks are displayed.

4.237 XLE (EXECUTE LINK EDITOR)

The XLE command executes the Link Editor after a control file has been developed defining the Link Edit functions. Refer to the *DNOS Link Editor Reference Manual* for further information on the XLE command and the following prompts:

```
EXECUTE LINK EDITOR
        CONTROL ACCESS NAME:  [site:]pathname@        (*)
 LINKED OUTPUT ACCESS NAME:   [site:][pathname]@      (*)
        LISTING ACCESS NAME:  [site:][pathname]@      (*)
      PRINT WIDTH (CHARS):    integer                 (80)
             PAGE LENGTH:     integer                 (59)
```

XMA (EXECUTE MACRO ASSEMBLER)

4.238 XMA (EXECUTE MACRO ASSEMBLER)

The XMA command executes the Macro Assembler on a DNOS operating system. Refer to the *DNOS Assembly Language Programmer's Guide* for further information on the XMA command and the following prompts:

```
EXECUTE MACRO ASSEMBLER
          SOURCE ACCESS NAME:  [site:]pathname@
          OBJECT ACCESS NAME:  [site:][pathname@
         LISTING ACCESS NAME:  [site:]pathname@
           ERROR ACCESS NAME:  [site:][pathname@
                     OPTIONS:  [character(s)]                (*)
       MACRO LIBRARY PATHNAME: [site:][pathname]@            (*)
         PRINT WIDTH (CHARS):  integer                       (80)
        PAGE LENGTH (LINES):   integer                       (60)
```

XOI (EXECUTE OPERATOR INTERFACE)

4.239 XOI (EXECUTE OPERATOR INTERFACE)

Purpose:

The XOI command bids the operator interface task and identifies you as the system operator and your terminal as the system console. Your terminal screen becomes blank. Operator messages and requests are then displayed on your terminal. Any messages not yet seen by the operator are displayed to the console as soon as the XOI command is issued. One of these messages may already have been killed by use of the Kill Operator Interface Request (KOR) command. The XOI command is only valid interactively.

After initially bidding the operator interface task (XOI command), you can continue to execute SCI commands while receiving operator messages and requests. Press the Command key on the terminal to return control to SCI and suspend output by the operator interface task. When you want to resume output of operator messages and requests, enter the XOI command.

You can respond to or kill operator requests without suspending the operator interface task by pressing the F4 or F5 function key. Pressing the F4 function kyy is equivalent to entering the ROR command. Pressing the F5 function key is equivalent to entering the KOR command.

In addition to being able to use the operator interface commands to process messages to the operator, if you are the system operator, you have special privileges. The operator can use the job commands (HJ, KJ, LJ, MJP, RJ) to affect any user job. The operator can examine status of all system resources (SIS command) and spooler resources (HO and KO commands).

Example:

In the following example, the XOI command bids the operator interface task and identifies the user as the system operator and the user's terminal as the system console:

```
[] XOI
```

Notes:

If the user has previously issued a ROM command, successful execution of XOI cancels operator messages being sent to the user's terminal, essentially performing the function of the KOM command.

Operator requests are created by the Create Operator Message command, the spooler subsystem, and by programs that use the OI$ routines described in the *DNOS Systems Programmer's Guide.*

XOI (EXECUTE OPERATOR INTERFACE)

Related Commands:

COM	(Create Operator Message)
KOM	(Kill Operator Messages)
KOR	(Kill Operator Interface Request)
LOM	(List Operator Messages)
QOI	(Quit Operator Interface)
ROM	(Receive Operator Messages)
ROR	(Respond to Operator Interface Request)

XPD (EXECUTE PERFORMANCE DISPLAY)

4.240 XPD (EXECUTE PERFORMANCE DISPLAY)

Purpose:

The XPD command displays system performance statistics, including task counts, job counts, and memory usage. Because the counters are sampled at specific intervals, not constantly, the percentages in the display are estimates.

Press the F1 function key during execution of XPD to reinitialize the display. Press the Command key to terminate the display and return SCI to command mode. To print a copy of the display, assign the logical name SCREEN to a spooler device. Press the Print key each time the display is to be copied.

Figure 4-5 is a sample of the display produced by execution of XPD. Table 4-27 lists the field description for the XPD display. The *DNOS System Programmer's Guide* contains a detailed explanation of the data presented.

Example:

In the following example, the XPD command displays system performance statistics on the terminal where the command is executed.

```
                        D N O S   P E R F O R M A N C E

Number of Clock Ticks:        6685                              Count Active  Wom
      Number Ticks    %  Spot %  Counters:                Tasks:    36      1    0
DS01 U:        43     0     7    Jobs  Cmp:      0        Jobs :    10      1    0
CPU  U:       229     3    24    Tasks Cmp:      0     Memory:   Use     Max   Avail
Schd U:      6460    96    76    Seg   Mgr:      0  Sys Table:   5688   6772  18834
FM T U:        15     0     1    File  Mgr:    177  JCA Table:   2082   2524  12288
Ldr  U:         1     0     0    IPC Calls:      0   SM Table:   4832   5038  12288
Mp 1 U:        75     1     9    Roll Outs:      0   FM Table:   5660   6032  12288
FM X U:        41     0     5    FM Queues:     15   Buf Area:    2848         4096
DS02 U:         0     0     0    Ovly  Lds:      7   User Mem:  355680       956384
DS03 U:         0     0     0    Name  Mgr:      1      Rolled Out: Segments -    0
DS04 U:         0     0     0    IOU Calls:      1              Disk Records -    0
DS05 U:         0     0     0    WOT Calls:      0      Disk Res Tasks in Mem:   31
DS06 U:         0     0     0    JCA Expds:      0  Cache List: Buf -  61 Prg -  16
```

Figure 4-5. Display Produced by XPD Command

Assumptions:

The user will consider the percentages listed in the display produced by the XPD command as estimates.

Notes:

Disk statistics are displayed only if the disks are assigned names such as DS01, DS02, and so forth.

XPD (EXECUTE PERFORMANCE DISPLAY)

Table 4-27. Field Description for XPD Display

Field	Description
Number of Clock Ticks	Number of clock ticks that have elasped since the display was last initialized (for 60 Hz power, 1 tick = 8.33 ms; for 50 Hz power, 1 tick = 10 ms).
Number Ticks	Number of clock ticks in which this item was busy.
%	Percent of time this item was busy since the display was last initialized.
Spot %	Percent of time since the display was last updated.
Counters	Counts of factors since the last initialization of display.
DS01 U	Utilization figures for the disk unit DS01.
CPU U	Utilization of the central processing unit.
Schd U	Task scheduler utilization (includes idle time).
FM T U	File management task utilization.
Ldr U	Task loader utilization.
Mp1 U	Map file 1 utilization. Virtually all task code executes in map file 1.
FM X U	File management XOP level processor (fast transfer) utilization.
DS02 U	Utilization figures for disk unit DS02.
DS03 U	Utilization figures for disk unit DS03.
DS04 U	Utilization figures for disk unit DS04.
DS05 U	Utilization figures for disk unit DS05.
DS06 U	Utilization figures for disk unit DS06.
Jobs Cmp	Number of jobs completed.
Tasks Cmp	Number of tasks completed.
Seg Mgr	Number of segment manager SVCs issued.
File Mgr	Number of file management SVCs issued.
IPC Calls	Number of IPC SVCs issued.
Roll Outs	Number of tasks rolled out.

XPD (EXECUTE PERFORMANCE DISPLAY)

Table 4-27. Field Description for XPD Display (Continued)

Field	Description
FM Queues	Number of file management requests queued (that is, not handled by fast transfer).
Ovly Lds	Number of system overlays loaded.
Name Mgr	Number of Name Management SVCs.
IOU Calls	Number of I/O SVCs for Utility operations.
WOT Calls	Number of times the system table area has become full.
JCA EXPDS	Number of times the user JCAs have been rolled from memory in order to expand. If this figure is large, the system programmer can increase the medium JCA size using the Modify Scheduler/Swap Parameter (MSP) command.
Tasks	Current count of all tasks, active tasks, and tasks on WOM queue. The WOM queue is a queue of tasks that are waiting for memory.
Jobs	Current count of all jobs, active jobs, and jobs with tasks on WOM queue.
Memory	Byte count of memory currently in use, maximum memory used since the display began and the total amount of memory that exists.
Sys Table	System table area.
JCA Table	System job communications area.
SM Table	Segment management table area.
FM Table	File management table area.
Buf Area	I/O buffer table area.
User Mem	User memory.
Rolled Out: Segments	Number of segments on roll file.
Disk Records	Number of disk records in use on roll file.
Disk Res Tasks in Mem	Number of tasks in memory, including those which are memory-resident.
Cache List:	
Buf-	Number of buffers on the cache list.
Prg-	Number of program segments on the cache list.

XSCU (EXECUTE SYSTEM CONFIGURATION UTILITY)

4.241 XSCU (EXECUTE SYSTEM CONFIGURATION UTILITY)

Prompts:

```
EXECUTE SYSTEM CONFIGURATION UTILITY
              SYSTEM VOLUME:  devicename@          (*)
                SYSTEM NAME:  alphanumeric
```

Purpose:

The XSCU command initializes the System Configuration utility. When the utility is active, the following commands perform their functions on the disk image of the system specified in response to the XSCU prompts. All other SCI commands can be executed while XSCU is active. If XSCU is not active, the following commands either operate on the running system or, in the case of MCC, MDC, and MST, return an error message indicating that they can execute only when the System Configuration utility is active:

ISL	Initialize System Log
LDC	List Device Configuration
MCC	Modify Country Code
MDC	Modify Device Configuration
MDS	Modify Device State
MSP	Modify Scheduler/Swap Parameters
MST	Modify System Table Sizes
SCC	Show Country Code

When you enter the XSCU command, a listing of the device configuration for the specified system is displayed on your terminal. To return to command mode, press the Command key.

Terminate the XSCU command by executing the Quit Configuration Utility Session (QSCU) command. The response to the ABORT? prompt of the QSCU command determines if the modifications are applied or disregarded.

Prompt Details:

SYSTEM VOLUME:
The disk drive or volume name that contains the system image to be modified.

SYSTEM NAME:
The name of the system specified in response to tthe OUTPUT CONFIGURATION prompt for the Execute System Generation Utility (XSGU) command when the system was generated.

XSCU (EXECUTE SYSTEM CONFIGURATION UTILITY)

Example:

In the following example, the XSCU command is used to modify the disk image of the system, DNOSYS, installed in disk device DS03. A Modify Device Configuration (MDC) command is executed during XSCU activation to add a VDT to the device configuration. An LDC command is issued after the modification to allow inspection of the results. The QSCU command terminates XSCU.

```
[] XSCU

EXECUTE SYSTEM CONFIGURATION UTILITY
     SYSTEM VOLUME: DS03
       SYSTEM NAME: DNOSYS

                        CRU/      MUX                                          8-BIT
NAME   DEVICE          TILINE   CHANNEL   INTERRUPT   CHASSIS   POSITION   MODE  CHARS

DS01   TILINE DISK     F800                   13                            ON      Y
DS02   TILINE DISK     F800                   13                            DIAG    N
LP01   LP:PARALLEL     0460                    7          1         14      ON      N
ST01   VDT-911          0100                   10                            ON      N

[] MDC

MODIFY DEVICE CONFIGURATION
              DATA DISK/VOLUME: LNKDNOS

MODIFY DEVICE CONFIGURATION
    COMMAND(CHANGE,ADD,DELETE): A

ADD DEVICE
                   DEVICE TYPE: VDT

ADD VDT
                      VDT TYPE: 911

ADD VDT - 911
               OPENS VALIDATED?: NO
                       TIME OUT: 0
                CHARACTER QUEUE: 10
                    CRU ADDRESS: 0120
                      INTERRUPT: 8
              EXPANSION CHASSIS:
            EXPANSION POSITION:
                       CDT NUMBER: >0A
                         CDE MASK: >E000
```

XSCU (EXECUTE SYSTEM CONFIGURATION UTILITY)

```
[] LDC

LIST DEVICE CONFIGURATION
    LISTING ACCESS NAME:

                        CRU/      MUX                                          8-BIT
NAME    DEVICE          TILINE    CHANNEL   INTERRUPT   CHASSIS   POSITION   MODE   CHARS

DS01    TILINE DISK     F800                    13                           ON       Y
DS02    TILINE DISK     F800                    13                           DIAG     N
LP01    LP:PARALLEL     0460                     7          1         14     ON       N
ST01    VDT-911         0100                    10                           ON       N
ST02    VDT-911         0120                     8                           ON       N


[] QSCU

QUIT CONFIGURATION UTILITY SESSION
       ABORT?: NO
```

Related Commands:

ISL	(Initialize System Log)
LDC	(List Device Configuration)
MCC	(Modify Country Code)
MDC	(Modify Device Configuration)
MDS	(Modify Device State)
MSP	(Modify Scheduler/Swap Parameters)
MST	(Modify System Table Sizes)
QSCU	(Quit System Configuration Utility)
SCC	(Show Country Code)

XSGU (EXECUTE SYSTEM GENERATION UTILITY)

4.242 XSGU (EXECUTE SYSTEM GENERATION UTILITY)

The XSGU command generates a custom DNOS operating system.

NOTE

For a multiple disk system, the system disk might not contain enough space to perform a system generation. Create more room by copying the XSGU input files (library .SSGU) to a secondary disk and deleting them from the system disk.

Refer to the *DNOS System Generation Reference Manual* for further information on the XSGU command and the following prompts:

```
EXECUTE SYSTEM GENERATION UTILITY
          DATA DISK/VOLUME:  [site:]{devicename/volume name}@  (*)
        TARGET DISK/VOLUME:  [site:]{devicename/volume name}@  (*)
     INPUT CONFIGURATION:    [pathname]                        (*)
    OUTPUT CONFIGURATION:    pathname                          (*)
       ASSEMBLE AND LINK:    [YES/NO]                          (NO)
```

4.243 XT (EXECUTE TASK)

Prompts:

```
EXECUTE TASK
        PROGRAM FILE OR LUNO:  {filename/integer}@      (*)
             TASK NAME OR ID:  {alphanumeric/integer}   (*)
                       PARM1:  integer                  (0)
                       PARM2:  integer                  (0)
                  STATION ID:  {integer/ME}             (*)
```

Purpose:

The XT command activates a program that does not interact with your terminal. Two 16-bit words of information can be passed to the program being activated in response to the PARM1 and PARM2 prompts of the XT command. The operating system automatically assigns a run-time ID to each program that it activates. A task activated by the XT command cannot access event characters entered at your terminal.

Prompt Details:

PROGRAM FILE OR LUNO:
The file name of or the LUNO assigned to the program file on which the task to be executed is installed. If you specify a LUNO in response to this prompt, you must assign it prior to the execution of the XT command. If you specify zero, the .S$SHARED program file is assumed. If you specify > FF, the utility program file is assumed.

TASK NAME OR ID:
Either the name or ID under which the program is installed on the specified program file.

PARM1: and PARM2:
Decimal or hexadecimal numbers in the range of −32768 through 65,535 representing a value to be passed to the program.

STATION ID:
The number (that is, 2, not ST02) of the station with which the executing task is to be associated. A zero or the characters ME indicate your terminal. The use of > FF indicates the task is not to be associated with a station.

A task should not be associated with a station unless it is used by the task for terminal I/O. If you specify a station ID through the XT command and SCI is quit (via the Quit SCI (Q) command) before the task terminates, log-on to SCI cannot be performed until the task terminates.

XT (EXECUTE TASK)

Example:

In the following example, a task on the system program file, with an installed ID of >A5, is activated through the XT command and associated with the user's own station, ST05. The prompts and responses of the XT command are shown below:

```
[] XT

EXECUTE TASK
     PROGRAM FILE OR LUNO: .S$PROG
          TASK NAME OR ID: >A5
                     PARM1: 0
                     PARM2: 0
                STATION ID: 5
```

If the task is activated successfully, the system assigned run-time ID is displayed on the user's terminal:

```
RUNTIME TASK ID = >3A
```

Assumptions:

The task that is to be activated has previously been installed on the specified program file.

Notes:

The hidden prompt, DISPLAY, is not displayed in interactive mode but can be specified in interactive expert mode or batch mode. The DISPLAY prompt can be used to control whether or not the task ID is displayed on your terminal. If you respond NO to this prompt, the task ID is not displayed. The default is YES.

Related Commands:

XTS (Execute Task and Suspend SCI)

XTS (EXECUTE TASK AND SUSPEND SCI)

4.244 XTS (EXECUTE TASK AND SUSPEND SCI)

Prompts:

```
EXECUTE TASK AND SUSPEND SCI
        PROGRAM FILE OR LUNO:  {filename/integer}@      (*)
            TASK NAME OR ID:   {alphanumeric/integer}   (*)
                      PARM1:   integer                  (0)
                      PARM2:   integer                  (0)
                 STATION ID:   {integer/ME}             (*)
```

Purpose:

The XTS command activates an interactive program and automatically suspends SCI to prevent it from interfering with the execution of the program. If SCI were not suspended, it would continue to interpret data entered at the terminal as though that data were intended for SCI, and an error would result. Use this command also to make event characters available to a task other than SCI.

Prompt Details:

PROGRAM FILE OR LUNO:
The file name of or the LUNO assigned to the program file on which the task to be executed is installed. If you specify a LUNO as the response to this prompt, you must assign it prior to the execution of the XTS command. If you specify zero, the .S$SHARED program file is assumed. If you specify > FF, the utility program file is assumed.

TASK NAME OR ID:
Either the name or the ID number under which the program was installed.

PARM1: and PARM2:
Decimal or hexadecimal numbers in the range of – 32768 through 65,535 representing a value to be passed to the program.

STATION ID:
The number (that is, 2, not ST02) of the station with which the executing task is to be associated. Zero or ME indicates your terminal. The use of > FF indicates that the task is not to be associated with a station.

XTS (EXECUTE TASK AND SUSPEND SCI)

Example:

In the following example, the XTS command activates an interactive task with an installed ID of > A5, and the task is associated with the user's own station. The run-time ID (automatically assigned by the operating system) is listed when the program is successfully executed.

```
[] XTS

EXECUTE TASK AND SUSPEND SCI
     PROGRAM FILE OR LUNO: .S$SHARED
          TASK NAME OR ID: >A5
                     PARM1: 0
                     PARM2: 0
                STATION ID: 0

RUNTIME TASK ID = >38
```

Assumptions:

The program file specified exists, and the task specified has been installed on that program file.

Notes:

If you specify anything other than the initiating station in response to the prompt STATION ID, then SCI is still suspended at the initiating terminal. When the task terminates, SCI is reactivated at the initiating terminal.

The hidden prompt, DISPLAY, is not displayed in interactive mode but can be specified in interactive expert mode or batch mode. The DISPLAY prompt can be used to control whether or not the task ID is displayed on your terminal. If you respond NO to this prompt, the task ID is not displayed. The default is YES.

Related Commands:

XT (Execute Task)

Appendix A

Keycap Cross-Reference

Generic keycap names that apply to all terminals are used for keys on keyboards throughout this manual. This appendix contains specific keyboard information to help you identify individual keys on any supported terminal. For instance, every terminal has an Attention key, but not all Attention keys look alike or have the same position on the keyboard. You can use the terminal information in this appendix to find the Attention key on any terminal.

The terminals supported are the 931 VDT, 911 VDT, 915 VDT, 940 EVT, the Business System terminal, and hard-copy terminals (including teleprinter devices). The 820 KSR has been used as a typical hard-copy terminal. The 915 VDT keyboard information is the same as that for the 911 VDT except where noted in the tables.

Appendix A contains three tables and keyboard drawings of the supported terminals.

Table A-1 lists the generic keycap names alphabetically and provides illustrations of the corresponding keycaps on each of the currently supported keyboards. When you need to press two keys to obtain a function, both keys are shown in the table. For example, on the 940 EVT the Attention key function is activated by pressing and holding down the Shift key while pressing the key labeled PREV FORM NEXT. Table A-1 shows the generic keycap name as Attention, and a corresponding illustration shows a key labeled SHIFT above a key named PREV FORM NEXT.

Function keys, such as F1, F2, and so on, are considered to be already generic and do not need further definition. However, a function key becomes generic when it does not appear on a certain keyboard but has an alternate key sequence. For that reason, the function keys are included in the table.

Multiple key sequences and simultaneous keystrokes can also be described in generic keycap names that are applicable to all terminals. For example, you use a multiple key sequence and simultaneous keystrokes with the log-on function. You log on by *pressing the Attention key, then holding down the Shift key while you press the exclamation (!) key.* The same information in a table appears as *Attention/(Shift)!*.

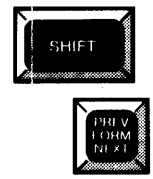
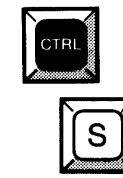
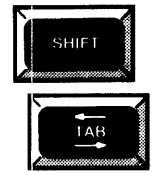
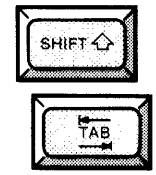
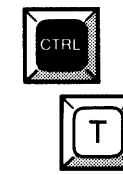
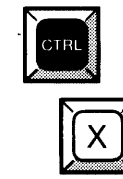
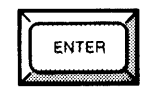
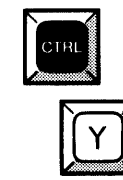
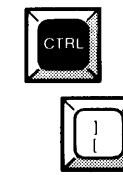
Table A-2 shows some frequently used multiple key sequences.

Table A-3 lists the generic names for 911 keycap designations used in previous manuals. You can use this table to translate existing documentation into generic keycap documentation.

Figures A-1 through A-5 show diagrams of the 911 VDT, 915 VDT, 940 EVT, 931 VDT, and Business System terminal, respectively. Figure A-6 shows a diagram of the 820 KSR.

2274834 (1/14)

Table A-1. Generic Keycap Names

Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
Alternate Mode	None				None
Attention[2]					
Back Tab	None			None	
Command[2]					
Control					
Delete Character					None
Enter					
Erase Field					

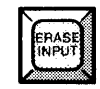
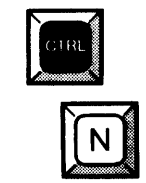
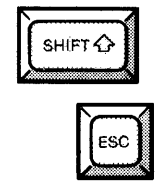
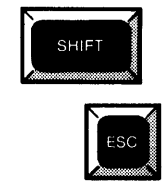
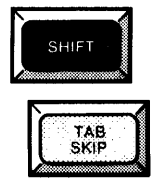
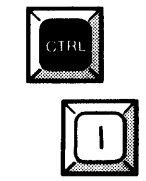
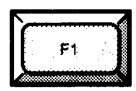
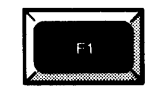
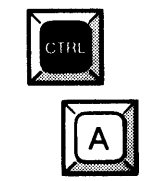
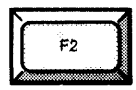
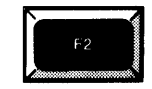
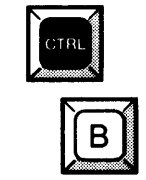
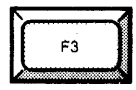
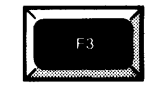
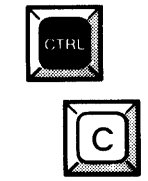
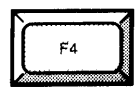
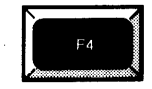
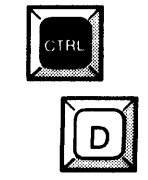
Notes:

[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

[2]On a 915 VDT the Command Key has the label F9 and the Attention Key has the label F10.

2284734 (2/14)

Table A-1. Generic Keycap Names (Continued)

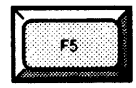
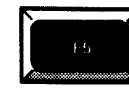
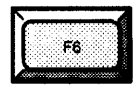
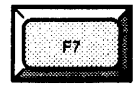
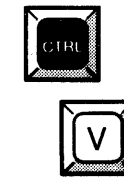
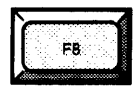
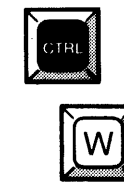
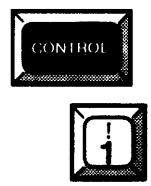
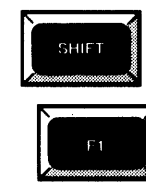
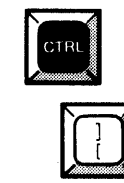
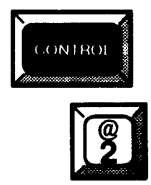
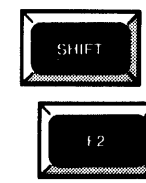
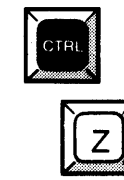
Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
Erase Input					
Exit					
Forward Tab					
F1					
F2					
F3					
F4					

Notes:

[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

2284734 (3/14)

Table A-1. Generic Keycap Names (Continued)

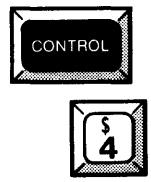
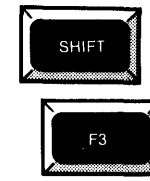
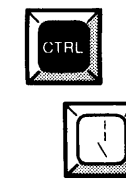
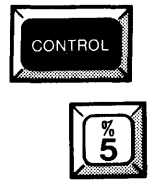
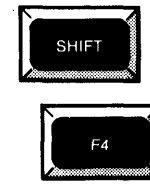
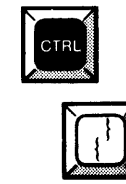
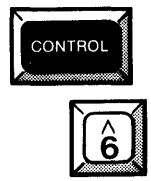
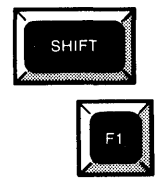
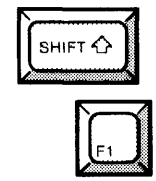
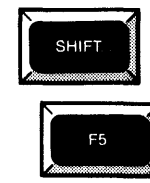
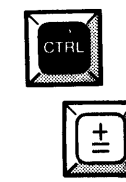
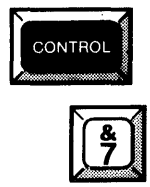
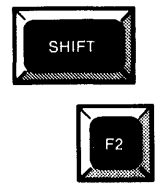
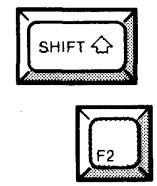
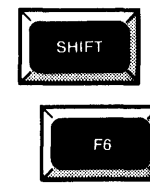
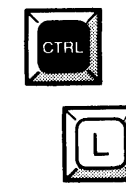
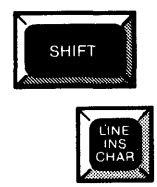
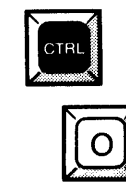
Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
F5					
F6					
F7					
F8					
F9					
F10					

Notes:

[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

2284734 (4/14)

Table A-1. Generic Keycap Names (Continued)

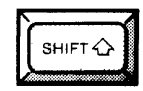
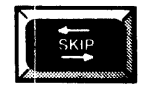
Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
F11					
F12					
F13					
F14					
Home					
Initialize Input					

Notes:

[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

2284734 (5/14)

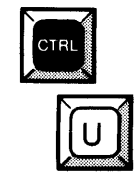
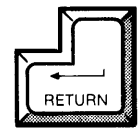
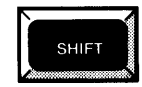
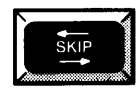
Table A-1. Generic Keycap Names (Continued)

Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
Insert Character					None
Next Character	 or  				None
Next Field	 		 	 	None
Next Line					  or 
Previous Character	 or 				None
Previous Field		 			None

Notes:
[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

2284734 (6/14)

Table A-1. Generic Keycap Names (Continued)

Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
Previous Line					
Print					None
Repeat		See Note 3	See Note 3	See Note 3	None
Return					
Shift					
Skip					None
Uppercase Lock					

Notes:

[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

[3]The keyboard is typamatic, and no repeat key is needed.

2284734 (7/14)

Table A-2. Frequently Used Key Sequences

Function	Key Sequence
Log-on	Attention/(Shift)!
Hard-break	Attention/(Control)x
Hold	Attention
Resume	Any key

Table A-3. 911 Keycap Name Equivalents

911 Phrase	Generic Name
Blank gray	Initialize Input
Blank orange	Attention
Down arrow	Next Line
Escape	Exit
Left arrow	Previous Character
Right arrow	Next Character
Up arrow	Previous Line

2284734 (8/14)

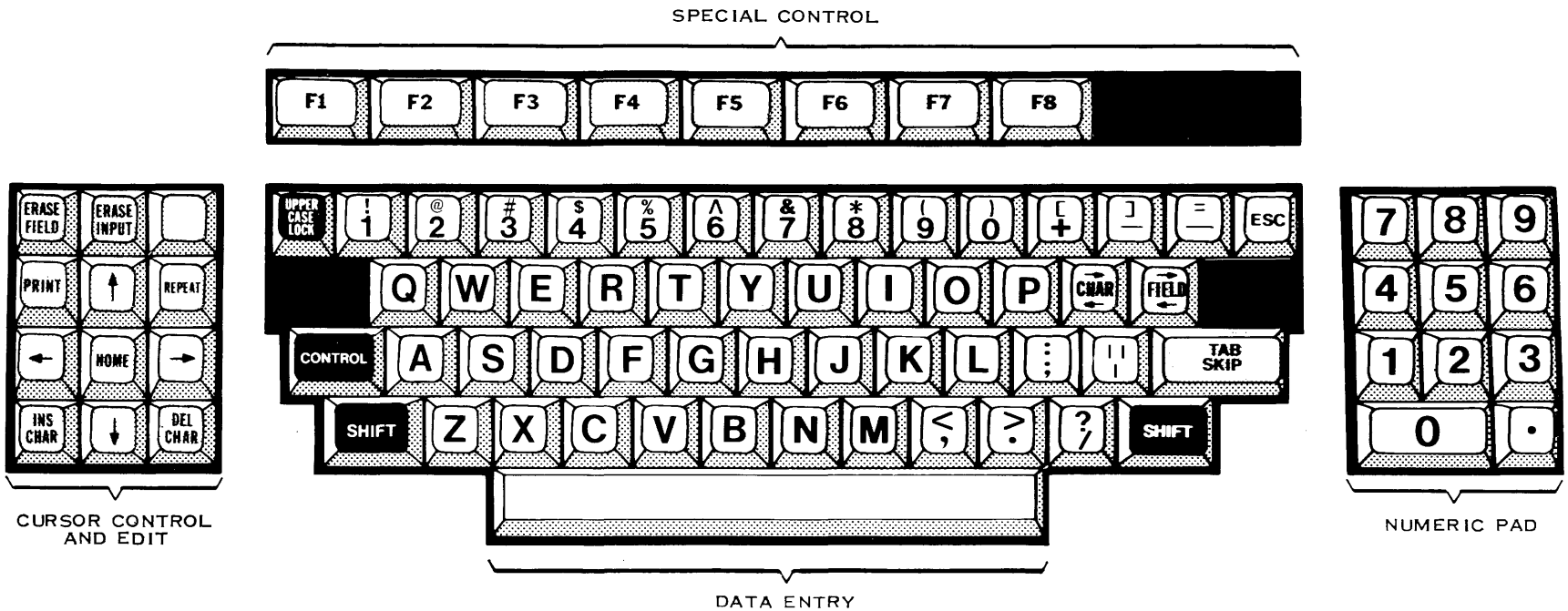


Figure A-1. 911 VDT Standard Keyboard Layout

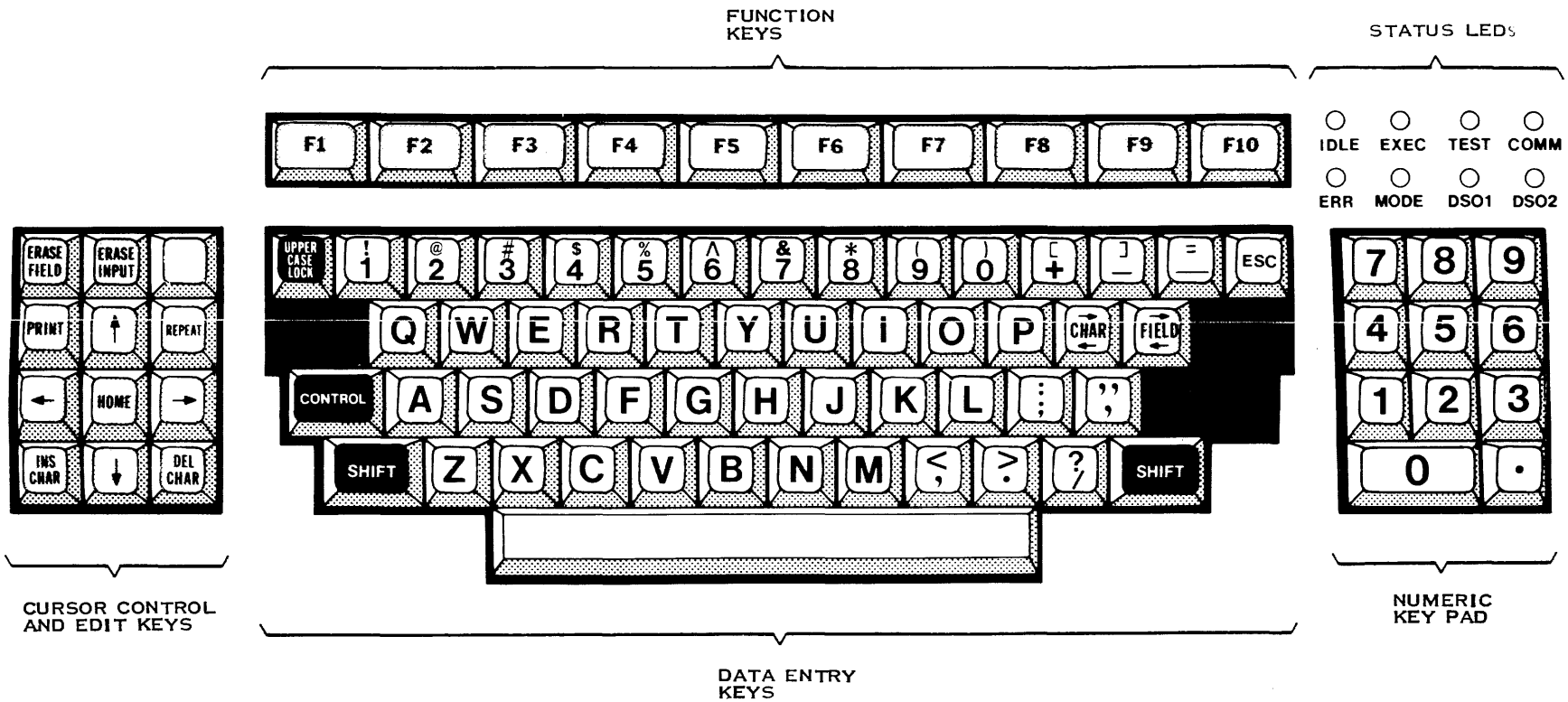
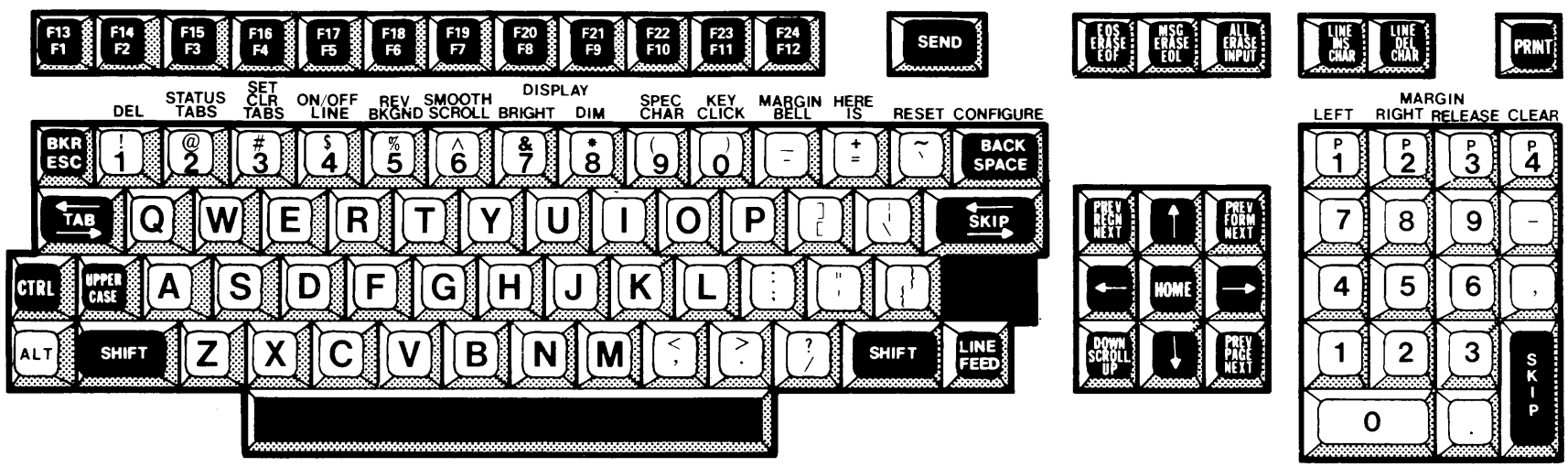


Figure A-2. 915 VDT Standard Keyboard Layout



2284734 (11/14)

Figure A-3. 940 EVT Standard Keyboard Layout

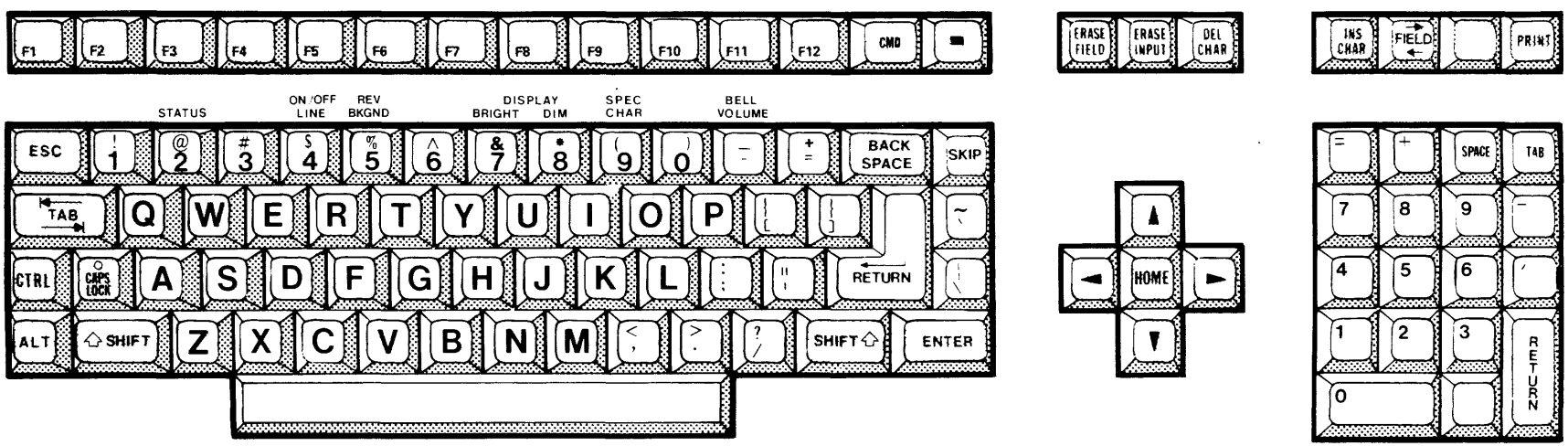
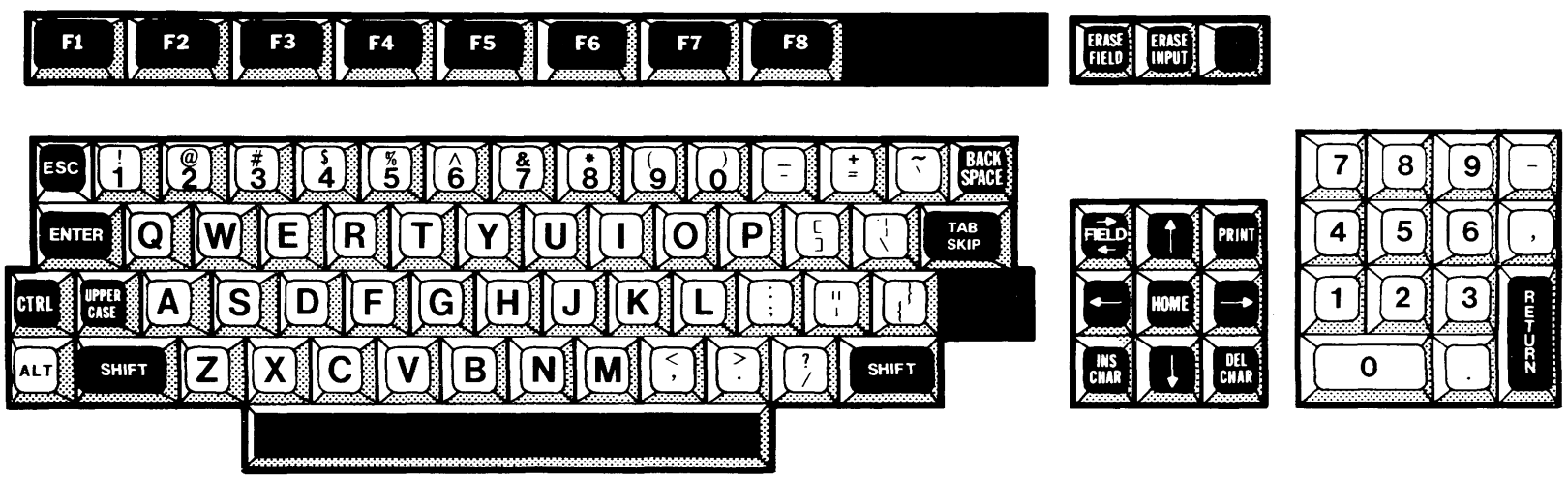
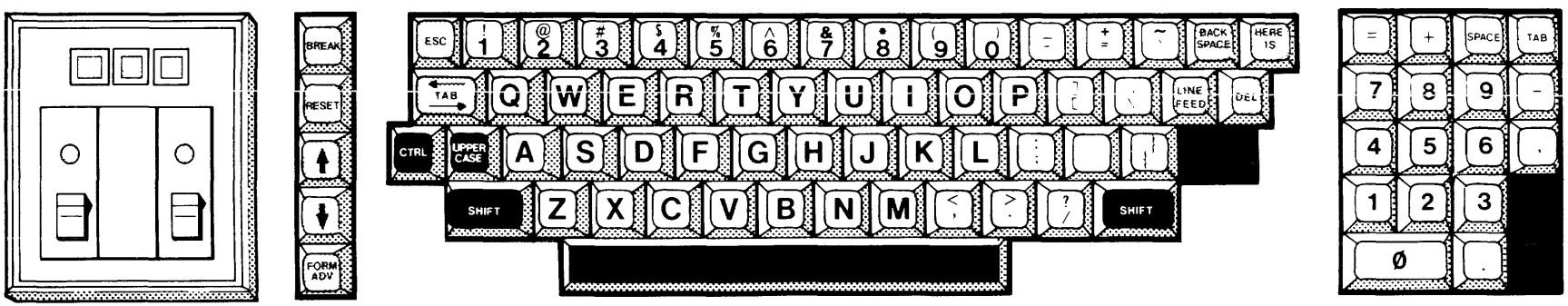


Figure A-4. 931 VDT Standard Keyboard Layout



2284734 (13/14)

Figure A-5. Business System Terminal Standard Keyboard Layout



2284734 (14/14)

Figure A-6. 820 KSR Standard Keyboard Layout

Appendix B

Index of SCI Commands

Table B-1 is a rearranged index of the SCI commands discussed in Section 4 of this manual to give you ease in referencing them.

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Table B-1. SCI Command Index

Alphabetical Index

Introduction

HOW TO USE INDEX

The index, table of contents, list of illustrations, and list of tables are used in conjunction to obtain the location of the desired subject. Once the subject or topic has been located in the index, use the appropriate paragraph number, figure number, or table number to obtain the corresponding page number from the table of contents, list of illustrations, or list of tables.

INDEX ENTRIES

The following index lists key words and concepts from the subject material of the manual together with the area(s) in the manual that supply major coverage of the listed concept. The numbers along the right side of the listing reference the following manual areas:

- Sections — Reference to Sections of the manual appear as "Sections x" with the symbol x representing any numeric quantity.

- Appendixes — Reference to Appendixes of the manual appear as "Appendix y" with the symbol y representing any capital letter.

- Paragraphs — Reference to paragraphs of the manual appear as a series of alphanumeric or numeric characters punctuated with decimal points. Only the first character of the string may be a letter; all subsequent characters are numbers. The first character refers to the section or appendix of the manual in which the paragraph may be found.

- Tables — References to tables in the manual are represented by the capital letter T followed immediately by another alphanumeric character (representing the section or appendix of the manual containing the table). The second character is followed by a dash (-) and a number.

  Tx-yy

- Figures — References to figures in the manual are represented by the capital letter F followed immediately by another alphanumeric character (representing the section or appendix of the manual containing the figure). The second character is followed by a dash (-) and a number.

  Fx-yy

- Other entries in the Index — References to other entries in the index preceded by the word "See" followed by the referenced entry.

USER'S RESPONSE SHEET

Manual Title: DNOS System Command Interpreter (SCI) Reference Manual (2270503-9701)

Manual Date: 15 November 1983 **Date of This Letter:** _____

User's Name: _____ **Telephone:** _____

Company: _____ **Office/Department:** _____

Street Address: _____

City/State/Zip Code: _____

Please list any discrepancy found in this manual by page, paragraph, figure, or table number in the following space. If there are any other suggestions that you wish to make, feel free to include them. Thank you.

Location in Manual	Comment/Suggestion
_____	_____
	_____
	_____
	_____
_____	_____
	_____
	_____
	_____
	_____
_____	_____
	_____
	_____
	_____

CUT ALONG LINE

NO POSTAGE NECESSARY IF MAILED IN U.S.A.
FOLD ON TWO LINES (LOCATED ON REVERSE SIDE), TAPE AND MAIL

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 7284 DALLAS, TX

POSTAGE WILL BE PAID BY ADDRESSEE

TEXAS INSTRUMENTS INCORPORATED
DIGITAL SYSTEMS GROUP

ATTN: TECHNICAL PUBLICATIONS
P.O. Box 2909 M/S 2146
Austin, Texas 78769

FOLD