August 1978

This manual provides the complete range of information required for efficient management of a TRAX system.

TRAX
System Manager's Guide

AA-D332A-TC

OPERATING SYSTEM AND VERSION: TRAX Version 1.0

To order additional copies of this document, contact the Software Distribution Center, Digital Equipment Corporation, Maynard, Massachusetts 01754

digital equipment corporation · maynard. massachusetts

First Printing, August 1978

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

The following are trademarks of Digital Equipment Corporation:

DIGITAL	DECsystem-10	MASSBUS
DEC	DECtape	OMNIBUS
PDP	DIBOL	OS/8
DECUS	EDUSYSTEM	PHA
UNIBUS	FLIP CHIP	RSTS
COMPUTER LABS	FOCAL	RSX
COMTEX	INDAC	TYPESET-8
DDT	LAB-8	TYPESET-11
DECCOMM	DECSYSTEM-20	TMS-11
ASSIST-11	RTS-8	ITPS-10

CONTENTS

PREFACE

This manual is intended for use by the TRAX System Manager and contains the full range of information necessary to perform this function.

Chapter 1 describes, in general terms, the operating responsibilities of a System Manager including management of system resources. System access, management of batch and spool facilities and TRAX utility dialog conventions are also described in a general manner.

Chapter 2 provides the specific operating information on starting, shutdown, and restart of a TRAX system.

Chapter 3 describes how to manage transaction processors including installation, starting, stopping and removing of a transaction processor.

Chapter 4 presents the detailed information on managing TRAX batch and spool facilities. These facilities are delineated in terms of queues, processors and jobs.

Chapter 5 describes the maintenance of transaction processors and data files in terms of the specific support environment utilities used to perform this management function.

Chapter 6 covers transaction processor software error logging and reporting by describing in detail the TRAX support environment utilities which are used to perform these functions.

Chapter 7 describes the TRAX Digital Command Language (DCL) commands used to invoke specific utilities, to communicate with other support terminal users, and to control support terminal access.

Chapter 8 describes how to manage system device resources including the volume and files supported by these devices.

Chapter 9 provides detailed information on the use of those support environment utilities used to maintain the support environment.

Chapter 10 presents the format convention for TRAX Digital Command Language (DCL) commands.

Chapter 11 is an alphabetical listing of each TRAX/DCL command along with all formal details and descriptions of all pertinent parameters, a brief example of the use of each command is also provided.

Appendix A presents a sample system dialog. Appendix B lists all TRAX support environmental messages. Appendix C lists TRAX I/O error codes. Appendix D lists RMS completion status codes.

CHAPTER 1
MANAGING THE TRAX SYSTEM

As the manager of the TRAX system, you are responsible for allocating system resources for optimum system performance, for monitoring the system performance, for maintaining locally generated and DIGITAL-supplied documentation, and for communicating documentation changes. To ensure that you perform these steps efficiently, you should be familiar with transaction-processing concepts and practices or you should have a close working relationship with a senior programmer or analyst who is experienced in transaction processing.

You allocate system resources (that is, devices and memory) during system generation; refer to the *System Generation Manual* for details. Depending upon the type of system you specify at system generation, the system allocates separate areas of memory for transaction-processing environment software, for support environment software, and for the operating system software. Each allocated area of memory is called a partition. You can specify the system to be:

1. Transaction-processing environment with a minimum support environment
2. Support environment-only
3. Transaction-processing environment with a full support environment.

The system supplies a GEN partition (that is, the General Partition), one or more transaction processor partitions, and several operating system partitions. The GEN partition is where the system runs most of the support environment software or tasks. After the system allocates a minimum amount of memory to the various partitions, the system permits you to allocate any remaining areas of memory to either the transaction-processing partitions or to the GEN partition. If there are any transaction-processing partitions, you are advised to allocate any remaining areas of memory to them unless you want to perform batch operations concurrently with transaction processing. The system uses any excess memory in a transaction processor partition for cache, which can reduce the number of disk transfers. If you do not allocate all of the remaining memory, the system allocates it to the GEN partition.

The partitions, tasks, system tables, etc. are stored in a file which is referred to as a system image. Chapter 2 in this guide explains how to handle system images.

1.1 OPERATING RESPONSIBILITIES

As the manager of the TRAX system, you must know the day-to-day operations of the system. To manage daily system operations, such as system startup, backup/restoration and error-monitoring, you must know how and when to run various support programs and transaction processors.

Finally, you should designate an individual to document locally generated procedures and to maintain the DIGITAL-supplied documentation. That individual should ensure that members of the staff receive the latest information. In many cases, lack of awareness of published data results in improper utilization of resources.

1.2 MANAGING SYSTEM RESOURCES

You allocate memory and devices at system generation. After you select devices for an environment, you can change them only by performing another system generation. You can reallocate memory to different partitions after system generation with the SETUP utility.

1.2.1 Allocating Memory

If you start up or boot a transaction-processing environment with a minimum support environment system, the system supplies a minimum GEN partition for controlling transaction processors. Restrict the support environment access to one terminal, TT0:. Refer to Chapter 7 in this guide for an explanation on how to restrict support terminal access. Permitting more than one user to access the support environment system can overload the system resources.

The GEN partition is large enough to perform some support environment functions, such as transaction processor control and error logging. However, program development functions cannot run in the minimum GEN partition. If you allocate additional memory to the GEN partition, the performance of the transaction processors will be decreased due to sharing system resources with the program development functions. Table 1-1 lists the support environment utilities that run concurrently with a transaction processor on a minimum-sized TRAX system.

If you start up or boot a support environment-only system, memory is available for program development and system support (that is, you can develop and debug a TST – transaction step task – but you cannot run a transaction processor).

If you start up or boot a transaction-processing environment with a full support environment, memory is available for transaction processing, program development and system support. This type of system requires a large memory configuration.

Table 1-1 Minimum General Partition Support Utilities

Utility	Function
AUXSTP	The AUXSTP utility stops the journal activities for all transaction processors and then stops the software error logger.
ERRLOG	The ERRLOG utility is the kernel error logger and writes any kernel hardware errors to a disk file. The system initiates the ERRLOG utility when you start the system and automatically cancels the ERRLOG utility at shutdown. For a description of how to display the kernel errors, refer to the SETUP utility.
SERLOG	The SERLOG utility records any user software errors and changes in status – such as an application terminal error – detected by a transaction processor. The system initiates the SERLOG utility when you start the system and automatically cancels the SERLOG utility at shutdown. For a description of how to display the user software errors, refer to either the SERDAY utility or SERANL utility.
SERDAY	You use the SERDAY utility to display a report of today's transaction processor software errors that are recorded by the SERLOG utility.
SHUTDOWN	You use the SHUTDOWN utility to shut down the system.
SYSMOD	You use the SYSMOD utility to assign or deassign a journal device. To start and stop the journal activities for a transaction processor, refer to the TPCTRL utility.
TPCTRL	You use the TPCTRL utility to install, start, stop, and remove the transaction processors and to start and stop journal activities.
TPMOD	You use the TPMOD utility to modify transaction processors (for instance, redirect stations or abort a transaction instance).
TPSTAT	You use the TPSTAT utility to record a sampling of transaction processor activities. For example, you can request a sampling at 5-minute intervals. The TPSTAT utility then records a sampling of the system activities during the sampling time period. You then use the STAREP utility to display a report of the activities gathered by the TPSTAT utility.

1.2.2 Allocating Devices

At system generation, you select devices – except for the system disk(s) – for either the transaction-processing environment or for the support environment. Terminals you select for the transaction-processing environment are called application terminals or stations while terminals you select for the support environment are called support terminals. Transaction processors utilize the application terminals. You utilize the support terminals through the DIGITAL Command Language (DCL) commands.

1.3 LOGGING AND REPORTING ERRORS

The system performs kernel error logging and transaction processor error logging. The kernel error logger monitors hardware errors (for example, disks, magnetic tapes, and memory) while the transaction processor error logger monitors transaction processor and application terminal errors.

1.3.1 Logging and Reporting Kernel Errors

At system startup, the system initiates the kernel error logger.

The error logging subsystem monitors the hardware reliability of the system. It continually detects and records information about every disk, magnetic tape, and memory parity error as it occurs. Then at user-determined intervals, you can run the SETUP utility to produce an error report.

Error logging reports are useful for efficient maintenance of the hardware. Problems, such as line noise, static discharges, and inherently error-prone media, can cause recoverable errors on systems that are otherwise functioning normally. By studying error logging reports, you can learn to distinguish these kinds of errors from those that might be symptoms of an impending device failure. On the other hand, some recoverable errors that are insignificant in themselves might be related to other, more serious errors, the effects of which are not immediately apparent. Information contained in the reports about each error and about the status of the system when the error occurred may alert you to a previously unforeseen hardware problem.

When the error reports seem to indicate an impending failure, you should contact a DIGITAL field service engineer who will use the reports to help in diagnosing the problem. Sometimes a device fails so quickly that error logging is unable to predict the failure in time to prevent it. In this case, the field service engineer can determine the cause more quickly if a report is available which describes the errors that may have occurred immediately prior to the failure. Refer to Chapter 9 in this guide for more information.

1.3.2 Logging and Reporting Software Errors

At system startup, the system initiates the transaction processor error logger if the system contains a transaction processor partition. If the transaction processor detects a user software error or a change in status such as an application terminal error, the transaction processor error logger records in an error log file information about every user software error.

At your convenience, you can access the error log file with transaction processor error log utilities to generate summary and detailed error reports. The reports can be for all transaction processors or for specific transaction processor(s). In addition, you can selectively determine which transaction processor(s) to monitor with the transaction processor error logger. Refer to Chapter 6 in this guide for more information.

1.4 CONTROLLING ACCESS TO THE SYSTEM

You control access to the transaction-processing environment and support environment through authorization facilities. You define authorizations to these environments by defining access codes and passwords. (You cannot access one environment from the other environment.) The system uses the codes as a protection mechanism by requiring that you know an access code before allowing you access. You define access codes to the transaction-processing environment by specifying access codes and work class definitions. Refer to the *Application Programmer's Guide* for more information. Similarly you define access codes to the support environment and then define the accesses to each device, volume, and file.

1.5 MANAGING BATCH AND SPOOL FACILITIES

When you start up or boot the system, the system initializes and starts the batch and spool facilities. You control the facilities through DCL commands. For example, you can stop the batch operation or add more simultaneous batch operations.

The batch facility handles DCL commands and utility commands as though you entered the commands from a keyboard. Functions which you do repeatedly through keyboard entry can be performed through the batch facility. The functions you invoke through the batch facility are run in the GEN partition, whose previously specified restrictions also apply here.

The spooling facility handles printer operations by sending print-formatted files to a printer. When you attempt to access a spooled device directly, the spooling facility creates a temporary file instead of outputting directly to the spooled device.

1.6 TRAX UTILITY DIALOG CONVENTIONS

All TRAX utility dialogs adhere to the following conventions:

1. Help Text – If the question does not give enough information to enable you to answer, type a question mark followed by the RETURN key (? <RET>)

   The utility responds with an explanation and repeats the questions.

   ```
   Transaction processor name? ? <RET>
   Enter the name of an existing transaction processor,
   a 1- to 6-character alphanumeric string.
   Transaction processor name?
   ```

2. The <RETURN> Key – Pressing the RETURN key (<RET>) terminates your response.

   Station type? TERMINAL <RET>

   Station name?

3. Responses – The content of each question indicates the type of response expected by the utility. The utility checks the input as you enter it. If the input is incorrect in any way, the utility returns an error message and repeats the question.
4. Defaults – If the system assumes a default value as a response to a question, that value is shown in brackets following the question.

   Station priority <128>

   You can accept the default value by pressing the RETURN key. Values other than the default are specified as a normal response to the question. For example, to accept the default, simply press RETURN:

   Station priority <128> <RET>

   To specify a different value, type the response followed by a carriage return.

   Station priority <128> 124 <RET>

5. YES/NO Answers – If a question requires a YES or NO answer, you can respond by typing Y, YE, YES, N, or NO as a response to the question.

   Stage updates <YES>? N <RET>

   Repeat <NO>? YE <RET>

6. Numeric Replies – Any question that requires a numeric reply expects decimal input, unless the question indicates otherwise.

   Exchange time limit [minutes]? 5 <RET>

7. [text] – Some questions include text within brackets to clarify the entity or units to be specified.

   Maximum size of exchange message [bytes]?

8. Abbreviating Responses – If a question asks you to select an item from a known set (utility commands, for example), you need to enter only the number of characters required to uniquely identify the selected item within the set.

   Command? P <RET>

   P stands for PRINT, one command from a set that includes ADD, DELETE, EDIT, EXIT, INDEX, PRINT, and SHOW.
9. The <ESC> Key – All utilities ask questions in a specific order. To reverse the order, that is, to return to the last question asked, press <ESC>. In the case where the utility executes a loop of questions, <ESC> returns you to the first question of the loop.

Exchange label? ACNT <RET>
Form name? <ESC>
Exchange label?

Destinations command <DONE>? ADD <RET>
Destination station name? INSERT <RET>

Wait <YES>? <ESC>
Destinations command <DONE>?

10. CTRL/Z to Exit – You may make an orderly exit from a TRAX utility at any time by typing CTRL/Z in response to a question.

CHAPTER 2
SYSTEM STARTUP, SHUTDOWN, AND RESTART

This chapter contains the information required for system startup and shutdown. System startup begins when you manipulate console switches and ends with an executed system image. The system then performs various functions as you direct. The system interacts with you by menas of messages and user responses on your terminal.

Also explained here is the process of booting other system images and the procedures to be followed if a system crash occurs. You can bootstrap only one system image with the console switches as described in Section 2.1. You can bootstrap other system images with the SETUP utility as described in Section 2.4.

2.1 BOOTSTRAPPING WITH THE CONSOLE SWITCHES

The procedure for bootstrapping depends upon the type of host processor. Refer to either Section 2.1.1 or 2.1.2 to bootstrap the system.

2.1.1 Bootstrapping a PDP-11/34 and PDP-11/60

Perform the following steps to bootstrap the PDP-11/34 or PDP-11/60 processor.

1. Press the CTRL and HALT switches simultaneously.
2. Press the CTRL and BOOT switches.
3. In response to the $ prompt on the console terminal, enter either DB, DM, DR, or MM and the 1- or 2-digit device unit number.

   DB is the device name for RP04, RP05, and RP06 disks.

   DM is the device name for the RK07 disk.

   DR is the device name for the RM02 disk.

   MM is the device name for TE16, TU16, and TU45 magnetic tapes.
4. Terminate the response with a carriage return.

Refer to Section 2.2 for starting up the system.

2.1.2 Bootstrapping a PDP-11/70

Perform the following steps to activate the bootstrap on the PDP-11/70 processor.

1. Press the HALT switch.
2. Enter the startup address of 17765000 into the console switches.

3. Press the LOAD ADDR switch.
4. Set the console switches to the device code.

Device	Code
RM03 disk	0000001n
TE16/TU16/TU45 magnetic tape	0000006n
RP04/RP05/RP06 disk	0000007n

   The value of n is the device unit number where the system finds the hardware-bootstrapped system image.
5. Reset the HALT switch.
6. Press the START switch.
7. The bootstrap performs processor tests, instruction and addressing tests, and memory and cache tests.
8. Refer to Section 2.2 for starting up the system.

If the bootstrap tests detect a hardware failure, the bootstrap halts. If a failure occurs, call the local DIGITAL field service engineer. However, you may continue with the bootstrap operation when the following console light codes appear. These codes indicate a cache failure.

17773644
17773736
17773764

To continue the bootstrap operation with an indicated cache failure, press the CONT switch. Refer to Section 2.2 for starting up the system.

2.2 STARTING UP THE SYSTEM

If the system contains the driver for a device, not in the host configuration, a message appears on the console terminal. These units are declared OFFLINE by the bootstrap code.

The system mounts the system disk, reads the [1,2] STARTUP.CMD file, and enters into an interactive mode.

NOTE

Do not alter the [1,2] STARTUP.CMD file contents.

The dialog proceeds as follows:

1. Target System Name?
   The system is requesting the name of the system image which you booted. Respond with a 6-character name.

2. Time?
   The system is requesting the time of day. Respond with the current time in the hh:mm format where:

   hh is the hour using the 24 hour format
   mm is the number of minutes after the hour.

3. Date?
   The system is requesting the date. Respond with the current date in the dd-mmm-yy format, where:

   dd is the day of the month
   mmm is the 3-letter abbreviation of the month
   yy is the last 2 digits of the year.

The system first runs the INIT utility after which it is ready to accept DCL commands to start up the transaction processor. Refer to Chapter 3 in this guide to start up a transaction processor. When the system runs the INIT utility, it always:

1. Runs ERRLOG – the hardware error log utility.

   The ERRLOG utility creates the [1,6] ERR.TMP file if it does not exist, and writes the system configuration information into the file.
2. Runs the TPINIT utility for crash recovery.
3. Sets the system disk (SY0:) to the public status.
4. Allocates checkpoint space on the system disk.
5. Starts the batch and print facilities by running the [1,2] QMGSTART.CMD file.

   The [1,2] QMGSTART.CMD command file contains the DCL commands to initialize the print and batch facilities. The system delays the assignment of the BAP0 batch processor to the BATCH queue to ensure that the batch processor runs the [1,2] sys-name.BAT file immediately after the system assigns the processor. If you wish to change the initialization, simply modify the command file contents such as by adding more processors.

   NOTE

   Do not assign BAP0 to any other queue at system startup.

6. Submits a user batch job.

   If the user [1,2] sys-name.BAT file exists, the system submits the file to the batch processor. The variable filename, sys-name, must be identical to the target system name. You can use the file to change the initialization of devices, terminals, etc. Refer to the *Support Environment User's Guide* for creating a batch file.
7. Sends a message to all support terminals that the system is operating.

When appropriate, the utility:

1. Initializes the disk to overlap seeks
2. Initializes the Application Terminal Support (ATS)
3. Runs the software error logger, SERLOG
4. Runs the journaller.

If the INIT utility detects a severe error, such as a write-protected system disk, the INIT utility displays an error message. Press CTRL/C to cause an orderly termination of the INIT utility. Correct the error and issue the @[1,2] STARTUP.CMD command to retry the startup sequence.

NOTE

The application of CTRL/C to the INIT utility is a special case.

The INIT utility attaches the terminal. Normally, you would use CTRL/Z to terminate a task abruptly. However, the INIT utility ignores CTRL/Z since the INIT utility requires a graceful termination. When you issue a CTRL/C, the INIT utility intercepts the command and performs an orderly shutdown, such as deallocating checkpoint space and enabling logins.

2.3 SHUTTING DOWN THE SYSTEM (SHUTDOWN)

The privileged SHUTDOWN utility provides an orderly shutdown of the system and can run in a minimum support environment. You can access this utility either directly with the RUN $SHUTDOWN command or indirectly with the SETUP utility. Upon completion of the utility, the system is either halted in the case of direct access (that is, RUN $SHUTDOWN) or rebooted in the case of an indirect access.

2.3.1 Running the SHUTDOWN Utility

You run the SHUTDOWN utility with the command:

```
RUN $SHUTDOWN
```

The SHUTDOWN utility dialog proceeds as follows:

1. Minutes until start of transaction processor shutdown procedure <5>?

   Respond with the number of minutes the system is to wait until starting to shut down the transaction processors.
2. Minutes until transaction aborts <10>?

   Respond with the number of minutes that the system waits, after initiating the shutdown of the transaction processors, to abort active transactions.

3. Minutes until system shutdown <0>?

   Respond with the number of minutes that the system waits, after the termination of transaction processing, to shut down the system.

After completing the dialog for the SHUTDOWN utility, the system:

1. Sends the warning shutdown message
2. Stops and removes all transaction processors
3. Disables further logins
4. Issues the LOGOUT command for most logged in support terminals

   When logging out other support terminals, the system aborts all active, nonprivileged tasks.
5. Displays a termination message at each of the terminals the SHUTDOWN utility logs out
6. Runs the commands in the [1,2] SHUTUP.CMD file

   The [1,2] SHUTUP.CMD file contains the DCL commands to shut down the batch and spool facilities. You may modify the file contents to shut down additional batch and spool facilities that you initialized in the [1,2] QMGSTART.CMD file.
7. Dismounts all devices after the remaining active tasks become inactive.

2.4 USING THE SETUP UTILITY

The privileged SETUP utility provides you with four general services to:

1. Complete the system generation procedure.

   The SETUP utility functions to complete the system generation procedure are explained in the *System Generation Manual.*
2. Permit you to initiate execution of a different system image.

   The system disk can contain more than one system image. However, all the system images must have the identical hardware configuration. At any time, only one system image can be hardware-bootstrapped. The hardware-bootstrapped system image is the system image you execute when following the procedures described in Section 2.1. Other system images can be booted into the system with the BOOT function of the SETUP utility. You can change the hardware bootstrap to load another system image with the HARDSAVE function of the SETUP utility.
3. Provide you with several system maintenance facilities.

   For instance, you can add or delete user access to the support environment and generate a report of kernel errors.
4. Recreate sections of the system disk.

   The SETUP functions to perform these functions are presented in Chapter 9 in this guide.

Refer to Section 2.4.1 for a general description of invoking the SETUP utility and the SETUP utility functions. The remaining sections explain the SETUP functions you use to boot another system image and to save a changed system image.

2.4.1 Selecting a SETUP Utility Function

You invoke the privileged SETUP utility, in a transaction-processing environment with a full support environment, with the command:

```
RUN $SETUP
```

Before you invoke the privileged SETUP utility in a transaction-processing environment with a minimum support environment, you must first remove any transaction processors from the TP1PAR partition. Use the SHOW PARTITIONS command to determine if the TP1PAR partition contains any transaction processors. Use the TPCTRL utility to remove a transaction processor from the TP1PAR partition. You may then invoke the privileged SETUP utility with the command:

```
RUN/PARTITION:TP1PAR $SETUP
```

The SETUP utility responds with the question:

```
Function <EXIT>?
```

The SETUP functions are:

1. VERSION
   If you select the VERSION function, the utility prints the version and name of the system currently running. The VERSION function identifies the system software revision level. When you submit a SPR, you are required to identify your system revision level. This information helps the DIGITAL personnel to recreate and correct the problem.
2. BOOT
   If you select the BOOT function, you can perform a software-boot to change the current system image to another system image. You can software boot any system image after system generation. The BOOT function is discussed in Section 2.4.2.
3. SAVE
   If you select the SAVE function, you can perform a system save, without changing the hardware bootstrap. The memory resident system image is written over the file from which it was originally booted. The SAVE function is discussed in Section 2.4.3.
4. HARDSAVE
   If you select the HARDSAVE function, you can perform a system save and change the hardware bootstrap to point to the new system image. The HARDSAVE function is discussed in Section 2.4.4.

5. SETTIME
   If you select the SETTIME function, you can change the time and date. The SETTIME function is discussed in Section 2.4.5.
6. ERRORLOG
   If you select the ERRORLOG function, you can print a report of all hardware errors detected by the kernel error logger. The ERRORLOG function is discussed in Chapter 9 in this guide.
7. ACCOUNT
   If you select the ACCOUNT function, you can add, modify, delete, and list user accounts. The ACCOUNT function is discussed in Chapter 9 in this guide.
8. TERMINAL
   If you select the TERMINAL function, you can set up or change the support environment terminal specifications file located on the system disk. Use the SET command to immediately change the terminal specifications on-line. The TERMINAL function is discussed in Chapter 9 in this guide.
9. REDO
   If you select the REDO function, you can incorporate the terminal changes on the disk-resident system image. The REDO function is discussed in Chapter 9 in this guide.
10. TRANSFER
    If you select the TRANSFER function, you can move user files from an old system disk to a new system disk. The TRANSFER function is discussed in Chapter 9 in this guide.
11. BAD
    If you select the BAD function, you can perform bad block-checking on a disk. The BAD function is discussed in Chapter 9 in this guide.
12. RENEW
    If you select the RENEW function, you can recreate a disk-resident system image from existing files. The RENEW function may be required if the integrity of a disk-resident system image is destroyed. The RENEW function is discussed in Chapter 9 in this guide.
13. STARTUP
    If you select the STARTUP function, you can recreate the system initialization commands executed for you by the INIT utility. The STARTUP function is discussed in Chapter 9 in this guide.
14. PARTITION
    If you select the PARTITION function, you can set up or change the partition specifications file located on the system disk. The PARTITION function is discussed in the *System Generation Manual.*
15. RETRY SYSGEN
    If you select the RETRY SYSGEN function, you can recreate from the most recent SYSGEN procedure the system generation environment and recreate a new system. Use the RETRY

SYSGEN function if the SYSGEN function of the SETUP utility was unable to finish. Correct the problem and use the RETRY SYSGEN function. The RETRY SYSGEN function is discussed in the *System Generation Manual.*

16. RETRY UPDATE
    If you select the RETRY UPDATE function, you can recreate the CHGMAKER environment from the most recent CHGMAKER procedure which incorporates DIGITAL-supplied software changes into your current system. The RETRY UPDATE function is similar to the RETRY SYSGEN function. The RETRY UPDATE function is discussed in the *System Generation Manual.*
17. DEV
    If you select the DEV function, you can create a support environment-only system from information derived from the most recent SYSGEN procedure. The support environment-only system contains the same kernel as the newly created transaction-processing environment. The DEV function is discussed in the *System Generation Manual.*
18. EXIT
    If you select the EXIT function, you can terminate the SETUP utility.

2.4.2 Booting a System

To terminate execution of the existing system image and execute another system image, use the BOOT function of the SETUP utility. After you initiate the BOOT function, it:

1. Runs the SHUTDOWN utility.
   Refer to Section 2.3 for a description of the SHUTDOWN utility.
2. Bootstraps another system image.

Refer to Section 2.2 to complete the bootstrap operation.

After you select the BOOT function as described in Section 2.4.1, the function responds with the following dialog:

1. What file should be booted in <TRAX>?
   You respond with the file specification of the system image to be booted. You must specify a .SYS filetype. If you omit a portion of the file specification, the system uses a default value. The complete default file specification is SY:[1,54] TRAX.SYS.
   If the BOOT is successful, you never reach Question 2. Refer to Section 2.2 for starting up the system.
2. Do you want to try to BOOT again <YES>?
   If the BOOT is not successful, the system displays Question 2. If you wish to retry the BOOT function, correct the problem and

answer YES. If you answer NO, the system initializes itself by running the INIT utility and re-establishes the existing system image, which was active when you executed the BOOT function.

The hardware bootstrap initially points to the system generation baseline system image, [4,54] BASLIN. If you have not run the HARDSAVE function of the SETUP utility to change the hardware bootstrap, run the RENEW function of the SETUP utility to recover the system image which does not boot. If you have changed the hardware bootstrap and the general partition is too small to run SETUP and you have no other system images available, boot the [4,54] BASLIN system image and run the RENEW function.

2.4.3 Saving and Not Changing the Hardware Bootstrap

After you select the SAVE function (Section 2.4.1), it writes the resident system image into the file from which the file was originally booted. Thus you can make your changes to the system image permanent. The SAVE function:

1. Runs the SHUTDOWN utility.

   Refer to Section 2.3 for a description of the SHUTDOWN utility.
2. Verifies that the system is inactive by checking that:
   A. There are no tasks with outstanding I/O
   B. There are no mounted devices
   C. There are no active checkpoint files
   D. The error logger is turned OFF
   E. There are no active transaction processors
   F. The queue manager is not active.

   The system reports an error if any of these checks fail.
3. Saves the system by over-writing the file from which the system image was booted.
4. Boots the saved system image.

Refer to Section 2.2 to complete the bootstrap operation.

If the SAVE is successful, refer to Section 2.2 for starting up the system. If the SAVE is not successful, the SAVE function displays the question:

Do you want to try to SAVE again <YES>?

You may retry the SAVE function by correcting the problem and answer YES to the qeustion. If you answer NO to the question, the system initializes itself by running the INIT utility and re-establishes the existing system image, which was active when you executed the SAVE function.

2.4.4 Saving and Changing Hardware Bootstrap

After you select the HARDSAVE function (Section 2.4.1), it writes the resident system image into the file from which it was originally booted, and changes the hardware bootstrap to point to this new system image. The HARDSAVE function:

1. Runs the SHUTDOWN utility.

   Refer to Section 2.3 for a description of the SHUTDOWN utility.
2. Verifies that the system is inactive by checking that:
   A. There are no tasks with outstanding I/O
   B. There are no mounted devices
   C. There are no active checkpoint files
   D. The error logger is turned OFF
   E. There are no active transaction processors
   F. The queue manager is not active.

   The system reports an error if any of these checks fail.
3. Bootstraps the saved system image.

Refer to Section 2.2 to complete the bootstrap operation.

If the HARDSAVE is successful, refer to Section 2.2 for starting up the system. If the HARDSAVE is not successful, the system displays the question:

Do you want to try to HARDSAVE again <YES>?

You may retry the HARDSAVE function by correcting the problem and answer YES to the question. If you answer NO to the question, the system initializes itself by running the INIT utility and re-establishes the existing system image, which was active when you executed the HARDSAVE function.

2.4.5 Changing Time and Date

Use the SETTIME function of the SETUP utility to change the system time and date. After you select the SETTIME function (Section 2.4.1), it responds with the following dialog:

1. Time?

   The system is requesting the time of day. Respond with the current time in the hh:mm format where:

   hh is the hour using the 24-hour format
   mm is the number of minutes after the hour.
2. Date?

   The system is requesting the date. Respond with the current date in the dd-mmm-yy format, where:

   dd is the day of the month
   mmm is the 3-letter abbreviation of the month
   yy is the last 2 digits of the year.

2.5 CRASH RECOVERY AND RESTART

The system automatically recovers from 3 types of system failures:

1. A crash of a single transaction processor
2. A crash resulting from a transient system failure
3. A kernel crash.

When a single transaction processor fails, the system restarts it if the crash recovery option is specified for that transaction processor. The system cleans up all inprogress transactions:

1. Completes transactions in the end status
2. Aborts the transactions.

If you did not select the transaction processor crash recovery option, the system does not restart.

After a transaction processor crash, it is recommended that you run either the SERANL or SERDAY utility to provide an error log report for the transaction processor. Refer to Chapter 6 in this guide to use these utilities.

For a transient system failure (such as a power failure), the system saves volatile information. After the transient failure is corrected, the system restores the volatile information. For a kernel crash, the system dumps all of memory into a crash dump file and reloads the existing system. All active transactions are aborted as explained for a transaction processor crash. It is recommended that you run the ANALYSIS utility after a kernel crash to provide a report based upon the crash dump file. Refer to Chapter 9 in this guide to use the ANALYSIS utility.

Use the RECOVR utility, described in Chapter 5 in this guide, to recover disk data files inadvertently corrupted in a disk crash.

CHAPTER 3
MANAGING TRANSACTION PROCESSORS

A transaction processor is defined by utilities which are described in the *Application Programmer's Guide.* If journalling will be needed by any transaction processor, assign a journal device to the journal before you start any transaction processors. Refer to Section 3.1 to assign a journal device. If statistics are to be gathered, run the TPSTAT utility after you install a transaction processor. Refer to Chapter 5 in this guide to run the TPSTAT utility.

You install, start, stop, and remove the transaction processors with the TPCTRL utility which is described in Section 3.2. Once installed, a transaction processor can be changed with modification utilities which are described in Chapter 5.

The control and journal utilities are summarized in Table 3-1.

Table 3-1 Transaction Processor Utility Functions

Utility	Function
AUXSTP	The AUXSTP utility stops the journal activities for all transaction processors and the software error logger, SERLOG. Refer to Section 3.1.1 for information on the AUXSTP utility.
DSKINT	The DSKINT utility initializes a disk volume as a journal volume. Refer to Section 3.1.3 for information on the DSKINT utility.
SYSMOD	The SYSMOD utility displays and defines global system characteristics, such as journalling. Refer to Chapter 5 for information of the SYSMOD utility.
TRCTRL	The TPCTRL utility starts and stops a transaction processor and enables or disables recording activities to the journal. Refer to Section 3.1 for information on the journal activities and to Section 3.2 for information on the TPCTRL utility.
TPMOD	The TPMOD utility displays and modifies data used by an installed transaction processor. Refer to Chapter 5 for information on the TPMOD utility.

3.1 CONTROLLING THE JOURNAL

The TRAX journal file contains two record types. The first record type is called a transaction slot record and provides for recovery of the transaction data base. The second record type is called a log record and provides for general logging purposes.

The transaction slot record is an exact image of the transaction slot at the time the transaction ended. All information needed to recreate the changes to the application data file set is included in the transaction slot record. Use the RECOVER utility to recreate the changes to the application data file set on an archived volume.

A TST produces the log record which contains user defined information. Flexibility is provided to create up to 26 different types of log records using alphabetic identifiers. Use the SHOLOG utility to selectively report the journal log records.

The system writes the transaction slot records and log records to either a disk or magnetic tape journal volume. Through the FILDEF and TRADEF utilities, you specify the transaction processor journal activities for writing to the journal on a file-by-file basis.

Before you start the journal activities, you must define and ready the journal device(s). You define to the system one or two journal devices with the SYSMOD utility and ready the device(s) by loading the magnetic tape or disk journal volume(s). The system formats and mounts a magnetic tape volume after you load it. You must format, load, and mount a disk journal volume. Refer to Section 3.1.3 to format a disk journal volume and use the DCL MOUNT command to mount the disk journal volume. After journalling is properly set up, any transaction processor with journalling requirements will successfully be able to perform them.

When the system fills a journal volume, the system:

1. Switches to the secondary journal device
2. Displays a full journal volume message on the console terminal
3. Closes the full journal file
4. Dismounts the full volume.

You must then replace the volume with a new volume. If you defined only one journal device or if the second device is not ready, the system:

1. Displays a full journal volume message on the console terminal
2. Closes the full journal file
3. Dismounts the full volume
4. Delays all journal requests until you replace the volume with a new volume.

3.1.1 Assigning Journal Devices (SYSMOD)

Use the SYSMOD utility to define the one or two journal devices to the system. Refer to Chapter 5 in this guide to run the SYSMOD utility.

When you use the TPCTRL utility to start a transaction processor, you also inform the system to enable journalling and/or logging for the transaction processor. After you start a transaction processor, the system writes to the first device you assigned. When it becomes full, the system displays a device full message on the console terminal and begins writing to the other journal. See Section 3.1.4 for journal error message descriptions. Replace the full journal while the system is writing to the other journal. If you assign only one journal device, the system automatically halts when the journal is full and continues operations after you replace the volume.

3.1.2 Stopping the Journal (AUXSTP)

Use the AUXSTP utility to stop the journal activities and the software error logger without shutting down the system. You can use the AUXSTP utility in a minimum support environment. You also indirectly use the AUXSTP utility when you shut down the system, which is explained in Chapter 2. You invoke the AUXSTP utility with the command:

```
RUN $AUXSTP
```

The utility then:

1. Closes the journal file
2. Dismounts the volume
3. Removes the journal device definitions from the system.

Until you start the journal activities again with a new journal volume, the system rejects all requests to write to a journal.

3.1.3 Formatting a Disk Journal Volume (DSKINT)

To initialize a disk journal, use the DSKINT utility in a full support environment. You invoke the DSKINT utility with the following command:

```
RUN $DSKINT
```

The utility displays:

```
Specify device and volume?
```

You respond with a device name and the volume identification in the format:

```
Ddnn:volume-id
```

where:

Ddnn:	is the device name containing the volume such as *DB1:* for an RP06.

volume-id: is the name DSKINT utility writes on the volume. The name can be up to nine alphanumeric characters such as JOURNAL 1.

Once you complete the dialog, the DSKINT utility:

1. Finds all the bad blocks on the volume
2. Initializes the volume and assigns the volume identification
3. Mounts the volume
4. Creates a journal file the size of the volume avoiding all the bad blocks
5. Dismounts the volume.

Once a volume is formatted, you can use the volume any number of times as a journal volume.

3.1.4 Displaying Journal Error Messages

The error messages presented in this section result from the system trying to write to the journal.

Hard Device Error on Journal Device <device>
An unrecoverable device error was encountered while writing to the journal. In this case the system switches to the other journal device, if you assigned one, and retries the operation. The volume mounted on the erroneous device is still usable for recovery purposes and therefore remains part of a journal volume set. The RECOVR utility makes extensive checks to ensure the validity of all recorded journal data.

Journal Device Write Locked <device>
The system determined that the journal volume is write-locked and waits 15 seconds for you to write-enable the volume. At the end of the 15 seconds, the system retries the operation.

Journal Device not ready <device>
The system determined that the journal device is not ready and waits 15 seconds for you to ready the device. At the end of the 15 seconds, the system retries the operation.

Please Remove Journal <device>
The system filled the journal and dismounted the volume of the device indicated in the message. Remove the volume from the device.

Mount New Journal Media on <device>
When the system switched to the new journal device, the system determined that you did not mount the volume. The system waits 15 seconds for you to mount the volume and then the system retries the operation.

3.2 CONTROLLING TRANSACTION PROCESSORS (TPCTRL)

Use the TPCTRL utility in a minimum support environment to install, start, stop, and remove a transaction processor. You invoke the TPCTRL utility with the command:

```
RUN $TPCTRL
```

The TPCTRL dialog begins with the question:

1. Command <BRIEF>?

   The TPCTRL commands are:

   A. INSTALL
      The INSTALL command allocates and initializes the resources for a transaction processor. In addition, the INSTALL command performs consistency checks and saves the install information for use by the BRIEF command. Refer to the INSTALL Command section for a description of the dialog.

   B. BRIEF
      The BRIEF command allocates the resources for a transaction processor after the consistency checks without undergoing a full install. If you change any definitions, use the INSTALL command rather than the BRIEF command. Refer to the BRIEF Command section for a description of the dialog.

   C. START
      The START command readies an installed transaction processor for terminal input. Refer to The START Command section for a description of the dialog.

   D. STOP
      The STOP command terminates the execution of a started transaction processor. Refer to The STOP Command section for a description of the dialog.

   E. REMOVE
      The REMOVE command frees the resources of an installed transaction processor. Refer to The REMOVE Command section for a description of the dialog.

   F. EXIT
      The EXIT command terminates the TPCTRL utility.

If you run the TPCTRL utility in a batch stream, TPCTRL returns most errors to the batch processor as a "SEVERE ERROR". All TPCTRL errors are fatal unless otherwise stated in the chapter.

The remaining sections describe the dialog for each TPCTRL command. A sample display sequence appears as:

```
RUN $TPCTRL
Command <BRIEF>? INSTALL
Transaction processor name? TPTEST
```

```
Partition <TPIPAR>? TPIPAR
Trace transaction processor <NO>? YES
Write protect data base <NO>? YES
Hihg-Performance RMS <YES>? YES
Forms definition file version <LATEST>? 2
Command <START>? START
Transaction processor name <TPTEST>? TPTEST
Enable journalling <YES>? NO
Enable logging <NO>? NO
Command <EXIT>? EXIT
```

3.2.1 Installing a Transaction Processor (INSTALL)

The INSTALL command identifies a transaction processor and its components to the system. This command allocates the resources required by the transaction processor and accesses or creates the files for the transaction processor to run. Also the command builds system tables based on data in files and does consistency checks. Results are saved on the disk for use with the BRIEF command.

Table 3-2 lists the files created or accessed by the INSTALL command. Throughout Table 3-2 and the remainder of the chapter, "tpname" is a symbolic transaction processor name. The only requirements for a transaction processor name are that it be from 1- to 6-alphanumeric characters and it must be defined.

Table 3-2 Files Accessed by INSTALL

File	Description
[1, 1] TPDEF. TPF	The TPDEF. TPF file is a transaction processor definition file created at system generation. The TPDEF utility creates a record, within the TPDEF. TPF file, for each transaction processor that you define. You use the INSTALL command to update the corresponding transaction processor record. The INSTALL command performs the following: 1. Sets the install status indicator 2. Enters the name of the partition where the transaction processor resides 3. Enters a pointer to the address of the partition.
[1, 10] tpname. TRC	The tpname. TRC file is a transaction processor trace file. The START command creates the file if you specify YES to the trace question in

Table 3-2 (Cont.) Files Accessed by INSTALL

File	Description
	the INSTALL command dialog. The TPTRAC utility, described in the *Application Programmer's Guide*, provides reports from the tpname. TRC file.
[1, 300] tpname. TSK	The contiguous tpname. TSK task image file contains the common area information for the transaction processor. The TPDEF utility updates the tpname. TSK file which the INSTALL command uses to load the initialize the common area for the transaction processor.
[1, 300] tpname. STA	The tpname. STA file is a station definition file for initializing station characteristics. The STADEF utility updates the tpname. STA file which the INSTALL command uses to initialize the stations.
[1, 300] tpname. TRA	The tpname. TRA file is a transaction definition file for initializing transaction characteristics. The TRADEF utility updates the tpname. TRA file which the INSTALL command uses to initialize transactions.
[1, 300] tpname. FIL	The tpname. FIL file is a user files definition file. The FILDEF utility updates the tpname. FIL file which the INSTALL command uses to initialize the file processor.
[1, 300] tpname. WOR	The tpname. WOR file is a work class definition file. The WORDEF utility updates the tpname. WOR file which the INSTALL command uses to define work classes.
[1, 300] tpname. AUT	The tpname. AUT file is a user authorization file. The AUTDEF utility updates the tpname. AUT file which the INSTALL command uses to determine which users (with correct password) are permitted to access a transaction processor.

Table 3-2 (Cont.) Files Accessed by INSTALL

File	Description
[1, 300] tpname. FDF	The tpname, FDF file is a forms definition file. The ATL utility updates the tpname. FDF file which the INSTALL command uses to initialize the forms information.
[1, 10] tpname. FMX; 1	The contiguous tpname. FMX file contains a copy of the forms definition file. The INSTALL command creates the tpname. FMX file which the transaction processor uses while the transaction processor runs. If the file already exists, the INSTALL command supersedes the old version (that is, tpname. FMX; 1).
[1, 300] tpname. TIM; 1	The contiguous tpname. TIM task image file contains the code for controlling application terminals. The TPDEF utility creates the tpname. TIM file which the INSTALL command loads into memory.
[1, 10] tpname. TDB; 1	The contiguous tpname. TDB file contains transaction description information needed while the transaction processor runs. The INSTALL command creates the tpname. TDB file. If the tpname. TDB file already exists, the INSTALL command supersedes the old version (that is, tpname. TDB;1).
[1, 10] tpname. WRK; 1	The tpname. WRK file is a system workspace file created by the INSTALL command. If the tpname. WRK file already exists, the INSTALL command supersedes the old version (that is, tpname. WRK; 1).
[1, 10] stanam. typ; 1	The "stanam" and "typ" are a symbolic filename and filetype. The file is a mailbox file created by the INSTALL command. The filename, stanam, is the mailbox station name and the filetype, typ, is the unique identifier associated with the transaction processor.

Table 3-2 (Cont.) Files Accessed by INSTALL

File	Description
[1, 1] tpname. BRF; 1	The tpname. BRF file contains the BRIEF command information. At the end of the INSTALL command operation, the system writes the initialized data structures to the tpname. BRF file. The BRIEF (install) command uses the tpname. BRF file to provide a faster installation than you achieved with the INSTALL command. The BRIEF command installs the transaction processor with the same characteristics that you specified with the INSTALL command dialog. If the tpname. BRF file already exists, the INSTALL command supersedes the old version (that is, tpname. BRF; 1). You can modify the tpname. BRF file with the TPMOD utility.

The INSTALL command dialog proceeds as follows:

1. Command <BRIEF>? INSTALL
2. Transaction processor name <tpname>?

   Specify the name of the transaction processor for TPCTRL to install. A 1- to 6-alphanumeric character transaction processor name is required. The default name is the last correct transaction processor name that you entered. If no previous transaction processor name is specified, there is no default.

   If you:

   A. Specify either an illegal or a nonexistent transaction processor name, TPCTRL displays an error message and repeats Question 2.

      Respecify the transaction processor name.

   B. Specify an installed transaction processor. TPCTRL prints an error message and returns to Question 1 with a new default:

      Command <START>?

      Refer to the START command description.

   C. Specify a transaction processor which is started. TPCTRL prints an error message and returns to Question 1 with a new default:

      Command <EXIT>?

      Terminate the TPCTRL utility.

D. Specify a transaction processor which another user is accessing with a TPCTRL command. TPCTRL declares the your command an error and returns to Question 1:

Command <EXIT>?

Only one user at a time should run TPCTRL. Choose another transaction processor or terminate the TPCTRL utility.

TPCTRL accesses the [1, 1] TPDEF. TPF file to verify the characteristics of a transaction processor. If the [1, 1] TPDEF. TPF file is either inaccessible or nonexistent, TPCTRL prints an error message and returns to Question 1 with a new default:

Command <EXIT>?

Either terminate the TPCTRL utility or attempt the access again.

NOTE

The [1, 1] TPDEF. TPF file may be temporarily inaccessible (for example, locked by the TPDEF utility).

3. Partition <TPIPAR>?

   Specify an inactive transaction processor partition where TPCTRL can install the transaction processor.

   TPCTRL displays an error message and repeats Question 3 if you specify:

   A. An invalid partition name. The partition name format is TPxPAR, where x is a value from 1 to 8.
   B. An active partition.
   C. A nonexistent partition.

4. Trace transaction processor <NO>?

   If you specify YES, the flow of the transaction processor is recorded in the file [1, 10] tpname. TRC (where tpname is the name of the transaction processor). Continue with Question 5.

   If you specify NO, the flow is not recorded. Continue with Question 6.

5. Write protect data base <NO>?

   If you specify YES, updates are not made to the application data base. If you specify NO, updates are made to the application data base.

6. High-Performance RMS <YES>?

   If you specify YES, TPCTRL installs the data manager with high performance memory resident overlays.

   If you specify NO, TPCTRL installs the data manager with disk-resident overlays. The disk-resident overlays use less memory, but require more time to access than the nonresident overlays.

7. Forms definition file version <LATEST>?

   Specify either "LATEST" or a version number, from 1 to 77777, of the [1, 300] tpname. FDF Forms Definition File to be used. If you specify "LATEST", the system accesses the highest version listed in the directory.

   If you specify a nonexistent tpname. FDF file version or an invalid version number, TPCTRL prints an error message and repeats Question 7.

   If there is no [1, 300] tpname. FDF, TPCTRL prints an error message and returns to Question 1 with a new default:

   Command <EXIT>?

TPCTRL ends the INSTALL dialog with Question 7. The TPCTRL utility detects for inconsistencies and displays an error message when they are found.

If the INSTALL command is successful, TPCTRL installs, allocates, and initializes the transaction processor. Control returns to Question 1 with a new default:

Command <START>?

Refer to the START command description.

3.2.2 Installing a Transaction Processor (BRIEF)

The BRIEF command installs the transaction processor with the same characteristics that you specified with the most recent INSTALL command dialog for a transaction processor. If you make any changes by using the definition utilities, ATL, or the TSTBLD utility, you must use the INSTALL command; the BRIEF command cannot install a transaction processor which you modify.

The BRIEF command accesses the TPDEF, TPF file:

1. Sets the install status indicator
2. Accesses the name of the partition where the transaction processor resides
3. Compares the address of the partition previously specified to the current partition.

The BRIEF command reads the remaining files described in Table 3-2.

The BRIEF command dialog proceeds as follows:

1. Command <BRIEF>? BRIEF
2. Transaction processor name <tpname>?

   Specify the name of the transaction processor for TPCTRL to install. A 1- to 6-alphanumeric character transaction processor name is required. The default name is the last correct transaction

processor name that you entered. If no previous transaction processor name is specified, there is no default. If you specify a valid response, TPCTRL installs the transaction processor and returns to Question 1 with a new default:

Command <START>?

Refer to the START command description.

If you:

A. Specify either an illegal or a nonexistent transaction processor name, TPCTRL displays an error message and repeats Question 2.

B. Specify an installed transaction processor, TPCTRL prints an error message and returns to Question 1 with a new default:

Command <START>?

Refer to the START command description.

C. Specify a transaction processor which is started, TPCTRL prints an error message and returns to Question 1 with a new default:

Command <EXIT>?

Exit the TPCTRL utility.

D. Specify a transaction processor which another user is accessing with a TPCTRL command, TPCTRL declares the your command an error, and returns to Question 1:

Command <EXIT>?

Only one user at a time should run TPCTRL. Choose another transaction processor or terminate the TPCTRL utility.

TPCTRL accesses the [1,1]TPDEF.TPF file to verify a transaction processor characteristics. If the [1,1] TPDEF.TPF file is either inaccessible or nonexistent, TPCTRL prints an error message and returns to Question 1 with a new default:

Command <EXIT>?

Either terminate the TPCTRL utility or attempt the access again.

NOTE

The [1,1]TPDEF.TPF file may be temporarily inaccessible (for example, locked by the TPDEF utility).

If the system finds, as a result of a previous installation:

1. A nonexistent or inaccessible [1,1] tpname.BRF;1 file, TPCTRL prints an error message and returns to Question 1 with a new default:

   Command <INSTALL>?

   Refer to the INSTALL command description.
2. A transaction processor with partition characteristics different from those of the partition in which it was installed, TPCTRL prints an error message and returns to Question 1 with a new default:

   Command <INSTALL>?

   Refer to the INSTALL command description.
3. An active partition, TPCTRL prints an error message and returns to Question 1 with a new default:

   Command <INSTALL>?

   Refer to the INSTALL command description.

3.2.3 Starting a Transaction Processor (START)

The START command begins running an installed transaction processor and sets the start status in the transaction processor record of the TPDEF.TPF file.

The START command dialog proceeds as follows:

1. Command <START> ? START
2. Transaction processor name <tpname> ?

   Specify the name of the transaction processor for TPCTRL to start. A 1- to 6- alphanumeric character transaction processor name is required, the default name being the last correct transaction processor name that you entered. If no previous transaction processor name is specified, there is no default.

   If you:

   A. Specify either an illegal or a nonexistent transaction processor name, TPCTRL displays an error message and repeats Question 2. Respecify the transaction processor name.
   B. Specify a transaction processor which is running, TPCTRL prints an error message and returns to Question 1 with a new default:

      Command <EXIT>?

   C. Specify a transaction processor which is not installed, TPCTRL prints an error message and returns to Question 1 with a new default:

      Command <BRIEF>?

      Refer to the BRIEF command description.

D. Specify a transaction processor which another user is accessing with a TPCTRL command. The TPCTRL utility declares your command an error and returns to Question 1:

Command <EXIT>?

Only one user at a time should run TPCTRL. Choose another transaction processor or terminate the TPCTRL utility.

TPCTRL accesses the [1, 1] TPDEF. TPF file to verify the characteristics of a transaction processor. If the [1, 1] TPDEF. TPF file is either inaccessible or nonexistent, TPCTRL prints an error message and returns to Question 1 with a new default:

Command <EXIT>?

Either terminate the TPCTRL utility or attempt the access again.

NOTE

The [1, 1] TPDEF. TPF file may be temporarily inaccessible (for example, locked by the TPDEF utility).

3. Enable journalling <default>?

   The default response is either NO if you did not define to the system a journal device with the SYSMOD utility, or YES if you did define a journal device.

   If you specify YES, journal activities are enabled for the transaction processor.

   The system displays an error message and repeats the question if you did not define a journal device. You can either respond with NO and continue with the TPCTRL utility or terminate TPCTRL by pressing CTRL/Z. Use the SYSMOD utility to define a journal device, and again run TPCTRL to start the transaction processor and the journal activities.

   If you specify NO, journal activities are disabled for the transaction processor.

4. Enable logging <NO>?

   If you specify YES, logging is enabled for the transaction processor.

   The system displays an error message and repeats the question if you did not define a journal device. You can either respond with NO and continue with the TPCTRL utility or terminate TPCTRL by pressing CTRL/Z. Use the SYSMOD utility to define a journal device, and then again run TPCTRL to start the transaction processor and the journal activities.

If you specify NO, logging is disabled for the transaction processor.

Question 4 is the final question for starting a transaction processor. TPCTRL starts the transaction processor and returns to Question 1 with a new default:

Command <EXIT>?

3.2.4 Stopping a Transaction Processor (STOP)

The STOP command halts a running transaction processor in an orderly fashion. The STOP command also resets the start status in the transaction processor record of the TPDEF. TPF file.

The STOP command dialog proceeds as follows:

1. Command <STOP>? STOP
2. Transaction processor name <tpname>?

   Specify the name of the transaction processor for TPCTRL to stop. You can specify either a 1- to 6-alphanumeric character transaction processor name or ALL. If you specify ALL, TPCTRL stops all running transaction processors. The default name is the last correct transaction processor name you entered. If no previous transaction processor name is specified, the default is ALL.

   If you:

   A. Specify either an illegal or a nonexistent transaction processor name, TPCTRL displays an error message and repeats Question 2.
   B. Specify an installed transaction processor which is not running, TPCTRL displays an error message and returns to Question 1 with a new default:

      Command <REMOVE>?

      Refer to the REMOVE command description.
   C. Specify a transaction processor which is not installed, TPCTRL prints an error message and returns to Question 1 with a new default:

      Command <EXIT>?

      Refer to the EXIT command description.
   D. Specify ALL and there are no running transaction processors, TPCTRL displays an error message and returns to Question 1 with a new default:

      Command <REMOVE>?

      Refer to the REMOVE command description.

      If TPCTRL is running in a batch stream, TPCTRL returns an error message to the batch processor. However, the error is not fatal.

E. A transaction processor which another user is accessing with a TPCTRL command, TPCTRL declares your command an error and returns to Question 1:

Command <EXIT>?

Only one user at a time should run TPCTRL. Choose another transaction processor or terminate the TPCTRL utility.

TPCTRL access the [1,1]TPDEF.TPF file to verify the characteristics of a transaction processor. If the [1,1]TPDEF.TPF file is either inaccessible or nonexistent, TPCTRL prints an error message and returns to Question 1 with a new default:

Command <EXIT>?

Either terminate the TPCTRL utility or attempt the access again.

NOTE

The [1,1]TPDEF.TPF file may be temporarily inaccessible (for example, locked by the TPDEF utility).

3. Minutes until transaction initiation is disabled <10>?

   Indicate the number of minutes, 0 to 15, until STOP is to be initiated. No more transactions may be initiated after STOP is initiated. If you exceed 15 minutes, TPCTRL displays an error message and repeats Question 3.

4. Minutes till incomplete processing is aborted <10>?

   TPCTRL aborts all transaction instances that remain active after the specified number of minutes following the issuance of the STOP. You can specify from 0 to 15 minutes.

   If you exceed 15 minutes, TPCTRL displays an error message and repeats the question.

   If you specify valid answers and the STOP command is successful, TPCTRL prints a message and returns to Question 1 with a new default:

   Command <REMOVE>?

   Refer to the REMOVE command description.

   If you specify valid answers and the system has not stopped the transaction processor within 5 minutes after aborting incomplete processing, the TPCTRL utility abruptly stops and removes the transaction processor. The TPCTRL utility displays an error message and returns to Question 1 with a new default:

   Command <EXIT>?

3.2.5 Removing a Transaction Processor (REMOVE)

The REMOVE command releases the memory occupied by a stopped transaction processor. The REMOVE command also resets the install status in the transaction processor record of the TPDEF.TPF file.

The REMOVE command dialog proceeds as follows:

1. Command <REMOVE> ? REMOVE
2. Transaction processor name <tpname>?

   Specify the name of the transaction processor for TPCTRL to remove. You must specify either a 6-alphanumeric character transaction processor name or ALL. The default name is the last correct transaction processor name that you entered. If no previous transaction processor name is specified, the default is ALL. If you specify a valid response, TPCTRL removes the transaction processor(s) and returns to Question 1 with a new default:

   Command <EXIT>?

   If you:

   A. Specify either an illegal or a nonexistent transaction processor name, TPCTRL displays an error message and repeats Question 2.

   B. Specify a transaction processor which is not installed, TPCTRL prints an error message and returns to Question 1 with a new default:

      Command <EXIT>?

      Refer to the EXIT command description.

   C. Specify a transaction processor which is running, TPCTRL prints an error message and returns to Question 1 with a new default:

      Command <STOP>?

      Refer to the STOP command description.

   D. Specify ALL and there is no transaction processor installed, TPCTRL prints a warning message and returns to Question 1 with a new default:

      Command <EXIT>?

      If TPCTRL is running in a batch stream, TPCTRL returns an error message to the batch processor. However, the error is not fatal.

   E. Specify a transaction processor which another user is accessing with a TPCTRL command, TPCTRL declares the second command an error and returns to Question 1:

      Command <EXIT>?

      Only one user at a time should run TPCTRL. Choose another transaction processor or terminate the TPCTRL utility.

TPCTRL accesses the [1,1] TPDEF.TPF file to verify the characteristics of a transaction processor. If the [1,1] TPDEF.TPF file is either inaccessible or nonexistent, TPCTRL prints an error message and returns to Question 1 with a new default:

Command <EXIT>?

Either terminate the TPCTRL utility or attempt the access again.

NOTE

The [1,1] TPDEF.TPF file may be temporarily inaccessible (for example, locked by the TPDEF utility).

CHAPTER 4
MANAGING BATCH AND SPOOL FACILITIES

When you start or boot the system, the system initializes and starts the batch and spool facilities. The batch facility accepts files whose contents are DCL commands and specific task commands, and provides a means of directing your commands as though you input them from a terminal keyboard. The spool facility includes any function in which the system sends print-formatted output to a printer. You can specify the line printer (LP:) and the spooled device (SP0:) for print-formatted output. If you do not specify a public file-structured device for input, the system writes the output to a temporary file on the SP: device and then spools the file to the printer. This chapter explains how to control the batch and spool facilities.

For further understanding of Chapter 4, the following glossary is provided.

Queue Manager
The queue manager controls the overall operation of the batch queues and print queues.

Queue
A queue stores jobs to be processed by an assigned processor.

Processor
A processor, which is assigned to a queue, performs the functions specified in the job.

Job
A job is a set of files that contain batch commands or print-formatted data.

File
A file is an organized set of data stored on disk or magnetic tape and identified with an alphanumeric name.

4.1 MANAGING QUEUES, PROCESSORS, AND JOBS

When you start or boot the system, the system runs a program called the queue manager to control the batch and spool facilities (that is, queues, processors, and the queue file). A queue is a list of jobs to be run while a job is a set of files associated with a command. For instance, the files you specify with one SUBMIT command represent one job. A processor is a task for running a job. The queue file contains a list of jobs, queues, and processors.

You can stop and start all the batch and spool facilities by directing the queue manager operations with DCL commands. For instance, when you use the STOP command to stop the queue manager, you also stop all the queues and processors. Refer to Section 4.2 to stop and start the queue manager.

When the queue manager is running, it scans the queues for idle processors which are idle at system startup or when a processor completes a job. The queue manager passes to an idle processor the highest priority job with attributes matching those attributes of the processor, such as the type of form to be printed. Once the queue manager passes a job to a processor, the processor processes the job until completion or until you abort the job with the DELETE command.

The queue manager uses a disk-resident queue file as a directory for temporary print-formatted files and as storage for job entries and control information. The queue file and temporary files reside on the SP0: logical device in the User File Directory, [1,7]. Logical devices are explained in Chapter 8 in this guide.

The control information in the queue file includes the queue names, processor names, and the status (such as unassigned and stopped) and job entry categories. The queue manager sorts jobs and lists them after each category header. The five job category headers are:

1. ACTIVE JOBS
   Active jobs are currently being processed by a processor.
2. WAITING JOBS
   Waiting jobs are passed to a processor when it becomes idle. The queue manager then deletes the job entry from a WAITING JOBS list and passes the job entry to the ACTIVE JOBS list.
3. HELD JOBS
   The queue manager places jobs into the HELD JOBS list when you specify the SET QUEUE command or when the queue manager aborts an active job.
4. TIME-BLOCKED JOBS
   The queue manager places jobs into the TIME-BLOCKED JOBS list when you specify the /AFTER qualifier with the PRINT, SUBMIT, or SET QUEUE commands. The queue manager moves the job to the WAITING JOBS list at the time you indicated.
5. JOBS UNDER CREATION
   The queue manager places jobs into the JOBS UNDER CREATION list when directed to do so by a task (such as a compiler or batch processor).

The queue manager removes a category header from the queue file when the queue manager removes the last job in a category.

4.1.1 Managing Queues

The system activates print and batch queues when you start up or boot the system. A queue is a priority-ordered list of jobs. You define the priority when you enter the job into the queue. Refer to Section 4.1.3 to start a job.

Through DCL commands you can:

1. Stop and start all the operations of a queue

   Use the STOP/QUEUE and START/QUEUE commands to control all the waiting jobs in a queue.
2. Add a queue to the batch and spool facilities

   Using the INITIALIZE command, you can name a batch or print queue to the batch and spool facilities. Refer to Section 4.6 to initialize a queue.

   Once you initialize a queue, you can insert jobs into the queue. However, the jobs cannot be processed until you assign one or more processors to the queue.
3. Delete a queue from the batch and spool facilities

   Using the DELETE command, you can remove a batch or print queue from the batch and print facilities. Refer to Section 4.9 to delete a queue.

4.1.2 Managing Processors

When you startup or boot the system, the system activates a batch processor to process batch jobs and a print processor to process print jobs. The system restricts you to one print processor for each printer. That is, a printer and print processor can only be assigned on a one-to-one basis. However, since the system permits up to 16 batch processors, you can have multi-stream batch processing by adding the desired number of batch processors.

Through DCL commands, you can:

1. Stop and start a processor

   After you stop a processor with the STOP command, the queue manager does not pass any jobs to the processor until you start the processor with the START command. Refer to Sections 4.3 and 4.4 to stop and start a processor.
2. Add a processor to the batch and spool facilities

   Using the INITIALIZE command, you can name a batch or print processor to the batch and spool facilities. Refer to Section 4.6 to initialize a processor.

   Once you initialize a processor, you must assign it to a queue before the queue manager can send a job to the processor.
3. Delete a processor from the batch and spool facilities

   Using the DELETE command, you can remove a batch or print processor from the batch and spool facilities. Refer to Section 4.9 to delete a processor.

4.1.3 Managing Batch and Print Jobs

Each batch job contains one or more file entries. The batch control information is embedded in the batch files. A batch file can contain one or more of a series of batch commands. The first command of a series is a $JOB command, and the last command of a series is an $EOF command (End Of Job).

Each print job contains one or more print file entries and several job attributes. Each print file entry contains one file specification. A print processor spools each file to its printer in the order the files are listed in the queue file which is the order they were specified in a PRINT command. Besides the file specification, the print file entry also includes the number of copies to print and the information needed to open, read, and delete or keep file(s).

Job attributes common to batch jobs and print jobs are:

1. User identification code

   The UIC of the originating task permits you to distinguish between multiple jobs with the same job name in the same queue.
2. A 1-to-6 character job name

   By default, the queue manager uses the first 6 characters of the first filename as the job name. Moreover, you can specify a job name with a /JOB qualifier when you submit a job.
3. Job entry number

   The queue manager assigns each job a unique job entry number which is a pair of numbers enclosed in parentheses and separated by a comma.
4. Terminal number

   The queue manager assigns your terminal number to the job entry.
5. Job priority

   The job priority in the range of 1 to 250(10) determines the position in the waiting job list, 1 being the lowest priority. Jobs with priority 1 are located at the bottom of the list; all higher priorities are processed first. Jobs with the same priority are processed on a first-in first-out basis.
6. Restart/no restart status

   The restart/no restart status indicates the action the system takes when you restart a job with the START command or when a partially processed job is deferred to be restarted at a later time.

To create a job, use the PRINT and SUBMIT commands. For information on submitting a job to a queue, refer to the *Support Environment User's Guide.* You can stop, abort, and delete a job with the SET QUEUE, DELETE, and STOP commands.

1. Stop a job

   You can momentarily stop a job with the command:

   STOP/PROCESSOR processor-name/PAUSE

   Using the /PAUSE qualifier to the STOP command permits you to continue the job when you restart the processor. Refer to Section 4.3 to stop a processor.

2. Abort a job

   You can abort a job with the commands:

   SET QUEUE queue-name JOB/:job-name HOLD
   STOP/PROCESSOR processor-name/ABORT
   STOP/PROCESSOR processor-name/FILE_END

   Using the SET QUEUE command the /ABORT qualifier to the STOP command permits you to abort the job immediately and place the job into the HELD JOBS list. Using the /FILE_END qualifier permits you to abort the job and place it into the HELD JOBS list after processing the current file entry. Refer to Section 4.3 to stop a processor and to Section 4.5 to use the SET QUEUE command.

   Jobs put into a hold state in this fashion can be restarted at a later time by using the SET QUEUE command to put them back in the WAITING JOBS LIST.

3. Delete a job

   You can delete a job with the command:

   DELETE/QUEUE queue-name/JOB:job-name

   Using the /JOB qualifier to the DELETE command permits you to immediately abort the job and remove it from the queue file. Refer to Section 4.9 to delete a job.

4.2 STOPPING AND STARTING THE QUEUE MANAGER: The STOP/QUEUE/MANAGER Command The START/QUEUE/MANAGER Command

You are provided with privileged STOP and START commands to control the queue manager. After you specify the STOP/QUEUE/MANAGER command, the queue manager performs two functions:

1. Recognizes only the STOP/PROCESSOR command.
2. Terminates batch and spool operations after the system completes all the current jobs or you abort all the current jobs.

Use the START/QUEUE/MANAGER command to restart queuing operations.

4.3 STOPPING QUEUES AND PROCESSORS: The STOP/QUEUE Command The STOP/PROCESSOR Command

Use the privileged STOP command to halt a queue or processor. The STOP command does not affect any processor or queue assignments. After you specify the STOP/QUEUE command, the queue manager stops passing jobs to the assigned processor(s). The queue manager then accepts additional jobs and places them into the stopped queue. Jobs currently being processed from the stopped queue are allowed to complete.

When you specify the STOP/PROCESSOR command, the queue manager stops the processor immediately if it is idle. You cannot request the queue manager to pass a job to a stopped processor. You can request the queue manager to stop a busy processor at various locations in a job. The processor can be:

1. Stopped after either the current print line or batch job statement
2. Stopped at the end of the current print file entry
3. Stopped at the end of the current batch command file
4. Stopped at the end of the current job
5. Stopped after aborting the current job.

4.4 STARTING QUEUES AND PROCESSORS: The START/QUEUE Command The START/PROCESSOR Command

Use the privileged START command to restart a queue or processor. The START command does not affect processor and queue assignments. After you specify the START/QUEUE command, the queue manager disburses jobs from the queue. After you specify the START/PROCESSOR command, the queue manager passes to the processor either the first waiting job or the current job that was running at the time the processor was stopped. The current job can be:

1. Resumed after the current batch job statement or printed line
2. Restarted at the end of the current batch job series
3. Restarted at the beginning of the job
4. Restarted at the beginning of the current print file
5. Restarted after spacing forward or backward a specified number of pages in the current print file entry
6. Restarted at a specified page in a print file
7. Restarted with the next job in any assigned queue after aborting the current job
8. Restarted with a new form attribute. (This also implies the previous statement.)

4.5 DISPLAYING AND CHANGING JOB ATTRIBUTES: The SHOW QUEUE Command The SET QUEUE Command

Use the SHOW QUEUE command to display the queue file contents. Through options to the SHOW QUEUE command, you can request the system to display either the full queue file contents or selected portions of the queue file. You can request a display of a specific queue, queue file entry, or all entries of a specific job.

Use the SET QUEUE command to modify a batch or print job. You are permitted to modify any attribute of any job. In addition, you can move any job from one job list to another job list. For instance, you may wish to move a large print job from the WAITING JOBS list to the TIME-BLOCKED JOBS list. Specify the SET QUEUE command with the /AFTER option to inform the queue manager to delay activating the job until an off-peak time. The queue manager moves the job to the TIME-BLOCKED JOBS list.

4.6 INITIALIZING A QUEUE OR PROCESSOR: The INITIALIZE Command

Use the INITIALIZE command to assign batch and print queues to the system. The queue manager names the queue and enters it into the queue file. For example, you may wish to have one queue for daytime work and another for nighttime work or to divide the work load by types of work. In either case, the work must be submitted to the appropriate queue. Through the queue manager, you can start and stop the queues at the appropriate time. Thus you select the work to be performed in a given time period.

You can also add batch and print processors to a queue with the INITIALIZE command. However, for the queue manager to pass jobs to the processor, you must also assign the processor to the queue with the ASSIGN command. For example, you may wish to add a second batch processor to a queue. Suppose you are running a batch job which requires 30 minutes to complete and you wish to run shorter jobs too. You can add another batch processor to the queue to process the additional jobs immediately.

4.7 ASSIGNING A PROCESSOR TO A QUEUE: The ASSIGN/QUEUE Command

Use the privileged ASSIGN/QUEUE command to direct the queue manager to make an association between a queue and processor. This allows eligible jobs to be routed from the queue to the processor. You must assign print processors to print queues and batch processors to batch queues; however, you can assign a processor to one or more queues of the same type. You may also assign multiple processors to one queue. Full flexibility of assignments between queues and processors of a given type is allowed.

4.8 DEASSIGNING PROCESSORS: The DEASSIGN/QUEUE Command

Use the privileged DEASSIGN/QUEUE command to dissociate a processor from a queue. The queue remains listed in the queue file. The queue manager labels a queue, PROCESSOR ASSIGNED NONE, when you deassign the final processor from a queue.

4.9 DELETING QUEUES, PROCESSORS, AND JOBS: The DELETE/QUEUE Command The DELETE/PROCESSOR Command

Use the privileged DELETE command to remove a queue, processor, or job from the queue file when you no longer need it. The status lists associated with the queue or processor must be emptied, or the jobs aborted before the system deletes the processor or queue.

If the lists contain one or more jobs, the queue manager labels the queue or processor for deletion. Furthermore, you cannot submit a job to a processor or queue after the processor or queue is either deleted or labeled for deletion. When the queue manager finally deletes a queue, the queue manager automatically deassigns any assigned processors.

The system deletes an idle processor immediately. The queue manager labels a busy processor, MARKED FOR DELETE, and deletes the processor when either the processor completes the current job or you abort the current job.

CHAPTER 5

MAINTAINING TRANSACTION PROCESSORS AND DATA FILES

This chapter presents the utilities to maintain transaction processors and the journal. There are utilities to gather and report transaction processor statistics, to modify an installed transaction processor, to start and stop writing activities to the journal, and to read the journal. The utilities are:

1. TPMOD – The Transaction Processing Modification Utility

   Use the TPMOD utility to modify transaction processors (for example, redirect stations and abort a transaction instance).
2. SYSMOD – The System Modification Utility

   Use the SYSMOD utility to assign/deassign a journal device to the system, which it uses to record transaction processor activities. To start and stop the journal activities, refer to the TPCTRL utility description in Chapter 3 in this guide.
3. TPSTAT – The Transaction Processing Statistics Gathering Utility

   Use the TPSTAT utility to record a sampling of transaction processor activities. For instance, you can request a sampling at 10-minute intervals. The TPSTAT utility then records a sample of the system activities during the sampling time. You then use the STAREP utility to display a report of the activities gathered by the TPSTAT utility.
4. STAREP – The Transaction Processor Statistics Report Generator Utility

   Use the STAREP utility to generate print-formatted reports of statistics compiled by the TPSTAT utility.
5. RECOVR – The Recovery Utility

   Use the RECOVR utility together with a previous archived copy of your data files to update application data files if they are corrupted by a crash.
6. SHOLOG – The Show Log Utility

   Use the SHOLOG utility to generate a report based on logged messages in the journal.

5.1 MODIFYING A TRANSACTION PROCESSOR (TPMOD)

The transaction processing Modification Utility (TPMOD) displays and modifies memory-resident data used by an installed transaction processor. TPMOD can be run in a minimum support environment to redirect stations,

connect or disconnect terminal stations, broadcast a message, abort a transaction instance and modify or display selective elements in the transaction processor data structure. TPMOD can also be used to alter a transaction processor without undergoing a full install.

The TPMOD changes can be made on either a temporary or permanent basis. The system retains temporary changes until you remove the transaction processor. The system applies permanent changes to the copy of the data structures saved on disk. Thus, the changes remain in effect for subsequent starts and reinstallations if you use the BRIEF option. If you reinstall the transaction processor without the BRIEF option, the transaction processor reverts to its original definition.

There is also the option of maintaining a record of the changes. That is, you can create a log file in which a copy of the utility dialog is recorded for future reference.

The TPMOD utility serves many purposes. For instance, the possibility exists that the demands of the transaction processor might change, as in the increased usage of a given transaction for a particular time of day. At that time, a TST associated with the transaction needs to have its characteristics changed, such as the number of active copies. You can accomplish this change by creating a batch stream that would run the TPMOD utility at a specified time. This batch stream can change the maximum number of active copies of the TST. Later, you run another batch stream to undo these changes, without halting the current processing.

Another example is any situation interfering with the normal processing of the transaction processor. You can use TPMOD to take some remedial action. For instance, if an output-only station went down, the system manager could redirect all messages to another station.

This utility is also useful during the transaction processor development phase. For instance, TPMOD provides a fast and easy way to observe the effects when a designer changes system parameters.

You invoke TPMOD from a privileged terminal by specifying:

```
RUN $TPMOD
```

The TPMOD dialog proceeds as follows:

1. Transaction processor name?

   Specify the transaction processor to be modified or displayed by TPMOD. This transaction processor must be currently installed.

2. Log file <TPMOD.LOG>?

   Respond with the file specification for the log file. The default extension is LOG. If this file already exists, TPMOD appends the current entries to the file. If a log is not to be kept, respond with NONE.

3. Permanent changes <NO>?

   If you respond with YES, TPMOD applies all changes to the copy of the data structures saved on the disk. This process keeps the change in effect as long as you install the transaction processor with the BRIEF command of the TPCTRL utility.

4. Command?

   Select the next function for this utility to execute. Valid responses are:

   A. REDIRECT
      The REDIRECT command associates a station with another station and causes messages from the former terminal to be redirected to the latter terminal. Continue with Question 5.
   B. ACQUIRE
      The ACQUIRE command gives the transaction processor exclusive access to a terminal station resource. Continue with Question 11.
   C. OPEN
      The OPEN command makes the terminal station available to the transaction processor. Continue with Question 11.
   D. CLOSE
      The CLOSE command closes a file or station. Continue with Question 11.
   E. RELEASE
      The RELEASE command removes the exclusive access of a transaction processor to a terminal station. Continue with Question 11.
   F. BROADCAST
      The BROADCAST command allows you to send up to 80 characters of a message to all the terminals in the system. Continue with Question 13.
   G. SET
      The SET command allows you to modify the transaction processor data base. Continue with Question 7.
   H. SHOW
      The SHOW command provides the facility for displaying system parameters. Continue with Question 8.
   I. EXIT
      The EXIT command closes all the files and terminates the utility.

5. From station?

   Specify the name of the station whose messages are to be redirected.

6. To station?

   Specify the name of the station to which the messages are to be directed. Mailbox stations, link stations, and output-only stations may be redirected. The following redirections are valid:

   A. Master link station to master link station
   B. Mailbox station to mailbox station or mailbox station to output-only device
   C. Output-only station to output-only stations and output-only station to mailbox station

   The TPMOD utility checks for valid redirections. Also if you inadvertently specify a circular redirection, the TPMOD utility prints an error message and repeats the prompt. If you take the default response, the TPMOD utility terminates the any redirection in effect redirecting the station to itself. Return to Question 4.

7. Set command?

   You can respond with any of the following commands:

   A. COPIES
      The COPIES command changes the maximum number of copies of a TST that may be active at one time. Continue with Question 11.
   B. ENABLE
      The ENABLE command enables a transaction or permits the system to establish a link from/to a station. Continue with Question 11.
   C. DISABLE
      The DISABLE command disables a transaction or disconnects a link from a station. Continue with Question 11.
   D. LOCK
      The LOCK command changes the time delay before the second attempt to access a locked record. Continue with Question 11.
   E. PRIORITY
      The PRIORITY command changes the priority of the TST station. Continue with Question 11.
   F. ERROR COUNT
      The ERROR COUNT command sets the value of the global error limit. If a transaction type has more than this number of system-detected errors, the system disables the transaction. Continue with Question 12.
   G. ABORT TRANSACTION
      The ABORT TRANSACTION command requests the source station name and the transaction identification, and then aborts the transaction instance. Continue with Question 11.

8. Listing device <TI:>

   Specify the device to which listings, produced by the TPMOD command, are to be output. You can specify either an output device or a file specification. The default extension for the file specification is .LST. Taking the default directs the listings to your terminal.

9. Data block?

   Specify the type of data to display. Valid responses are:

   A. TERMINAL

      The TERMINAL response lists the following items:

      1. Station name
      2. Is the station redirected? If yes, where?
      3. The allowed operations at the terminal
      4. Is the station initialized?

   B. TST

      The TST response lists the following items:

      1. Station name
      2. Station priority
      3. Maximum number of copies

   C. TRANSACTION

      The TRANSACTION response lists the following items:

      1. Transaction name
      2. Is the transaction disabled?
      3. Error count for this transaction type

   D. FILES

      The FILES response lists the following items:

      1. Logical filename
      2. Filename
      3. Lock wait time

   E. ERROR COUNT

      The ERROR COUNT response lists the value of the system error cut-off for a particular transaction. Return to Question 4.

   F. ACTIVITY

      The ACTIVITY response lists all active transactions for the named station. The list also includes the transaction identification and the transaction state for each of these transaction instances.

   G. LINK

      The LINK response lists the selected master link station:

      1. Station name
      2. Number of sublinks
      3. System error count.

10. Name <ALL>?

    Specify the device name to be used with the previous response. For stations you may specify either the individual name or a group name (ending in two asterisks). For more information, see the STADEF utility in the *Application Programmer's Guide.* The TPMOD utility repeats the prompt until you take the default response. The first time TPMOD displays the question, the default is ALL. For the remainder of the time, the default is DONE. Return to Question 4.

11. Name?

    Specify the logical name of the station or file for which the change you specified in Question 7 is to be made. The station type must be valid for the command.

12. New value?

    If the selected command requires an additional parameter, specify the value here. Return to Question 4.

13. Broadcast Message?

    The user enters up to 80 characters of text as a message. The system broadcasts the message to all the terminals. Return to Question 4.

5.2 MODIFYING SYSTEM PARAMETERS (SYSMOD)

Use the System Modification Utility, SYSMOD, in a minimum support environment to:

1. Assign or change the primary and/or secondary journal devices

   NOTE

   If the device is currently in use, the change does not take effect until the system switches back to it.

2. Display the names of the primary and secondary journal devices and identify which device the system is currently using to record the journal and log data.

You invoke SYSMOD from a privileged terminal by typing:

```
RUN $SYSMOD
```

The SYSMOD dialog proceeds as follows:

1. Command <EXIT>?

   Specify the function to be performed. Valid responses are:

   A. CHANGE
      The CHANGE command assigns a journal device. Control continues with Question 2.
   B. SHOW
      The SHOW command displays the journal devices. Control returns to Question 1.
   C. EXIT
      The EXIT command terminates the SYSMOD utility and causes any applied changes to take effect.

2. Old device?

   Specify the name of the journal device to be changed. Use a response of NONE to indicate that no device was previously defined.

3. New device?

   Specify the name of the device which replaces the journal device you specified in Question 2. If, as a result of your responses there is no journal device, the system issues a warning message. Control then returns to Question 1.

5.3 REPORTING SYSTEM STATISTICS

The Cumulative Statistics Utilities, TPSTAT and STAREP, measure and report the performance of the transaction processing system over a period of time. The transaction processors have a built-in capability, via statistics cells, to keep track of various system parameters. The TPSTAT utility, which can run in a minimum support environment, sample statistics cells at specific time intervals and saves the parameters in a file. The STAREP utility, which must run in a full support environment, can then analyze the sample data and produce a report of the analysis. These cells are only initialized when the transaction processor is started. Sampling these cells via TPSTAT does not initialize the statistics cells.

The ability to monitor system performance is very useful during the various phases of the life cycle of a transaction processor. During development it can be used to determine the performance characteristics of the system. Similarly, it can be used to ensure that the performance does not decline during day-to-day operations. Finally, it can be used as an indicator of future system needs.

The reports generated by STAREP are in an easy-to-read format. However, a full interpretation of any report requires some knowledge of the system. The transaction processor gathers the sample data as quickly as possible, with minimal effect on the system performance.

The report generation utility runs independently of the sample gathering. This separation occurs in order that the former can be run during the off hours or as a low-priority background job.

5.3.1 Gathering Statistics (TPSTAT)

The privileged TPSTAT utility, a high priority task, runs at specified intervals and selectively reads and records in the output file the transaction processor statistics cells. To invoke the TPSTAT utility, use the command:

```
RUN $TPSTAT
```

The utility displays the following dialog:

1. Transaction Processor Name?

   Specify the name of the transaction processor which is to be sampled. The transaction processor must be in the run state.

2. Interval Between Samples?

   Specify the time interval between statistics cells samplings. The unit is in minutes.
3. Number of Samples?

   Specify the total number of samples to be taken.
4. Output File <tpname.SMP>?

   Specify the name of the file in which the collected samples are to be stored. The default filename is tpname.SMP where tpname is the name of the transaction processor being sampled.

5.3.2 Reporting Statistics (STAREP)

The data gathered by the TPSTAT utility can be analyzed and displayed in a meaningful format by the Statistics Report Utility, (STAREP). To invoke the STAREP utility, use the command:

```
RUN $STAREP
```

The utility displays the following dialog:

1. Input File?

   Specify the name of the file which contains the accumulated sample data. This name is normally the same as that you typed in as the response to the output file question in TPSTAT.
2. Listing Device <LP:>?

   Specify the filename or device where the utility is to write the report. If you respond with a file specification, the utility assigns .LST as the default filetype. A carriage return response causes the utility to spool the report on the line printer.
3. Sample Report <YES>?

   Respond with a YES (or CR) if the utility is to include the data from each sample in the report.
4. Summary Report <YES>?

   Respond with a YES (or CR) if the utility is to include the summary pages in the report, giving extreme values, mean values, and totals of the sampled quantities.

NOTE

Both the sample and summary reports may be requested at the same time.

5.4 RECOVERING APPLICATION DATA FILES (RECOVR)

The RECOVR utility, which must run in a full support environment, uses a journal set to update application data files which were corrupted by a catastrophic system malfunction. A journal set is all the journal volumes used by the system since the last system archive. The RECOVR utility provides for selective recovery based on the transaction processor, one or more data files within the transaction processor, and the date and the time.

This utility requires that the definition of the application data files defined for each transaction processor remain static over the life of a journal set. If an application programmer uses the FILDEF utility to redefine the application data files, you must archive the volume containing the application data files and start a new journal set.

To invoke the RECOVR utility, use the command:

```
RUN $RECOVR
```

The utility displays the following dialog:

1. Transaction processor name <ALL>?

   Select the transaction processors from the list cataloged in the file [1,1] TPDEF.TPF. This question is repeated until you respond with DONE. You can specify a maximum of 16 transaction processors.

   ALL is the default for the first occurrence of the prompt. It causes the selection of all transaction processors in the TPDEF.TPF file.

   DONE is the default on all later occurrences of this prompt which causes the utility to proceed to the next question.

2. Ending date <last transaction>?

   Specify the date for the ending point when scanning the journal file for records to use as updates to the application data file set(s) under recovery. That is, recovery stops at the time specified in Question 3 on the date specified to this question. This specification is binding over all transaction processors specified for recovery in Question 1. Valid responses are:

   A. A date

      Respond with a date in the dd-mmm-yy format. Go to Question 3.

   B. <CR>

      A carriage return-only invokes the default which is the date of the last record in the file. Go to Question 4.

3. Ending time <23:59>?

   This question is asked as a modifier to Question 2. Its purpose is to fine-tune the ending point for recovery. Valid responses are:

   A. Time in hh:mm format

   B. <CR>

      A carriage return-only invokes the default which is one minute before midnight on the specified date.

4. Please make selections for transaction processor <tpname>. Logical filename <ALL>?

   The statement and question are repeated for each transaction processor you specified in Question 1.

   Select a logical filename associated with the transaction processor. Logical filenames are associated with transaction processors through use of the FILDEF utility.

   ALL is the default for the first occurrence of the prompt, and DONE is the default for all later occurrences of the prompt. The utility repeats the prompt until you accept the default; at which time the utility selects the next transaction processor. When the last transaction processor is completed, go to Question 5.

5. Journal File <DONE>?

   Load the journal volume and enter the device name containing the journal volume.

   If you respond with DONE, continue with Question 7.

6. Volume started at hh:mm on dd-mmm-yy. Continue <YES>?

   This question permits validation that you mounted the correct volume. A response of NO causes the utility to return to Question 5 where you are expected to replace the volume with the correct volume.

   If you respond with YES, the utility recovers the data files, displays the following message, and returns to Question 5.

   > Last Transaction on volume was initiated at hh:mm on dd-mmm-yy

   If the date and time that the utility displays represent the final transaction, take the default response to Question 5. If the date and time that the utility displays does not represent the final transaction, load another journal volume and specify the device name in response to Question 5.

7. Report Listing file <CL:>

   This question allows you to specify the device that RECOVR will use to output the summary report.

   NOTE

   The summary report, for each transaction entered on the journal tape, provides such information as the transaction name, the ID, the source station and time executed.

8. Date on which to start summary report?
9. Time at which to start summary report?

   Questions 8 and 9 solicit information needed by RECOVR in order to commence generation of the summary report. If the date and time specified do not appear on the last journal volume, the next question is asked.

10. Previous Volume?

    Load the journal volume containing the correct starting time and date for the report. Enter the device name where you loaded the journal volume. After reading the specified volume, this prompt is repeated if you did not load the correct volume.

11. Next Volume?

    After the report commences, the utility repeats this prompt at the end of each volume until the utility reads the final transaction recovered.

5.5 REPORTING JOURNAL-LOGGED MESSAGES (SHOLOG)

The SHOLOG utility, which must run in a full support environment, provides a report based on logged messages for a particular transaction processor. Selection of messages to be included in the report is made on a transaction, station, or log identifier basis. A date/time range is also available as a selection criterion.

To invoke the SHOLOG utility, use the command:

```
RUN $SHOLOG
```

The system responds with the following dialog:

1. Transaction processor name <ALL>?

   Select the name of a transaction processor cataloged in the [1,1] TPDEF.TPF file.

   The utility continues with Question 3 if your response matches an established transaction processor. If they do not match, continue with Question 2.

   If you select all the transaction processors, the system implicitly selects all transactions and all source stations. Therefore, Questions 3 to 6 are not asked. The dialog continues with Question 7.

2. % <tpname> is not an established transaction processor. Should this name be accepted anyway <NO>?

   Go to Question 3 if you respond with YES. Return to Question 1 if you respond with NO.

   You might typically answer YES if the transaction processor was deleted from the system after the errors to be reported were logged. You might typically answer NO if you incorrectly typed a transaction processor name.

3. Transaction name <ALL>?

   You can respond with either a transaction name or ALL. If you respond with ALL, the utility selects all of the log records without regard to the transaction name. If you respond with a transaction name, the system selects the log records belonging to the transaction. After the utility acknowledges the selection for the

transaction, the utility repeats Question 3. You can respond with either another transaction name or with DONE. If you respond with a transaction name, the utility repeats the action previously described. If you respond with DONE, the utility continues with Question 5.

The system validates each transaction name against the tpname.TRA file. If the utility finds a match, the utility accepts the transaction name and repeats the prompt. If there is no match, the utility goes to Question 4 and then returns to Question 3.

4. Should this transaction name be accepted <NO>?

   Respond with NO if you entered an incorrect transaction name. The utility rejects the transaction name and returns to Question 3.

   Respond with YES if the transaction name is correct but it was deleted from the validation file. The utility accepts the transaction name for selecting the log records as explained in Question 3.

5. Source Stations <ALL>?

   You can respond with either a source station name or ALL. If you respond with ALL, the utility selects all of the log records without regard to station names. If you respond with a source station name, the utility selects the log records originating from that station. After the utility acknowledges the selection for the source station, the utility repeats Question 5. You can respond with either another station name or with DONE. If you respond with a station name, the utility repeats the action previously described. If you respond with DONE, the utility continues with Question 7.

   The system validates each source station name against the source stations established in the <tpname>.STA file. If the utility finds a match, the utility accepts the source station name and repeats the prompt. If there is no match, the utility goes to Question 6 and then returns to Question 5.

6. Should this station name be accepted <NO>?

   Respond with NO if you responded with an incorrect station name. The utility rejects the station name and returns to Question 5.

   If the station name is correct, but was deleted from the validation file, respond with YES and the station name is accepted. Return to Question 5, allowing for additional station specifications.

7. Starting date <FIRST MESSAGE>?

   Specify the date the utility is to use as a starting point for the selection of log records. Valid responses are:

A. A date

Respond with a date in the dd-mmm-yy format. Go to Question 8.

B. <CR>

Respond with a carriage return-only to indicate the date of the first record in the file. Go to Question 9.

8. Starting time <00:00>?

The utility displays this question to indicate the starting time for the selection of log records. Valid responses are:

A. Time

Respond with a time in the hh:mm format.

B. <CR>

Respond with a carriage return-only to indicate midnight on the specified date. Go to Question 9.

9. Ending date <LAST MESSAGE>?

Specify the date the utility is to use as an ending point for the selection of log records. Valid responses are:

A. A date

Respond with a date in the dd-mmm-yy format.

B. <CR>

Respond with a carriage return-only to indicate the date of the last record in the file. Go to Question 11.

10. Ending time <23:59>?

The utility displays this question to indicate the ending time for the selection of log records. Valid responses are:

A. Time

Respond with a time in the hh:mm format.

B. <CR>

Respond with a carriage return-only to indicate one second before midnight on the specified date.

11. Log type <ALL>?

You can respond with either a log type or ALL. If you respond with ALL, the utility selects all of the log records without regard to the log type. If you respond with a log type, the system selects the log records belonging to the transaction. After the utility acknowledges the selection for the transaction, the utility repeats Question 11. You can respond with either another log type or with DONE. If you respond with a log type, the utility continues to repeat the question until you respond with DONE, at which time the utility goes to Question 12.

12. Print File Specification <LP:>?

You can respond with:

A. LP:

The LP: response causes the utility to generate and spool a print-formatted file to the line printer.

B. File specification

A file specification causes the utility to generate a print-formatted file, but not to spool the file to the line printer.

C. NONE

The NONE response inhibits the utility from generating the print-formatted file.

13. Output data file specification <NONE>?

Respond with a file specification where the utility is to write a nonprint-formatted file of the selected log records. The utility formats the file as explained in Table 5-1. DIGITAL makes this file available to you to use with a program you write for accessing this output file.

Table 5-1 SHOLOG Output Data File Format

Field Description	Type	Length (bytes)
Transaction Processor	ASCII	6
Transaction Name	ASCII	6
Transaction Instance	Binary	4
Log Type	ASCII	1
Source Station Name	ASCII	6
Logging Station Name	ASCII	6
Year	Binary	2
Month in the Year	Binary	2
Day in the Month	Binary	2
Hour in the Day	Binary	2
Minute in the Hour	Binary	2
Second in the Minute	Binary	2
Length of User Log Data	Binary	2
User-Defined Log Data	ASCII	Variable

14. Journal file <DONE>?

Enter a device name which defines to the utility the journal file. While processing is taking place, the utility repeats this prompt when the utility reaches the end of the volume. If you respond with DONE, the system terminates the utility.

15. Volume started at hh:mm on dd-mmm-yy. Continue <YES>?

This question validates that the proper volume is mounted. If you respond with NO, return to Question 14.

CHAPTER 6

TRANSACTION PROCESSOR SOFTWARE ERROR-LOGGING/REPORTING

The system performs software error-logging with the SERLOG utility. SERLOG receives messages containing information about errors detected in software modules and writes the associated data out to the error-logging file, SERLOG.LOG. The system logs errors for only those transaction processors you select.

The error analysis utilities, SERANL and SERDAY, provide a wide variety of individual detail and/or summary reports from the data in the error log file.

The SERCTL utility provides several control functions for the software error-logging operation.

6.1 LOGGING ERRORS (SERLOG)

The SERLOG utility, which can run in a minimum support environment, logs software errors. If an error is detected, the system checks a control file to determine if the transaction processor name is listed for logging errors or if the errors are to be ignored. If the error is to be logged, the system logs the error with the current date and time. SERLOG writes the error message to the [1,300] SERLOG.LOG file and has the option to format the error and to display it on a support environment terminal.

The system runs SERLOG with the INIT utility when you start up the system. You can stop the SERLOG utility with the AUXSTP utility and you can restart the SERLOG utility with the command:

```
RUN/TASK:SERLOG $SERLOG
```

6.2 CONTROLLING THE ERROR LOGGER (SERCTL)

The SERCTL utility, which must run in a full support environment, controls the error-logging process. You may enable or disable error-logging on a transaction processor basis, have data deleted from the error log file or cause the error-logging module to exit.

The utility may be invoked by the following command:

```
RUN $SERCTL
```

The SERCTL dialog proceeds as follows:

1. Command <EXIT>?

   The SERCTL commands are:

   A. SELECT
      The SELECT command selects or deselects error-logging for a specific transaction processor. Continue with Question 2.
   B. REMOVE
      The REMOVE command removes all or part of the current error log file. The SERLOG utility must be running. Continue with Question 2.
   C. STOP
      The STOP command stops the SERLOG utility. The SERCTL utility repeats Question 1.
   D. TERMINAL
      The TERMINAL command changes the secondary logging device, and the dialog continues at Question 6. The secondary logging device displays a 3-line message for each error message. The display includes:
      1. Date and time
      2. Transaction name, identification, and source station
      3. Error
   E. EXIT
      The EXIT command exits the SERCTL utility.

2. Transaction processor name <ALL>?

   You can respond with a transaction processor name, DONE, and ALL. If you respond with a transaction processor name, the utility applies the command you specified in Question 1 to the transaction processor.

   If you respond with DONE, return to Question 1.

   If you specified the SELECT command in Question 1 and respond with ALL or a transaction processor name, continue with Question 3.

   If you specified the REMOVE command in Question 1 and respond with ALL on a transaction processor name, continue with Question 4.

3. Select error monitoring <YES>?

   If you wish to log errors for the named transaction processor, answer YES. If you respond with NO, the SERLOG utility ignores the errors for that transaction processor. Return to Question 2 or if you specified ALL return to Question 1.

4. Ending date <LAST ERROR>?

   Respond with a date in the format:

   dd is the day of the month
   mmm is the abbreviation of the month
   yy is the last 2 digits of the year.

   If you respond with a carriage return-only, the utility removes from the file all errors for the named transaction processor that occurred on or before this date.
5. Ending time <23:59>?

   The system requests a time of day. SERCTL purges all errors for the named transaction processor which occurred on the ending date and at or before the ending time. The SERCTL utility sends a message to the error-logging module. Return to Question 1.
6. Device name <default>?

   Specify the name of the secondary error-logging device. Valid responses are a terminal device or NONE. If you specify NONE, there is no secondary reporting of errors. The default value is the current value. The initial default at the time of system startup is the console-listing device. The new secondary error-logging device is placed into effect the next time the error logger is run.

6.3 REPORTING ERRORS (SERANL)

The SERANL utility, which must run in a full support environment, prepares three error reports.

1. A detailed error listing containing the errors occurring in a given time window for a given transaction processor and transaction. This, contains enough data to identify the error, but does not have all the information that was present on the error log file. This should be the most commonly used report.
2. A formatted error dump of all the data on the error log file in a given time window and for a given transaction processor and transaction. The information in this report is for the software specialist.
3. A summary total report containing totals for each error type. These totals are also accumulated from all errors occurring in a given time window for a given transaction processor and transaction.

The utility may be invoked by the following command:

```
RUN $SERANL
```

The SERANL dialog proceeds as follows:

1. Input file <[1,300]SERLOG.LOG>?

   Enter the file specification for the error log file to be examined. The default specification is the latest version of the software error log file1 [1,300]SERLOG.LOG.

2. Report <LIST>?

   The allowed responses are:

   A. DUMP

      The DUMP response provides a detailed dump (formatted) of the error log file for the specified window, transaction processor, and transaction.

      Since the data may be in any format, a dump report shows the ASCII and the octal representation of each word. The ASCII representation presents the two component bytes in the word. Beneath this, the octal representation of the word is printed in a columnar manner so that the alignment is preserved. The dump is printed in sequential groups of 64 bytes or 32 words.

   B. LIST

      The LIST response provides a detailed report of the error log file for the specified window, transaction processor, and transaction. The report contains all the information needed to identify the error and includes the summary report.

   C. SUMMARY

      The SUMMARY response provides a summary report. The report contains the totals for the entire transaction processor for the specified window and totals for the particular transaction processor and transaction.

3. Output file <default>?

   Enter the device or file specification indicating where SERANL outputs the report file.

   If you select either a full-detailed list or a dump from the SERANL utility, it selects the line printer (LP:) as the default output device. If you select the summary total report, the SERANL utility selects your terminal (TI:) as the default output device.

4. Transaction processor name <ALL>?

   Respond with 1- to 6- alphanumeric characters identifying a specific transaction processor name. The SERANL utility uses only the error entries belonging to this transaction processor to prepare the report.

   The SERANL utility validates the response and if you:

A. Respond with a recognized transaction processor, continue with Question 6.
B. Respond with an unrecognized transaction processor SERANL prints an error message and goes to Question 5.
C. Respond with ALL, the SERANL utility includes error entries from every transaction processor in the report. Continue with Question 8.

5. Should this transaction processor name be accepted <NO>?

   If you respond with YES, SERANL accepts the response in Question 4 for selecting error log entries. Continue with Question 6.

   You might typically answer YES if that transaction processor was deleted from the system after the errors to be reported were logged. You might typically answer NO if you incorrectly typed a transaction processor name. In this case, the SERANL utility rejects the response to Question 4 and returns you to Question 4 to allow you to respond with another transaction processor name.

6. Transaction name <ALL>?

   Respond with 1- to 6- alphanumeric characters identifying a specific transaction name. The SERANL utility selects the error entries belonging to this transaction in the specified transaction processor for inclusion in the report.

   A. Respond with a recognized transaction, SERANL accepts the name and repeats the prompt.
   B. Respond with an unrecognized transaction, SERANL prints an error message and goes to Question 7 and then returns to Question 6.
   C. Respond with ALL, SERANL includes error records from any transaction within the specified transaction processor on the list and in the summary totals. Continue with Question 8.
   D. Respond with DONE, SERANL ends the transaction name input loop and go to Question 8.

7. Should the transaction name be accepted <NO>?

   If you respond with YES, the SERANL utility accepts your response to Question 6. If you respond with NO, SERANL rejects the name you specified in Question 6. In either case, the utility returns to Question 6.

8. Starting date <FIRST ERROR>?

   The SERANL utility excludes errors from the report that were logged before this date. Respond using the dd-mmm-yy format. If you select the default, SERANL ignores the starting date and time as part of the selection criteria.

9. Starting time <00:00>?

   Specify a starting time to be used in conjunction with the starting date established in Question 8.

10. Ending date <LAST ERROR>?

    The SERANL utility excludes error entries logged after the date. Respond using the format, dd-mmm-yy.

    If you respond with a carriage return-only, the SERANL utility ignores the ending date and time in the selection criteria. Also the dialog ends and SERANL prepares the report.

11. Ending time <23:59>?

    Specify an ending time for SERANL to use in conjunction with the ending date. The dialog ends and SERANL prepares the report.

6.4 REPORTING CURRENT DAY ERRORS (SERDAY)

The SERDAY utility, which runs in minimum support environment, produces an error report (no dump or totals). The error report is identical to that produced by the SERANL utility if you were to answer the following SERANL utility questions as indicated here.

1. Input File? SERLOG.LOG
2. Report? LIST
3. Output File? SERANL.LST
4. Transaction Processor? ALL
5. Starting Date? <today's date>
6. Starting Time? 00:00
7. Ending Date? Last error

You invoke the SERDAY utility with the following command:

```
RUN $SERDAY
```

The SERDAY utility is useful in a transaction-processing with a minimum support environment where the limited facilities prohibit using the SERANL utility.

6.5 MODIFYING A SPECIAL FORMS DEFINITION RECORD (FDFUPT)

Use the privileged FDFUPT utility, in a full support environment, to modify a special forms definition record which is included in the forms definition file of every transaction processor at the time the transaction processor is created. The record contains the ABORT form and the SHUTDOWN message. The ABORT form appears on an application terminal screen when a transaction aborts. The SHUTDOWN message appears on line 24 of the application terminal screen when you invoke the SHUTDOWN utility.

You can use the FDFUPT utility to:

1. Edit the appearance of the ABORT form or SHUTDOWN message.
2. Build a new form definition record.

   You can build a new record and include the new record in a transaction processor forms definition file.

You invoke the FDFUPT utility with the command:

```
RUN $FDFUPT
```

The FDFUPT utility displays the question:

```
Command <EDIT>?
```

The FDFUP commands are:

1. EDIT
   Use the EDIT command to modify the ABORT form or the SHUTDOWN message. If you select this command, go to Section 6.5.1 for further details.
2. UPDATE
   Use the UPDATE command to build a new forms definition record and add it to a forms definition file. If you select this command, go to Section 6.5.2 for further details.
3. EXIT
   The EXIT command terminates the FDFUPT utility.

6.5.1 Editing with FDFUPT

If you select the EDIT command, the FDFUPT utility displays the question:

```
ABORT/REPLY/DONE <DONE>?
```

Respond with one of the following options:

1. ABORT
   Use the ABORT option to edit the ABORT form. If you select this command, go to Section 6.5.1.1 for further details.
2. REPLY
   Use the REPLY option to edit the SHUTDOWN message. If you select this command, go to Section 6.5.1.2 for further details.
3. DONE
   Use the DONE option to terminate the EDIT command of the FDFUPT utility and to return to the command question?

   ```
   Command <EDIT>?
   ```

6.5.1.1 Editing the ABORT Form – If you respond with ABORT, the FDFUPT utility displays the existing forms definition and proceeds with the following dialog:

1. ROW <currently defined row or 1>
   Enter the row number (in a range from 1 to 23) where the message is to appear.
2. VALUE?
   You must define the message text that the system displays when an abort occurs as a series of VALUE arguments, separated by commas, which you enter using the following ATL VALUE constructs:

   NOTE

   If you respond with a carriage return-only, the FDFUPT utility assigns a null value which results in a blank ABORT form.

   A. "string"
      You can specify a string of characters enclosed by quotation marks. When the system displays the ABORT form on the terminal, the system displays the quoted string without the quotation marks.
   B. DATE
      The DATE argument causes the system to display the ABORT form with the date in the dd-mmm-yy format.
   C. TIME
      The TIME argument causes the system to display the ABORT form with the clock time in the hh:mm:ss format.
   D. TRANSACTION
      The TRANSACTION argument causes the system to display the ABORT form with the system-determined transaction instance number as 10 characters.
   E. NAME
      The NAME argument causes the system to display the ABORT form with the 6-character name of the transaction that is currently being run from the application terminal.
   F. STATION
      The STATION argument causes the system to display the ABORT form with the 6-character terminal station identification on the terminal.

   If the response to Question 2 requires two or more lines, you can continue the response by ending the line to be continued with a hyphen (-). Refer to Section 6.5.1.3 for more information.

For example, you can change the ABORT form with the following value response.

VALUE?"Transaction",NAME,"",TRANSACTION,"aborted at "DATE,"",TIME

In the example, the word Transaction and the phrase aborted at are quoted strings. In addition, the spaces between the system-supplied values such as DATE and TIME are quoted spaces.

6.5.1.2 Editing the SHUTDOWN Message – If you respond with REPLY, the FDFUPT utility displays the existing forms definition and proceeds with the question:

VALUE?

You must define the message text that the system displays when you shut down the system as a series of VALUE arguments, separated by commas, which you enter using the following ATL VALUE constructs:

NOTE

If you respond with a carriage return-only, the FDFUPT utility assigns a null value which results in a blank SHUTDOWN message.

1. "string"
   You can specify a string of characters enclosed by quotation marks. When the system displays the SHUTDOWN message on the terminal, the system displays the quoted string without the quotation marks.
2. DATE
   The DATE argument causes the system to display the SHUTDOWN message with the date in the dd-mmm-yy format.
3. TIME
   The TIME argument causes the system to display the SHUTDOWN message with the clock time in the hh:mm:ss format.
4. TRANSACTION
   The TRANSACTION argument causes the system to display the SHUTDOWN message with the system-determined transaction instance number as 10 characters.
5. NAME
   The NAME argument causes the system to display the SHUTDOWN message with the 6-character name of the transaction that is currently being run from the application terminal.
6. STATION
   The STATION argument causes the system to display the SHUTDOWN message with the 6-character terminal station identification on the terminal.

7. REQUEST
   The REQUEST construct causes the system to display the SHUTDOWN message with the number of minutes until the shutdown in the mm format. This is the only case where the syntax used in the FDFUPT utility differs from the standard ATL syntax.

If the response to the VALUE question requires two or more lines, you can continue the response by ending the line to be continued with a hyphen (-). Refer to Section 6.5.1.3 for more information.

For example, you can change the SHUTDOWN message with the following response.

VALUE? "System shutting down in ",REQUEST," minutes"

In the example, the phrase System shutting down in and the word minutes are quoted strings.

6.5.1.3 Extending the Value Response – If the entire text definition cannot fit on one line, you can end the line with a hyphen(-). If you end a line with a hyphen, the FDFUPT utility displays the question:

MORE?

You can respond by entering more text. Every text input line that you end with a hyphen causes the utility to repeat the question. After you complete the text definition, the utility incorporates the edit and if the utility:

1. Accepts your responses, the utility displays the message:

   % New Definition Accepted

   and returns to the command question:

   Command <EDIT>?

2. Does not accept your responses, the utility displays the message:

   % There were <number> fatal errors in this value definition, Please reenter.

   VALUE?

   Correct the errors and re-enter the message text.

6.5.2 Updating with FDFUPT

If you choose the UPDATE command, the FDFUPT utility displays the question:

Transaction Processor Name?

Specify the name of the target transaction processor. This defines which forms definition file you want to update.

If the specified forms definition file does not exist, the utility displays the following error message:

% <Transaction Processor name> does not exist

and returns to the command question:

Command <EDIT>?

You can either enter another transaction processor or exit the FDFUPT utility.

CHAPTER 7
SUPPORT ENVIRONMENT COMMANDS

With the system support commands, you can run: support utilities (such as the SETUP utility), communicate with other support terminal users, log in/log out, and enable/disable support terminal access.

7.1 BEGINNING A SUPPORT TERMINAL SESSION: The LOGIN Command

Use the LOGIN command to initiate access to the support environment. The command requires that you know either a user identification code (UIC) or a logical identification and password. Thus, the UIC or identification and password are a protection mechanism.

You enter the UIC, identification, and password into the Account File with the privileged ACCOUNT function of the SETUP utility. The LOGIN command validates the UIC and password against the Account File entries. Refer to Chapter 9 in this guide for more information on the ACCOUNT function.

The UIC format is two octal numbers enclosed in square brackets and separated by a comma such as: [ggg,mmm]. A group category shares the octal value, ggg, with the other members of the same group. The octal value, mmm, is a unique member code. That is:

```
LOGIN [300,5] TRAX
```

or something similar requires that you establish an account, [300,5], with a password, TRAX. The user identification – last name of the user – that you assign to [300,5] can be used in place of the UIC. That is, suppose you define DEVELOP as the last name associated with [300,5]. Then you could use:

```
LOGIN DEVELOP TRAX
```

to log in to the support environment.

After you log in, the system performs the following functions:

1. Sets the default UIC equal to the LOGIN UIC

   A privileged user has a group number from 1 to an octal 10. A nonprivileged user is any group number greater than an octal 10.
2. Assigns the LOGIN logical device name SY0: to the system disk (the disk that contains your files)

3. Uses the UIC and establishes the default value for the user file directory (UFD)

   A user file directory is a disk file containing your filenames and related information such as the disk location and file size. Refer to Chapter 8 in this guide for a more detailed description of the UFD.
4. Displays a system identification, the current date, and the time you logged into the terminal
5. Displays the contents of the [1,2] LOGIN.TXT file

   As system manager, you create the [1,2] LOGIN.TXT file with the EDIT command to display messages on support terminals at login. The messages inform the users of such things as:
   A. When the system is coming down for preventative maintenance
   B. A new COBOL compiler has been added to the system
   C. There is no disk space available, purge your files.

   The system displays the [1,2] LOGIN.TXT file on a support terminal when you login to the support environment. The system displays the file immediately following the login greeting. The file is created and maintained by a designated privileged user. You should have only one designated user to maintain the file. The file format is restricted to a line length of 80 characters with no restriction to the number of lines.
6. Executes your LOGIN.CMD file

   If you created the file in your UFD, the system sends the file to the indirect command file processor. Refer to the *Support Environment Programmer's Guide* for an explanation of the indirect command file processor.
7. Displays the > command prompt

   The > command prompt indicates the system is ready to accept a command.

When logging into the system, your default UFD is identical to your login UIC. If you are a privileged user and issue a SET DEFAULT command, the system alters the UFD and default UIC to the new values. If you are a non-privileged user and you issue a SET DEFAULT command, you can alter only your UFD.

The system grants you access to the system until you issue a LOGOUT command. DIGITAL recommends that at any time you plan to leave the vicinity of the support terminal, you should log it out. If you permit the terminal to remain logged into the system, anyone can access the system through your terminal.

7.2 ENDING A SUPPORT TERMINAL SESSION: The LOGOUT Command

Use the LOGOUT command to end the terminal session and make the terminal available to other users. When you log out, the system performs three functions:

1. Dismounts all private volumes associated with the session

   Dismounting a volume is explained in Chapter 8 in this guide.
2. Deallocates all private devices associated with the session

   Deallocating a device is explained in Chapter 8 in this guide
3. Aborts active, nonprivileged tasks initiated during the session.

7.3 ENABLING/DISABLING SUPPORT ACCESS TO THE SYSTEM: The SET [NO] LOGINS Command The SET TERMINAL SLAVE Command

Use the privileged SET NOLOGINS command to disable user access to the system. For example, SET NOLOGINS allows you to disable support environment users from logging into the system when you are running a transaction processor with limited system resources. When you enter the SET NOLOGINS command, a user with access to the system retains the access. However, no other user can log in to the system until you issue the privileged SET LOGINS command.

Use the privileged SET TERMINAL SLAVE command to disable or limit user access to the system. For example, the SET TERMINAL SLAVE command allows you to limit support terminal access to a current interactive unsolicited dialog task. When the task is completed, the system ignores any further keyboard entry from the slaved terminal. Use the SET TERMINAL NOSLAVE command to return a slaved terminal to general access.

7.4 CONTROLLING TASK EXECUTION

A task is an executable program. You can control and monitor the execution of a task from any support terminal in three ways. You can:

1. Execute a task using the RUN command
2. Abort a running task using the ABORT command
3. Display the status of a task using the SHOW TASKS command.

The following sections describe how to control and monitor task execution using the RUN, ABORT, and SHOW TASKS commands.

Tasks are executed at either system startup or when you issue the RUN command. When executing a task, the system is informed of the task: name, owner UIC, and task image file location. If the system executes a task at system startup, the system assigns the task name and owner UIC. If you execute a task with the RUN command, you can determine the task name. The system determines the owner UIC depending upon the privilege status of the task. If the task is privileged, the system uses your login UIC

as the owner UIC. If the task is nonprivileged, the system uses your default UIC as the owner UIC. Use the SHOW DEVICES command to display the login UIC. Use the SHOW DEFAULT command to display the default UIC.

A system task – which is executed by the system – such as the kernel error logger, remains active until terminated by the system. However, when you execute a task with the RUN command, the system removes the task upon its completion.

7.4.1 Executing Tasks: The RUN Command

Use the RUN command to execute a task. You can issue the RUN command as often as you like. The system assigns the default task name "TTnn" if you do not specify a task name. The value of "nn" is the number of the terminal from which the task is executed. To specify a unique task name, use the /TASK qualifier.

7.4.2 Aborting a Task or Command: The ABORT Command

Use the ABORT command to terminate a running task or command. To abort a command, you specify either:

1. The ABORT cmd sequence on the same terminal where you specified the "cmd" command

   The "cmd" variable is the first three letters of the command
2. The ABORT/TASK cmdTnn sequence on any privileged terminal

   The "nn" variable is the terminal number where the command was issued. You can abort any task from a privileged terminal while nonprivileged users can only abort tasks which they initiate.

You can abort a task you started with the RUN file command by specifying either the ABORT RUN or the ABORT/TASK TTnn command sequence. You can abort a task you started with the RUN/TASK:name file command by specifying the ABORT/TASK name command sequence.

You can abort the indirect command file processor task, AT.Tnn, by specifying the ABORT/TASK AT.nn command sequence. The value of "nn" is the terminal number where the task was executed.

7.4.3 Displaying Task Status: The SHOW TASKS Command

Use the SHOW TASKS command to display task names and the status of the tasks. For example, you can use the SHOW TASKS command to display the task names running on your terminal or running on all terminals. This is useful if you wish to abort a task but you are unsure of the task name.

7.5 COMMUNICATING WITH SUPPORT TERMINALS: The MESSAGE Command

Use the MESSAGE command to communicate with other support terminal users. Privileged users can send a message to either all terminals or all logged-on terminals with one MESSAGE command. Nonprivileged users can send a message to only one terminal with one MESSAGE command.

7.6 DISPLAYING COMMAND INFORMATION: The HELP Command

Use the HELP command to display the [1,2] DCLHELP.DAT file which contains the syntax for the DCL commands. If you specify the HELP command with no parameters, the system displays a complete list of the commands. For example:

```
HELP
THE FOLLOWING COMMANDS ARE AVAILABLE
ABORT          ALLOCATE       APPEND         ARCHIVE
ASSIGN         BASIC          COBOL          COPY
CREATE         DEALLOCATE     DEASSIGN       DELETE
DIRECTORY      DISMOUNT       EDIT           INITIALIZE
LIBRARIAN      LINK           LOGIN          LOGOUT
MACRO          MERGE          MESSAGE        MOUNT
PRINT          PURGE          RENAME         RUN
SET            SHOW           SORT           START
STOP           TYPE           UNLOCK
- FOR MORE INFORMATION TYPE 'HELP' FOLLOWED BY
THE COMMAND
```

To display the syntax of a particular command, specify HELP followed by the command name. For example if you enter:

```
HELP ABORT
ABORT[/QUALIFIER]  [TASKNAME]
        COMMAND
        TASK
        DUMP
```

The display indicates you can specify the optional /COMMAND, /TASK, or /DUMP qualifier and task name. (Refer to Chapter 11 in this guide for a full description of the command.)

When the system displays a qualifier followed by two asterisks (**), you can obtain further syntax information by respecifying the same HELP command and appending the qualifier. For example if you enter:

```
HELP SET
```

The system displays:

```
SET FUNCTION
        DEVICE      **
        DEFAULT [DEVICENAME] AND/OR [[UFD]]
        TERMINAL  **
        PROTECTION FILESPECLIST CODE
          WHERE CODE IS (SYSTEM:RWED,OWNER:RWED,
          GROUP:RWED,WORLD:RWED)
        [NO] LOGINS
        QUEUE       **
```

The display indicates specific syntax formats such as:

```
DEFAULT [DEVICENAME] AND/OR [[UFD]]
```

and indicates additional syntax information is available for: DEVICE, TERMINAL, and QUEUE functions. For example if you enter:

```
HELP SET QUEUE
```

The system displays:

```
SET QUEUE QUEUENAME      OPTION[,OPTION,...]
          ENTRY:(N,N)    JOB:JOBNAME
                         UPPERCASE
                         LOWERCASE
                         [NO] WIDE
                         PAGES:N
                         PRIORITY:N
                         FORMS:N
                         LENGTH:N
                         [NO] RESTART
                         [NO] FLAG,PAGE
                         AFTER:(DD-MMM-YY HH:MM)
                         HOLD
                         RELEASE
```

The display indicates the syntax for the SET QUEUE command.

CHAPTER 8
MANAGING DEVICES, VOLUMES, AND FILES

System resources include devices identified as line printers, support terminals, magnetic tapes, and disks. You inform the system of the devices and the characteristics during the system generation procedure. The system initializes the device characteristics, such as line width and speed, at system startup. This chapter discusses managing these resources. Such functions as permitting one user to access a device or permitting several users to access a device are presented in this chapter. You can also determine who can access a file or group of files by defining access codes to volumes, directories, and files. A volume is either a reel of magnetic tape or a disk pack. In addition to device, volume, and file accessibility, you can determine how to prepare a new volume for use.

Device, volume, and file resource management includes maintaining device and volume integrity. You must make a daily check of the kernel Error Logger and ensure any possible deficiencies are corrected; refer to Chapter 9 in this guide to print a report of the kernel error logger activities. Refer to the *User Mode Diagnostics Manual* to use the diagnostics. Finally, verify disk volume integrity by using the VFY utility discussed in Chapter 9 in this guide.

8.1 MANAGING DEVICES

Devices are resources which you can make available to one user, everyone, or no one. In addition to device accessibility, you can change device characteristics. Over a period of time or from task to task, the characteristics you specified at system generation are not desired. Section 8.1 presents information to make temporary changes to the system. That is, the change remains in effect until you either make further changes or until you boot in another system image. Booting in a system image causes the system to use the startup procedures which initialize the devices. To make permanent device changes, refer to the SETUP utility description located in Chapter 8 in this guide.

Once you have defined device accessibility, you may wish to change or verify device assignments. Device assignments determine the devices which tasks are to access. Since device assignments can be made regardless of the accessibility of devices, you must exercise care to ensure the accessibility of assigned devices.

8.1.1 Changing Device Characteristics: The ALLOCATE Command The DEALLOCATE Command The SET TERMINALS Command The SET DEVICES Command

You can explicitly specify a device as a shared or nonshared device. Use the SET DEVICE command to specify a device as a public device. Anyone can access a public device. Use the ALLOCATE command to specify a device as a private device. Nonprivileged users can access devices they allocated, but not other private devices. You must remove a public device status before allocating a device. For efficient resource management, use the DEALLOCATE command to deallocate a private device when you no longer need the device. Privileged users can deallocate any private device, while nonprivileged users can only deallocate devices they allocated. The system automatically deallocates devices when the owner logs off the system.

If someone requests a volume to be mounted and accessed by a batch job, the following message appears on TT0:.

```
05-Jul-78 BATCH JOB – TEST STARTED ON VT5:

LOAD "LABEL" "ON A MM:DRIVE.

ALLOCATE AND ASSIGN DRIVE MM0:. TERMINAL = VT5:.

START PROCESSOR BAP6.
```

Respond by loading the tape on any magnetic tape drive and:

1. Allocate a MM: drive
2. Assign the selected MM: drive to MM0: as indicated in the message and ensure to use the /TERMINAL = VT5: qualifier
3. Start the BAP6 processor.

The SET TERMINAL and SET DEVICE commands change device characteristics that you specify during the system generation procedure. Privileged users can change any terminal or device characteristic. Nonprivileged users can change the terminal they are using and the device characteristics of the devices they allocated. Use the SHOW TERMINAL and SHOW DEVICE commands to display current device characteristics.

Since the VT52 terminal contains modes that are not utilized by the TRAX system, you can invoke one of these modes if you inadvertently display a binary file, such as an object or task image file, on the terminal. If such a situation occurs, display the [1, 1] NORMAL. V52 file at the support terminal. To display the file, use the COPY command from another support terminal and copy the file to the troubled support terminal.

8.1.2 Making and Changing Device Assignments: The ASSIGN Command The DEASSIGN Command The ASSIGN/REDIRECT Command

A device name can be either a logical name or a physical device name. The system assigns a device a physical device name during the system generation procedure. A logical name uses the same format as a physical device name and is initially unassigned. Since some tasks use logical device names to access devices, logical names must be assigned to physical devices before such tasks can run. At system start up, the system assigns the system logical names such as SY0: and LB0:. If your installation uses logical names, it is either the responsibility of the programmers to make local assignments or the responsibility of the system manager to make global assignments of logical names to physical devices before running tasks which depend on such assignments. Using logical names is especially useful if you are not certain which devices are available when you need them. For instance, you can choose a logical name that everyone can use for a certain data pack. Then regardless of where the data pack resides, everyone can access it by using the same logical name.

Use the ASSIGN command to link a logical name to another logical name or physical device name. Assignments are made at three levels: local, login, and global. When two or three assignment levels specify the same device names, the system resolves the conflicting assignments based upon an established priority. The priority list appears as follows:

1. Local
   Tasks recognize local assignments before login and global assignments. Local assignments apply to tasks executed from the terminal where you made the assignments. You make local assignments with the ASSIGN/LOCAL command. As a privileged user, you can also make local assignments for other terminals by using the /TERMINAL qualifier.

   When you specify the SET DEFAULT command, the system reassigns the login logical name, SY0:, to the local device name you specify. The system accesses the login logical name as the system device, which contains your files.
2. Login
   Tasks recognize login assignments before global assignments. Login assignments apply to tasks executed from the same terminal. The system assigns the login logical name, SY0:, when you issue the LOGIN command. The system assigns the login logical name, SY0:, to the default system device. You can specify a different system logical name when creating or modifying the login account with the ACCOUNT function of the SETUP utility. Refer to Chapter 9 in this guide for more information on the ACCOUNT function of the SETUP utility.

The system assigns the system logical name to a physical device name at system startup. You can request a display of this information by specifying the SHOW DEVICE command.

3. Global
   Tasks recognize global assignments if there are no local and login assignments. Global assignments apply to all tasks running in the system. You make global assignments with the privileged ASSIGN/GLOBAL command.

Privileged users can assign and deassign any local, login, or global assignment, while nonprivileged users are restricted to the local assignments that they make. Use the DEASSIGN command to remove a local, login, or global logical name assignment.

Use the privileged ASSIGN/REDIRECT command to change all device requests from one physical device to another physical device. The system resolves all logical name assignments, which end with a redirected physical device name, to the new physical device name. The ASSIGN/REDIRECT command is especially useful if a required device is inoperable. The system:

1. Redirects an idle device immediately
2. Redirects a device with a current task access after you abort the task
3. Redirects a private device after you deallocate it
4. Redirects a mounted device after you dismount it

Use the SHOW DEVICE and SHOW ASSIGNMENTS commands to display device assignments. If you specify the SHOW ASSIGNMENTS command, the system displays the: local, login, and global assignments. If you specify the SHOW DEVICE command, the system displays the physical device names, status, and system logical names. The system assigns the system logical names, such as: CO0:, CL0:, SP0:, LB0:, and SY0:, at system startup.

8.2 MANAGING VOLUMES

Volumes are file-structured disks and magnetic tapes. Before a task can access a volume, the volume must be connected to a device. The system mounts the system disk volume at system startup. You can mount a volume by using the MOUNT command. Before a new volume can be mounted, you must create a file structure. Use the following procedure to prepare a volume.

1. Use the BAD function of the SETUP utility to identify bad areas on a disk volume. Magnetic tape volumes are certified error-free and do not require this step.
2. Initialize disk and magnetic tape volumes with the INITIALIZE command.

3. Mount the volume with the MOUNT command. Tasks can now access magnetic tape volumes while disk volumes require the next step.
4. For disk volumes, use the CREATE command to write the required number of user file directories on the volume. A disk volume requires at least one user file directory. Tasks can now access disk volumes.

8.2.1 Initializing a Volume for File Access: The INITIALIZE Command

Use the INITIALIZE command to create a file structure on a volume. (Removable disk packs and magnetic tapes are called volumes.) Privileged users can initialize any disk or magnetic tape volume. Nonprivileged users can initialize a volume on a device that they allocated. The system writes a master file directory – [0, 0] 000000. DIR; 1 – on disk and creates a volume label and dummy file on magnetic tape. A master file directory is a file which contains the filenames and locations of the user file directories. Volumes can be reinitialized, but the system destroys the existing files on the volume.

The system writes a dummy file on magnetic tape or control files on disk. The [0, 0] INDEXF. SYS index file is one of the control files on a disk volume and contains:

1. Bootstrap Block
2. Home Block
3. File Headers

The bootstrap block is used only on the operating system volume, and contains a program that loads the operating system into memory. It is physically the first block on the volume. All volumes have an area for this bootstrap block. However, if the volume is not the operating system volume, the bootstrap block area contains a different program. This program displays a message on the system console indicating that the volume is not the operating system volume, but is rather a volume containing only user files.

The home block is normally the next block on the volume and identifies the disk. In some cases, it may not physically follow the bootstrap block. If for some reason the home block cannot be read (physically unusable), an alternate block is selected for use as the home block. This block provides specific information about the volume and default values for files on the volume. Among the items in the home block are the following:

1. The volume name

   You must specify the volume label to mount and dismount a volume.
2. Information to locate the remainder of the index file
3. The maximum number of files that can be present on the volume volume at any one time

4. The user identification code (UIC) of the owner of the volume

   When a task attempts to access the volume, the system compares the task owner UIC with the volume owner UIC. This relationship defines the access to the volume. The accesses are: system, owner, group, and world. Refer to Section 8.3.4 for more information.
5. Volume protection information

   The volume protection information specifies which user tasks can read, write, extend, and delete files from the volume. Refer to Section 8.3.4 for more information.

Volumes contain several copies of the home block to ensure against accidental destruction of this information and the consequent inability to locate other files on the volume.

The bulk of the index file consists of file headers. Each file header describes one file on the volume, and contains such information as the file ownership, protection, creation date and time. Most important, the file header contains a list of extents that make up the file, providing the block numbers where the file is physically located on the volume.

When creating a file, you assign a filename. The system places the filename and file identifier in a directory which contains entries for each file. These entries point to the file location. When accessing the file, you supply this filename which the system associates with the file identifier. The file identifier in turn points to the location of the file header. The file header contains a list of the extent(s).

8.2.2 Mounting a Volume for File Access: The MOUNT Command

Use the MOUNT command to connect a volume logically to a device. Once you mount the volume, tasks access the volume by specifying the associated device name. The system verifies:

1. The device is on-line.
2. You are mounting the correct volume by comparing the volume label you specify with the label on the volume. If you specify the correct label, the system continues the mounting function. If you specify an incorrect volume label, the system aborts the command.

Before each file access, a task verifies that a volume is mounted.

Privileged users can mount volumes on any device. Nonprivileged users can mount volumes on the devices that they allocate. For efficient resource management, you should dismount volumes, using the DISMOUNT command, when you no longer need the volumes.

8.2.3 Creating and Deleting a Directory: The CREATE Command The DELETE Command

Use the CREATE command to create files and user file directories. The privileged SETUP utility creates a user file directory when you request the utility to create an account (that is, a user identification code). You create additional directories with the CREATE command by appending the /DIRECTORY qualifier to the command. The system creates the directory entry that you specify in the master file directory – [0, 0] 000000. dir; 1 –. If you wish to remove a directory, you must first delete all the files listed within the directory and then delete the directory from the master file directory. Refer to Section 8.3 for a description of how the system names a directory. To create a directory, you must have read, write, and extend access to the volume and the master file directory. To delete a directory, you must have delete access to the volume and the master file directory. Refer to Section 8.3.4 for more information on file protection.

8.2.4 Dismounting a Volume: The DISMOUNT Command

Use the DISMOUNT command to disconnect a volume logically from a device. The system verifies that you are dismounting the correct volume by checking the volume label. If you specify the correct volume label, the system continues the dismounting function. If you specify an incorrect volume label, the system aborts the command. The system immediately dismounts an idle volume. A task accessing a file causes the system to mark the volume for dismount, inhibiting other file accesses. The system then suspends dismounting the volume until either the task completes the access or the task is aborted. The system issues a message to the system logical device, CO:, when the dismount operation is complete.

Privileged users can dismount any volume, while nonprivileged users can only dismount volumes that they mount. The system dismounts volumes connected to private devices when the owner logs off the system. Privileged users' nonprivate volumes remain mounted when the owner logs off the system.

8.2.5 Archiving and Restoring Volumes: The ARCHIVE Command

The ARCHIVE command backs up and restores volumes and files. Refer to Section 8.3 for archiving and restoring files. Volume archiving is the act of copying a disk volume to either a magnetic tape volume or another disk volume. Restoring is the act of copying an archived magnetic tape volume to a disk volume. Before you archive, you must dismount the source disk volume and the destination volume. The system consolidates the disk area if you specify the merge option. If you specify a disk destination volume, you can immediately mount and access the resultant disk copy. If you specify a magnetic tape as the destination volume, you must restore the resultant magnetic tape volume to a disk volume before you can access the volume.

8.3 MANAGING FILES

The smallest addressable unit of information on a disk or magnetic tape is a block. Magnetic tapes have a variable block size and do not have a set number of blocks on them. A disk block is 512 eight-bit bytes, and is treated as a single unit for physical transfer between a device and memory. Blocks are numbered consecutively from 0 to n (the value of n depends on the size of the disk or magnetic tape).

Blocks are grouped into a contiguous cluster called an extent, the basic unit of volume space allocation. An extent can contain all or part of a file. The file is a logcially related collection of data treated as a unit. If enough contiguous area is available on a disk, the entire file can be allocated to one extent. Usually, you do not wish initially to reserve the entire amount of space and the file is extended non-contiguously. So, a file can consist of one or more extents located in separate or contiguous areas on the disk. You determine the number of blocks in an extent when you initialize or mount a volume.

To locate files, the system lists the files in a directory. A directory is a file listing the names of files and the pointers to each file header. The file header contains information about the file owner and the physical location of the file segments. The actual directory filename is a concatenation of the group and member numbers, with a filetype of: ".DIR". The system accesses the [203, 165] directory as filename, [0, 0] 203165.DIR. The system lists all user file directories (UFD) in the master file directory (MFD) for each volume.

The system creates a user file directory when you request the system to create a UIC or an account. The UIC and UFD formats and values are identical. You can create an additional UFD with the CREATE/DIRECTORY command.

8.3.1 File Format Specification and Defaults

There are many situations in which you must supply a complete file specification. Five different identifiers or fields are required to specify a file. They are:

dev: — is the physical or logical device name where the system can find the volume containing the file. (See Table 8-1.)

The system device initially defaults to the SY0: logical name. Use the SHOW DEFAULT command to display the physical name assigned to SY0:. You can change the default with the SET DEFAULT command.

[group, member] is the user file directory where the file is listed. The system initially defaults to the UFD established:

1. At system generation (see the *System Generation Manual*).
2. By means of the LOGIN command.
3. By means of the SET DEFAULT command.

Use the SHOW DEFAULT command to display the default user file directory.

filename. is the name of the file. You must specify either a filename or an asterisk (wildcard). The file is an alphanumeric string from 1 to 9 characters. A period (.) always separates the filename from the filetype.

filetype; is a 3-letter mnemonic identifying the nature of the contents of the file. Table 8-2 lists the standard types. For example, OBJ indicates that the contents are object (compiled) code. You must always specify a semicolon (;) to separate the filetype from the version.

A task selects a default filetype depending upon whether the file is an input or output file.

n is an *octal* number: -1, 0, and 1 to 77777. The version number differentiates between files with the same filename and type.

The numbers -1 and 0 have special significance: -1 implies the lowest existing version of a file, and 0 implies the highest existing version.

A task selects the highest version if you omit the field.

The format of a complete file specification is

dev: [group, member] filename, filetype; n

If you specify a field, you change the default for that field. That is, the command line:

MACRO TEST + DB1: TEST1 + [30, 30] TEST2 + TEST3

uses the system defaults for the TEST file. The TEST1 file defaults are used except for the device, DB1:. The TEST2 file defaults to the DB1 disk. The TEST3 file defaults to the DB1: disk and [30, 30] UFD.

Table 8-1 Device Names

Physical Devices	Device Name
DISK (RP04/RP05/RP06)	DBnn:
(RK07)	DMnn:
(RM02/RM03)	DRnn:
Line Printer (LP11)	LPnn:
Magnetic Tape (TU16/TU45/TE16)	MMnn:
Terminal (DL11/DZ11)	TTnn:
Communications Option (DMC)	XMnn:
Network Device	NTnn:
Logical Devices	
Console Listing	CLn:
Console Output	COn:
Input Terminal	TIn:
System Default Device	SYn:
System Spool Device	SPn:
Virtual Terminal	VTn:
System Library Input Device and Queue File	LBn:
Null Device	NLn:

Table 8-2 Standard File Types

Type	File Contents
ATL	Application terminal language source
B2S	BASIC language source
CBL	COBOL language source
CMD	DCL commands (an indirect command file or batch file)
DAT	Data (as opposed to a program)
DIR	Directory (such as a User File Directory)
DOC	Document (such as a manual or memo)
LOG	Log file created by a batch processor or transaction processor
LST	Printer listing
MAC	MACRO source
MAP	LINK memory allocation map
MLB	User macro library
OBJ	Object program (output from a compiler and MACRO assembler)
ODL	LINK overlay description
OLB	Object module library
RNO	RUNOFF source
SML	System macro library
STB	Symbol table
SYS	Bootable system image or related file
TMP	Temporary
TSK	Task image
TXT	Text

8.3.2 Standard Filetypes

The system has a standard set of filetypes used by all DIGITAL-supplied software to reflect actual file contents. Although you can assign arbitrary 3-letter filetypes, use the standard types described in Table 8-2.

The DIGITAL supplied software documentation, that pertains to the individual software, describes the defaults.

8.3.3 The Asterisk Convention (Wildcards)

The asterisk convention allows you to specify more than one file by placing an asterisk (*) or "wildcard" in one or more fields of the file specification. The system interprets the asterisk as all files that satisfy the remaining fields.

An asterisk can be placed in any part of the file specification, except the device field which must be explicitly supplied or defaulted to SY:.

The following example illustrates the use of wildcards. The command line:

```
DELETE PROG. MAC; 1, PROG. OBJ; 1, PROG. TSK; 1
```

deletes the three individual files specified. Since the three files use the same filename and version number but different filetypes, the following command line deletes the same files:

```
DELETE PROG.*; 1
```

The command also deletes any other files (on the default device) named PROG with a version number of 1. You should check existing files before using wildcards, for example, by issuing the command line:

```
DIRECTORY PROG.*; 1
```

8.3.4 File Protection

When you create a file, the system places the filename in a directory and stores the default User Identification Code (UIC) in the file header, indicating the owner of the file. Volumes, directories, and tasks also have an owner UIC. You can display the file owner UIC (including directories) with the DIRECTORY/FULL command. To display the volume owner UIC, specify the /SHOW qualifier when you mount the volume. A task owner UIC is explained in Chapter 7 in this guide.

When you want to access a file, you must know where the file resides; which includes the volume, UFD, filename, type, and version. Knowing where the file resides, however, is not sufficient to guarantee access. You must also satisfy protection masks associated with the volume, UFD, and file.

Files are accessed through tasks. Each task has a name and owner UIC. The task name and owner UIC assignments are discussed with the RUN command description. The system reads the task UIC to determine the task group and owner code. A group and owner code are defined in terms of the relationships between the task UIC and the owner UIC of the volume, user file directory, and file to be accessed. The relationships are defined as:

1. System – Tasks running under a UIC group number from 1 to 10 are system tasks.
2. Owner – Tasks that run under the same group and member numbers as the owner UIC.
3. Group – Tasks that run under the same group number (the first octal number within the brackets) as the owner UIC.
4. World – Any task in the system is a member of the world.

Each of these relationships has an associated protection code which defines the type of access. The protection code is defined as four types of accesses:

1. READ
   Read access permits a task to read the file. Such functions as displaying the contents of a file or executing a task image require read access.
2. WRITE
   Write access permits a task to change the contents of a file, Editing a file, compiling or assembling require write access.
3. EXTEND
   Extend access permits a task to enlarge the size of a file with the intention of adding more to the file. If a task creates a file, the system allocates volume space for the file. The task attempts to extend the file when all the allocated space is used and more is required.
4. DELETE
   Delete access permits a task to remove the file entry from the UFD and release the occupied space.

A task attempts the volume, UFD, and file accesses by using the following sequence: world, group, owner, system. That is, if the world protection code permits the access, the other codes are not checked. If the world protection code prohibits the access, the task attempts the access through the group protection code. Failing to access through the group protection code, the system attempts the access through the owner protection code and finally through the system protection code. The system grants file access if the volume, UFD, and file protection codes permit the task access.

You specify the protection access rights by using:

1. The INITIALIZE command. This command allows you to specify the owner UIC, default volume protection mask, and file protection mask for all the files created on the volume.
2. The CREATE/DIRECTORY command. This command establishes access rights to a user file directory. If you do not permit a group read access to a directory, you prohibit the group the access to all the files listed within the directory. If you permit a group read access to a directory, you can prohibit individual access to the files with the file protection mask.
3. The SET PROTECTION command. This command permits the file owner or any privileged user to alter file access rights, including directories.
4. The MOUNT command. This command allows you to change the default volume protection mask. The MOUNT protection mask overrides the default protection established with the INITIALIZE command. The MOUNT command does not change the default protection mask. That is, when you dismount the volume and then mount the volume, the system uses the default protection mask and UIC unless you specify different values.

8.3.5 Examples of File Specifications

1. MACRO/OBJ:DB1:[200, 200] CRGPT. OBJ DB1: [200, 200] CRGPT. MAC
   Assemble the MACRO source file CRGPT. MAC and create the object output file CRGPT. OBJ. Both files are on DB1: under UIC [200, 200].
2. DELETE [200, 200] SBG. OBJ; 5
   Delete the file SBG. OBJ; 5. The output filename string is not applicable and therefore is omitted. The input device defaults to SY:.
3. DELETE CATHDAT,*;*
   Delete all files with the name CATHDAT, regardless of type or version. The system uses default values for the device and UIC.

8.3.6 Displaying and Changing System Defaults: The SHOW DEFAULT Command The SET DEFAULT Command

Use the SHOW DEFAULT command to display, at the issuing terminal, the default system device and User File Directory (UFD) code. The system uses these defaults with file specifications. If a file specification does not include a device and/or UFD, the system uses the default values to complete the file specification.

Either the system device or the UFD can be changed with the SET DEFAULT command. When working with a device or UFD other than those assigned at login time, you must specify explicitly the device and

UFD in the file specifications. By using the SET DEFAULT command to specify the more frequently used device and UFD, you can simplify the file specification. The system applies the SET DEFAULT command to the commands and tasks that you issue after the SET DEFAULT command. The SET DEFAULT command does not affect active commands or tasks.

When a privileged user specifies the command, the UFD and UIC are both changed to the new value. When a nonprivileged user specifies the command, the UIC remains unchanged. As a result, privileged users can access any file as its owner by first changing the default UIC/UFD to the file owner UIC code. The DIRECTORY/FULL command displays a file owner UIC and access rights. Refer to Chapter 11 for more information.

8.3.7 Archiving and Restoring Files: The ARCHIVE Command

The ARCHIVE command backs up and restores volumes and files. Refer to Section 8.2 for archiving and restoring volumes. Archiving files is the process of copying disk file(s) to either a magnetic tape volume or another disk volume. Restoring file(s) is the process of copying either archived magnetic tape file(s) or archived disk file(s) to a disk volume. You must mount the source and destination volumes. When you archive to a disk volume and use the /VOLUME qualifier, you can immediately mount and access the new volume. If you specify a magnetic tape volume as the destination volume or if you specify file archiving, you must restore the copy to a disk volume and access the disk volume.

8.3.8 Deleting and Purging Files: The DELETE Command The PURGE Command

You can delete files from a User File Directory and release the occupied space with the DELETE and PURGE commands. The file access rights must permit you to delete the file. The file access rights are displayed with the DIRECTORY/FULL command and changed with the SET PROTECTION command. To delete a file, you must specify the filename, filetype, and version or the wild card attribute (*).

When you specify the PURGE/KEEP:n command, the system saves the latest n versions and deletes all earlier versions. The system assigns a default value of 1 to n if you do not specify its value.

For instance, to purge all but the latest two versions of the SAMPLE.DEL file, use the command:

```
PURGE/KEEP:2 SAMPLE.DEL
```

To delete all versions of the file, use the command:

```
DELETE SAMPLE.DEL;*
```

CHAPTER 9
USING SUPPORT ENVIRONMENT UTILITIES

This chapter presents the utilities you use to maintain the support environment. There are utilities to change support terminal characteristics, recover sections of the system disk, detect bad blocks on a disk before initializing it, and add or delete user access to the support environment. The utilities are:

1. ANALYSIS
   The ANALYSIS utility, which must run in a full support environment, provides a report of a crash dump file which the system creates if it crashes.
2. SETUP
   The SETUP utility, which must run in full support environment, provides a variety of functions which are fully listed in Chapter 2. The following list represents the SETUP functions which are presented in this chapter:
   A. ERRORLOG
      If you select the ERRORLOG function, you can print a report of all hardware errors detected by the kernel error logger. The ERRORLOG function is discussed in Section 9.2.
   B. ACCOUNT
      If you select the ACCOUNT function, you can add, modify, delete, and list user accounts. The ACCOUNT function is discussed in Section 9.3.
   C. TERMINAL
      If you select the TERMINAL function, you can set up or change the support environment terminal specifications file located on the system disk. Use the SET command to immediately change the terminal specifications on-line. The TERMINAL function is discussed in Section 9.4.
   D. REDO
      If you select the REDO function, you can incorporate the terminal changes on the disk-resident system image. The REDO function is discussed in Section 9.5.
   E. TRANSFER
      If you select the TRANSFER function, you can move user files from an old system disk to a new system disk. The TRANSFER function is discussed in Section 9.6.
   F. BAD
      If you select the BAD function, you can perform bad block-checking on a disk. The BAD function is discussed in Section 9.7.

G. RENEW
If you select the RENEW function, you can recreate the disk-resident system image from existing files. The RENEW function may be required if the integrity of the disk-resident system is destroyed. The RENEW function is discussed in Section 9.8.

H. STARTUP
If you select the STARTUP function, you can recreate the system initialization commands executed for you by the INIT utility. The STARTUP function is discussed in Section 9.9.

3. VFY
The VFY utility, which must run in a full support environment, verifies the file structures on a disk volume.

9.1 ANALYZING A CRASH DUMP (ANALYSIS)

If a kernel crash occurs, the system dumps the entire contents of memory to the contiguous file [0,0]CRASH.DMP;1. After the system recovers or at system startup, it determines if [0,0]CRASH.DMP;1 was copied to the contiguous [1,1]CRASH.CDA;1 file. If the file was not copied, the system copies the file provided enough contiguous is space available. If a file of the same name already exists, the system overwrites that file. The system displays an error message if there is not enough contiguous space available.

NOTE

Do not delete the [0,0]CRASH.DMP;1 and [1,1]CRASH.CDA;1 files once the system creates them.

Later, you can invoke the privileged Transaction Processing Dump Analyzer Utility, ANALYSIS, in a full support environment to provide a partial, readable display of the contents of the file. The output consists of a formatted and annotated breakdown of the various elements of the transaction processing system. Any detected inconsistencies are noted. The elements to be displayed include:

1. General system information
2. Task information
3. Partition information
4. Device information
5. Node pool.

A second function of the utility is to copy the dump from the [1,1]CRASH.CDA;1 file to another file. Copying a file is useful in order to save the dump for future reference or to prepare a copy to be forwarded to a software specialist for analysis. You may request both functions at the same time.

The ANALYSIS output is used by the software support specialists as an aid in analyzing system failures. It may be invoked from any privileged terminal, but an understanding of the output requires knowledge of the transaction processor executive system. It is recommended that you run the utility as soon as possible after a crash.

You invoke the ANALYSIS utility with the command:

```
RUN $ANALYSIS
```

The ANALYSIS dialog proceeds as follows:

1. Input specification <[1,1]CRASH.CDA;1>?

   Specify the file containing the crash dump.
2. Output specification <LP:>?

   Specify the device or file where the output is to be written. If you specify a file, the default filetype is LST. The system automatically spools the file. If you do not want a printed copy or file, respond with NONE.
3. Copy specification <NONE>?

   Respond with a file specification if you want to copy the [1,1]CRASH.CDA;1 file to another file. The default filetype is CDA. The system does not accept NONE if you specified NONE in Question 2.

The ANALYSIS dialog ends with Question 3. The ANALYSIS utility creates the file(s) and, if you indicated an output specification, spools a file to the line printer.

9.2 CREATING A KERNEL ERROR REPORT (SETUP)

Use the ERRORLOG function of the SETUP utility to create a print-formatted report from the error log file.

You invoke the SETUP utility with the command:

```
RUN $SETUP
```

The SETUP utility displays the question:

```
Function <EXIT>?
```

Respond with:

```
ERRORLOG
```

The ERRORLOG function displays the question:

```
What is the report file or device?
```

You can respond with a file specification, a line printer device name, and a terminal device name. If you respond with a file specification, the ERRORLOG function creates the file but does not spool it to the line printer. Use the DCL PRINT command to spool the file to a line printer. If you respond with a line printer device name, the ERRORLOG function creates a temporary file, spools it to the line printer, and deletes the file. If you respond with a terminal name, the ERRORLOG function formats the report and displays it on the terminal.

The ERRORLOG function:

1. Renames the [1,6] ERR.TMP kernel error log file to [1,6] ERROR.TMP
2. Creates another [1,6] ERR.TMP file for the system to write additional kernel errors

   No more than 1 ERR.TMP file should exist at any given time. There might be more than one if the system crashes or was shut down improperly. In this case, use the DCL RENAME command to change the name to the [1,6] ERROR.TMP file. You should be careful to preserve the correct version number.
3. Preformats the [1,6] ERROR.TMP file for inclusion into the [1,6] ERROR.SYS file
4. Creates the [1,6] ERROR.SYS file if you deleted or renamed the file
5. Appends [1,6] ERROR.TMP file to the [1,6] ERROR.SYS file and then deletes the [1,6] ERROR.TMP file
6. Creates the report from the [1,6] ERROR.SYS file.

When the report is completed, the SETUP utility terminates the ERRORLOG function and returns to the SETUP utility question:

Function <EXIT>?

Respond with EXIT to terminate the SETUP utility.

9.3 MAINTAINING THE ACCOUNT FILE (SETUP)

The SETUP utility provides the facilities for account file maintenance. When a user tries to log into the system, it checks the LOGIN command parameters against the account file to determine whether the user should be allowed access. The account file describes all the user identification codes (UIC) that you authorized for use within the system. One UIC can have several users by assigning a distinct password for each user.

NOTE

Although the ACCOUNT function permits you to create two different UIC codes with the same identification and password, it is not recommended since both users can find themselves in the same account if they log in using the identification and not the UIC code.

You invoke the SETUP utility with the command:

```
RUN $SETUP
```

The utility displays the question:

```
Function <EXIT>?
```

Respond with:

```
ACCOUNT
```

The ACCOUNT dialog begins with the question:

```
Account Function <EXIT>?
```

The ACCOUNT functions are:

1. ADD
   The ADD function adds an account. Refer to Section 9.3.1 to add an account.
2. DELETE
   The DELETE function deletes an account. Refer to Section 9.3.2 to delete an account.
3. EDIT
   The EDIT function modifies an account or examines an account. Refer to Section 9.3.3 to edit an account.
4. LIST
   The LIST function lists all or some of the accounts. Refer to Section 9.3.4 to list the accounts.
5. EXIT
   The ACCOUNT function records the account file changes as you make them. The EXIT function terminates the ACCOUNT function, incorporates the changes into the account file, and returns to the SETUP question, Function <EXIT>?. The ACCOUNT function temporarily disables logins while effecting the account file changes.

   If you use CTRL/Z to terminate the SETUP utility, the utility remembers the changes, but the changes will not be effective for the purposes of logging in until after you EXIT the SETUP utility at some later time.

The remaining sections describe the dialog for each ACCOUNT function.

9.3.1 Adding an Account

The ADD function establishes a record for each account that you add and creates a user file directory (UFD) if it does not exist.

The ADD function dialog proceeds as follows:

1. Account Function <EXIT>? ADD
2. What is the account's UIC?

   Respond with the user identification code to add. The form is n, m where n is the group code and m is the member code.
3. What is the password?

   Specify the new password.
4. Last name?

   Specify the last name of the user of the account. The user will use this parameter with the LOGIN command to log into the system.
5. First name?

   Specify the first name of the user of the account.
6. Default system device <SY>?

   Specify the default system device for the user.

   The system assigns the default system device associated with the UIC and password when a user logs into the system. Refer to Chapter 8 for more information.

The ACCOUNT function adds the account and returns to Question 2 for specifying additional accounts.

Respond with the ESCAPE key to terminate the ADD command dialog and return to Question 1.

9.3.2 Deleting an Account

The DELETE function removes a record from an account.

The DELETE function dialog proceeds as follows:

1. Account Function <EXIT>? DELETE
2. What is the account's UIC?

   Respond with the user identification code from which you wish to remove an account. The form is n, m where n is the group code and m is the member code.
3. Is the password <YES>?

   Respond with either a YES or NO. There may be multiple passwords for an account, each password corresponding to a record in the account.

   If the DELETE function displays the correct password, respond with YES. If you are deleting the final password for the account, the DELETE function continues with Question 4. If you are not deleting the final password for the account, the DELETE function returns to Question 2:

   What is the account's UIC?

Respond with another UIC or the ESCAPE key. The ESCAPE key terminates the DELETE command dialog and returns to Question 1 where you can terminate the DELETE function.

If the DELETE function displays an inappropriate password, respond with NO. The DELETE function repeats Question 3 if there are more passwords for the account. If there are no more passwords, the DELETE function displays an error message and returns to Question 3.

Respond with another UIC or the ESCAPE key. The ESCAPE key terminates the DELETE command dialog and returns to Question 1.

4. Delete directory and files for this UFD <YES>?

   If you respond with YES, the DELETE function:
   A. Removes the directory associated with the account
   B. Removes all the files listed within the directory.
   If you respond with NO, the files and directory remain on the volume.

   In either case, the dialog returns to Question 3.

   Respond with another UIC or the ESCAPE key. The ESCAPE key terminates the DELETE command dialog and returns to Question 1.

9.3.3 Editing an Account

The EDIT function changes a record for an account.

The EDIT function dialog proceeds as follows:

1. Account function <EXIT>? EDIT
2. What is the account's UIC?

   Respond with the user identification code of the account that you want to change. The form is n,m where n is the group code and m is the member code.
3. Last name <existing one>?

   Respond with a carriage return to retain the last name; respond with a different last name to change it.
4. First name <existing one>?

   Respond with a carriage return to retain the first name; respond with a different first name to change it.
5. Password <existing one>?

   Respond with a carriage return to retain the password; respond with a different password to change it.
6. Default system device <existing one>?

   Respond with a carriage return to retain the default system device; respond with a different device name to change the default system device.

The EDIT command dialog ends with Question 6. The ACCOUNT function records the changes and returns to Question 2:

What is the account UIC?

Respond with the ESCAPE key to terminate the EDIT command dialog and return to Question 1.

9.3.4 Listing an Account

The LIST function lists the accounts. The LIST function dialog proceeds as follows:

1. Account function <EXIT>? LIST
2. Listing device <LP:>?

   Respond with the device or file specification where you want the print-formatted list of accounts.
3. Should the passwords appear on this list <YES>?

   Respond with NO to delete the passwords from the listing; respond with YES to include the passwords in the listing.

The ACCOUNT function formats the listing as you specify. If the listing is sent to a spooled printer, the ACCOUNT function creates the file and spools the file to the line printer. Control returns to Question 1.

9.4 CHANGING DISK-RESIDENT TERMINAL CHARACTERISTICS (SETUP)

Support terminal characteristics can be changed on the current system or on any disk-resident system image. Use the SET command to change support terminal characteristics on the current system. Use the TERMINAL function of the SETUP utility to change a disk-resident file containing all support terminal characteristics. To incorporate the support terminal characteristics file into a disk-resident system image, run the REDO function of the SETUP utility.

Terminals are connected to the system on either DZ or DL lines. Terminals connected to a DL line set the transmit/receive speed with switches on the terminal. Also, terminals connected to a DL line cannot be designated as a remote terminal (that is, connected to telephone lines).

The TERMINAL function includes all the dialog for terminals connected to DZ lines. Consequently, the TERMINAL function omits the dialog for setting the terminal speed and remote status of terminals connected to DL lines.

You invoke the SETUP utility with the command:

RUN $SETUP

The utility displays the question:

Function <EXIT>?

Respond with:

TERMINAL

The TERMINAL function dialog proceeds as follows:

1. System name for source of information?

   Respond with the name of the system whose terminals you wish to modify in the disk file.
2. Terminal command <EXIT>?

   The TERMINAL commands are:

   A. ADD

      The ADD command creates the terminal characteristics for each terminal that you specify in the dialog. Refer to Section 9.4.1 for adding terminal specifications.

   B. DELETE

      The DELETE command removes the terminal characteristics from the file for the terminal that you specify in the dialog. Refer to Section 9.4.2 for deleting terminal characteristics.

   C. DISPLAY

      The DISPLAY command displays the terminal characteristics for the terminal that you specify in the dialog. Refer to Section 9.4.3 for displaying terminal characteristics.

   D. EDIT

      The EDIT command changes the terminal characteristics for the terminal that you specify in the dialog. Refer to Section 9.4.4 for editing terminal specifications.

   E. COPY

      The COPY command copies the terminal characteristics of one terminal to other terminals. The terminals to be duplicated can be new or existing terminals. Refer to Section 9.4.5 for copying terminal specifications.

   F. STEAL

      The STEAL command copies an existing support terminal characteristics file from another system image into the current system. Refer to Section 9.4.6 for stealing terminal specifications.

   G. EXIT

      The EXIT command incorporates the changes to the support terminal characteristics file and then terminates the TERMINAL function. Control returns to the SETUP utility question:

      Function <EXIT>?

The remainder of the dialog depends upon the TERMINAL command that you select. Refer to the appropriate section.

9.4.1 Adding Terminal Specifications

The ADD command is useful for specifying a unique set of terminal characteristics. The ADD command is primarily for system generation; however, you can use the ADD command:

1. After a system generation
2. If the characteristics of a terminal were inadvertently not created
3. If you steal a set of characteristics (that is, the STEAL command) and wish to add more terminal characteristics.

The ADD command dialog proceeds as follows:

1. System name for source of information?

   Respond with the name of the system whose terminals you wish to create in the disk file.
2. Terminal command <EXIT>? ADD
3. Terminal number?

   Respond with an octal terminal number. The TERMINAL function attaches the terminal name, TT, and the colon (:) to the number that you specify. The ADD command accepts a terminal number from 1 to the lower of 20 or the maximum specified at system generation.
4. Speed <2400>?

   Respond with the terminal transmit/receive speed. Valid responses are any one of the following values: 0, 110, 150, 300, 600, 1200, 2400, 4800, and 9600. It is recommended that you limit the use of the 9600 speed.
5. VT52 <NO>?

   There are two kinds of terminals: video and hardcopy. If you respond with YES, the terminal is to be handled like a video terminal. Video terminals erase the rubbed-out character(s) while hardcopy terminals print them.
6. Width <default>?

   Respond with the character width that you want the terminal to display. You can specify from 40 to 132 decimal characters. Video terminals are a maximum of 80 characters wide, while hardcopy terminals are a maximum of 132 characters wide. The default value is either 80 characters if you specified YES to Question 5, or 132 if you specified NO to Question 5.
7. Lower case <YES>?

   Respond with YES if you want to type and to display upper- and lower-case characters. Respond with NO if you want upper-case characters only; lower-case is converted to upper-case.

8. Remote terminal <NO>?

   Respond with YES if the terminal is connected to a dial-up telephone line. Respond with NO if the terminal is not connected to a dial-up telephone line.
9. Slave terminal <NO>?

   Respond with YES to slave the terminal. The system accepts input from a slaved terminal only if the system requests such input. Respond with NO to permit the system to recognize unsolicited input for the terminal.

The ADD command dialog ends with Question 9. The TERMINAL function enters the changes to the support terminal specifications file and returns to Question 2 anticipating that you want to access the COPY command.

9.4.2 Deleting Terminal Characteristics

The DELETE command removes all terminal characteristics for a terminal. Use the ADD command to respecify the terminal characteristics.

The DELETE command dialog proceeds as follows:

1. System name for source of information?

   Respond with the name of the system whose terminal you wish to delete from the disk file.
2. Terminal command <EXIT>? DELETE
3. Terminal number?

   Respond with the octal terminal number whose characteristics you wish to delete from the file. The DELETE command accepts a terminal number from 1 to the lower of 20 or the maximum specified at system generation.

The DELETE command dialog ends with Question 3. The command removes the support terminal characteristics and returns to Question 3:

Terminal number?

to delete additional terminal characteristics. Respond with the ESCAPE key to terminate the DELETE command dialog and return to Question 2.

9.4.3 Displaying Terminal Characteristics

The DISPLAY command shows all terminal characteristics for a terminal.

The DISPLAY command dialog proceeds as follows:

1. System name for source of information?

   Respond with the name of the system whose terminal you wish to display in the disk file.
2. Terminal command <EXIT>? DISPLAY

3. Terminal number?

   Respond with the octal terminal number whose characteristics you wish to display from the file. The DISPLAY command accepts a terminal number from 1 to the lower of 20 or the maximum specified at system generation.

The DISPLAY command dialog ends with Question 3. The command displays the support terminal characteristics and returns to Question 3:

Terminal number?

to display additional terminal characteristics. Respond with the ESCAPE key to terminate the DISPLAY command dialog and return to Question 2.

9.4.4 Editing Terminal Specifications

Use the EDIT command to change the characteristics of a terminal.

The EDIT command dialog proceeds as follows:

1. System name for source of information?

   Respond with the name of the system whose terminals you wish to modify in the disk file.
2. Terminal command <EXIT>? EDIT
3. Terminal number?

   Respond with an octal terminal number. The TERMINAL function attaches the terminal name, TT, and the colon (:) to the number that you specify. The EDIT command accepts a terminal number from 1 to the lower of 20 or the maximum specified at system generation.
4. Speed <default>?

   Respond with the terminal transmit/receive speed. Valid responses are any one of the following values: 0, 110, 150, 300, 600, 1200, 2400, 4800, and 9600. The default value is the current value.
5. VT52 <NO>?

   There are two kinds of terminals: video or hardcopy. If you respond with YES, the terminal is to be handled like a video terminal, video terminals erase the rubbed-out character(s) while hardcopy terminals print them.
6. Width <default>?

   Respond with the character width that you want the terminal to display. You can specify from 40 to 132 decimal characters. Video terminals are a maximum of 80 characters wide, while hardcopy terminals are a maximum of 132 characters wide. The default value is either 80 characters if you specified YES to Question 5 or 132 if you specified NO to Question 5.

7. Lower case <YES>?

   Respond with YES if you want to type and to display upper- and lower-case characters. Respond with NO if you want upper-case characters only; lower-case is converted to upper-case.
8. Remote terminal <NO>?

   Respond with YES if the terminal is connected to a dial-up telephone line; respond with NO if the terminal is not connected to a dial-up telephone line.
9. Slave terminal <NO>?

   Respond with YES to slave the terminal. The system accepts input from a slaved terminal only if the system requests such input. Respond with NO to permit the system to recognize unsolicited input for the terminal.

The EDIT command dialog ends with Question 9. The TERMINAL function enters the changes to the support terminal specifications file and returns to Question 3.

Terminal number?

To terminate the EDIT command dialog, respond with the ESCAPE key to return to Question 2.

9.4.5 Copying Terminal Specifications

The COPY command duplicates the characteristics of one terminal to other terminal characteristics. If the other terminal characteristics are currently specified, the COPY command replaces the existing characteristics.

The COPY command dialog proceeds as follows:

1. System name for source of information?

   Respond with the name of the system whose terminals you wish to copy from the disk file.
2. Terminal command <EXIT>? COPY
3. From terminal number?

   Respond with the octal terminal number whose characteristics the COPY command uses as a template.
4. To terminal number?

   Respond with either an octal terminal number or ALL. If you specify ALL, the terminal function copies the template to all the support terminals of the same type (DZ or DL), except for TT0: .

The COPY command dialog ends with Question 4. The COPY command copies the template terminal characteristics and returns to either Question 3 or if you responded with ALL the dialog returns to Question 2. To terminate the COPY command dialog from Question 3, respond with the ESCAPE key to return to Question 2.

9.4.6 Stealing Terminal Specifications

The STEAL command permits you to copy an entire terminal file from another system. That is, if there is another system which you know contains at least approximately the proper terminal characteristics, it is more convenient to steal the complete file than to create a number of terminal specifications in a new file.

The STEAL command dialog proceeds as follows:

1. System name for source of information?

   Respond with the name of the system whose terminals you wish to copy from the disk file.
2. Terminal command <EXIT> ? STEAL
3. Name of the system from which the terminal file will be stolen?

   Respond with a file specification of the support terminal characteristics file to copy. You must specify a filename, although you may also specify a device and/or directory. If you omit the device and directory, the system uses a default value. The complete default file specification is SY:[200,201]system-name.TER. The STEAL command dialog ends with Question 3.

The STEAL command copies the latest version of the file that you specify to the SY:[200,201]system-name.TER file. The system-name is the filename associated with your response to Question 1. Upon completion, the STEAL command dialog returns to Question 2.

9.5 INCORPORATING SUPPORT TERMINAL CHARACTERISTICS (SETUP)

Once you have changed the support terminal characteristics file, run the REDO function of the SETUP utility to incorporate the changes into the system image file. Then, when the system is booted, it initializes the support terminals to the new characteristics.

You invoke the SETUP utility with the command:

```
RUN $SETUP
```

The utility displays the question:

```
Function <EXIT>?
```

Respond with:

```
REDO
```

The REDO function displays the question:

```
System name for source of information?
```

Respond with a system name. The REDO function incorporates the SY:[200,201] system-name.TER;1 support terminal characteristics file into the SY:[1,54] system-name.SYS;1 file. The terminal changes then become permanent. When you bootstrap the system, the support terminals are initialized with the new characteristics.

Upon completion, the system terminates the REDO function and returns to the SETUP utility question:

Function <EXIT>?

9.6 TRANSFERRING USER FILES AFTER SYSGEN (SETUP)

After you complete the system generation procedure and you are running an acceptable system image, use the TRANSFER function of the SETUP utility to copy all the files associated with nonsystem-controlled user file directories to the system disk.

You invoke the SETUP utility with the command:

RUN $SETUP

The SETUP utility displays the question:

Function <EXIT>?

Respond with:

TRANSFER

The TRANSFER function dialog begins with the question:

1. What is the device specification of the old disk?

   Respond with a disk device name indicating where you mounted the original disk volume. This volume is the one containing the files you want to copy.
2. Should the account file be transferred <YES>?

   A situation may arise when you perform another system generation where you want to continue to provide access to the system for the accounts you specify following a previous system generation. In this case, respond with YES to the question.

   If you respond with YES, the TRANSFER function copies to the new disk the system-controlled [200,201] TPSACNT.DAT file. This file is the only system-controlled file that TRANSFER can copy.

   If you respond with NO, the TRANSFER function does not copy the system-controlled [200,201] TPSACNT.DAT file.

The TRANSFER function first displays the system-controlled user file directories. Then the function displays each nonsystem-controlled user file directory and error messages describing complete files in that nonsystem-controlled UFD as the function begins to copy the directory.

9.7 CHECKING FOR BAD BLOCKS (SETUP)

The SETUP utility also provides for bad block checking of disk volumes. The BAD function verifies disks by determining the location of bad blocks and marking the location(s) on the disk. This information is then used by the INITIALIZE command to create a dummy file – [0,0]BADBLK.SYS;1 – utilizing the bad blocks. Thus, the bad blocks cannot be accessed for storing data.

You invoke the SETUP utility with the command:

```
RUN $SETUP
```

The SETUP utility displays the question:

```
Function <EXIT>?
```

Respond with:

```
BAD
```

The BAD function displays the question:

```
Specify the device on which bad block checking is to be done?
```

You respond with the device name in the form of:

```
ddnn:
```

Where:

1. dd identifies the type of physical device
2. nn identifies the specific device (for example, DB0:).

When the function is completed, the SETUP utility terminates the BAD function, displays error messages, and returns to the SETUP utility question:

```
Function <EXIT>?
```

The BAD function verifies disks by writing a test pattern into each of the blocks on the disk, reading the blocks back into a buffer in storage, and comparing the data in the buffer with the data that was written onto the disk.

The BAD function writes a test pattern into several blocks in a single write operation. If an error occurs in comparing any of these blocks, the BAD function tests each block individually. If no bad blocks are found during the individual tests, BAD does not report bad blocks on the disk.

If BAD locates a bad block, it records the address of the bad block in a buffer in storage. When all bad blocks on the disk are located, the address and length of the bad blocks are recorded on the last good block of the disk.

In general, use BAD with the INITIALIZE command to initialize a disk. The INITIALIZE command uses the bad block information to create a file called [0,0]BADBLK.SYS;1, also, BADBLK.SYS occupies those blocks found to be bad, thus ensuring that the file system does not write a file into a bad block.

The INITIALIZE command accesses the bad block descriptor by means of the /VERIFIED switch.

9.8 RECREATING A SYSTEM DISK IMAGE (SETUP)

You may find you cannot boot the system disk image because of either improper file deletions or you archived the disk and did not save a system image. Use the RENEW function of the SETUP utility to recreate the violated system image. The utility recreates the system disk image from the existing files in the [200,201] and [1,54] directories.

You invoke the SETUP utility with the command:

```
RUN $SETUP
```

The utility displays the question:

```
Function <EXIT>?
```

Respond with:

```
RENEW
```

The RENEW function responds with the following question:

```
System name for source of information?
```

Respond with the name of the system that you wish to recreate. Upon completion, the RENEW function returns to the SETUP utility question:

```
Function <EXIT>?
```

9.9 RECREATING INITIALIZATION COMMANDS FOR INIT (SETUP)

If the system initialization commands are destroyed or inadvertently deleted, use the STARTUP function to recreate the commands.

You invoke the SETUP utility with the command:

```
RUN $SETUP
```

The utility displays the question:

```
Function <EXIT>?
```

Respond with:

```
STARTUP
```

The STARTUP function displays the question:

```
System name for source of information?
```

Respond with a system name. The STARTUP function recreates the system initialization commands.

Upon completion, the system terminates the STARTUP function and returns to the SETUP utility question:

```
Function <EXIT>?
```

9.10 VERIFYING FILE STRUCTURES (VFY)

The File Structure Verification Utility (VFY), which must run in full support environment, provides the facilities to:

1. Check the readability and validity of file-structured volume
2. Print the number of available blocks on a file-structured volume (/FR)
3. Search for files that exist in the index file but which do not exist in any directory, that is, files which are lost in the sense that they cannot be accessed by filename (/LO)
4. List all files in the index file, showing the file ID, filename, and owner (/LI)
5. Mark as used all the blocks that appear to be available but are actually allocated to a file (/UP)
6. Rebuild the storage allocation bit map so that it properly reflects the information in the index file (/RE)
7. Restore files that are marked for delete (/DE)
8. Perform a read check on every allocated block on a file-structured volume (/RC)

There should be no other activity on the volume while VFY is executing a function. In particular, activities which create new files, extend existing files, or delete files should not be attempted.

9.10.1 Invoking VFY

Either a privileged or nonprivileged user can run the VFY utility. Nonprivileged user access is permitted when using VFY for read-only functions. Only a privileged user, under a system UIC, can access VFY for read/write functions. The utility normally operates in a read-only mode, assuming that the scratch file, if required, is on another device. VFY requires write access under the following conditions:

1. If the /UP or /RE switch is used, VFY requires write access to the storage allocation map ([0,0] BITMAP.SYS).
2. If the /DE switch is specified, VFY requires write access to the index file ([0,0] INDEXF.SYS). That is, mount the disk volume with the /UNLOCK command qualifier.
3. If the /LO switch is specified and lost files are found, VFY requires write access to the [1,3] user file directory.

You invoke VFY with the following dialog:

```
RUN $VFY
VFY>
```

NOTE

Do not abort VFY while the /UP, /RE, or /DE switch is being processed. Aborting VFY may seriously endanger the integrity of the volume.

All VFY functions are invoked by entering a VFY command line through the initiating terminal. The VFY command line is:

```
[listfile],[scratchdev:]=[indev:] [/switch]
```

Where:

listfile — The listfile parameter specifies the output-listing file in the following format:

```
[dev:] [ufd] [filename,filetype] [;ver]
```

The listfile defaults appear as:

1. dev — TI:

   If you direct the output for VFY to a terminal device and you do not wish to see all the error messages for a given file, enter CTRL/O to terminate the listing of all further error messages for that file. The system always displays the summary error messages for the file and continues to display the detailed error messages of other files.
2. [ufd] — The default UIC is the UIC under which VFY is currently running.

3. filename — You must specify a filename if you specify dev as either a disk or magnetic tape.
4. filetype — You must specify a filetype if you specify dev as either a disk or magnetic tape.
5. ver — The default version is the latest version plus one.

scratchdev — The scratchdev parameter specifies the device on which VFY produces a scratch file.

VFY uses the scratch file during the verification scan and the lost file scan. If you do not specify a scratch device, the system assigns the SY0: system device. The VFY utility creates a scratch file but does not enter the filename into a directory. The VFY utility deletes the scratch file upon termination of the VFY program.

If the system disk is faulty, use this parameter to force the scratch file to another device. The VFY utility does not use the scratch file for the /FR and /LI switches.

indev — The indev parameter specifies the volume to be verified.

If you do not specify the parameter, the system assigns the SY0: system device.

/switch — The switch parameter specifies the function to be performed.

If you do not specify a switch, the VFY utility performs a validity check.

9.10.2 Using VFY Switches

Specify a VFY function in the form of a switch appended to the VFY command line. Table 9-1 summarizes the command switches.

Table 9-1 VFY Functions and Switches

switch	function	Purpose
Default	Validity Check	Check readability and validity of the volume mounted on the specified device.
/DE	Delete	Reset the marked-for-delete indicators.
/FR	Free	Print out the available space on a volume.
/LI	List	List the entire index file by the file identifications.
/LO	Lost	Scan the entire file structure looking for files which are not in any directory.

Table 9-1 (Cont.) VFY Functions and Switches

switch	function	Purpose
/RC[:n]	Read Check	Check the entire volume to see if every block of every file can be read. The optional parameter [:n] is the blocking factor which indicates the number of file blocks to be read at one time. The default value is the maximum number of blocks in dynamic memory available to VFY.
/RE	Rebuild	Recover blocks which appear to be allocated but are not contained in a file.
/UP	Update	Allocate blocks which appear to be available but have been allocated to a file.

9.10.2.1 Validity Check – If you do not specify a switch, a validity check verifies the readability and validity of the volume mounted on the specified device. This feature entails reading all the file headers in the index file and ensuring that all the disk blocks referenced in the map area of each file header are marked as allocated in the bit map (that is, allocated to that file).

The rules for running the validity check are that the volume:

1. Must be mounted
2. May be write-protected if either:
   A. It is not the system volume
   B. The required scratch file is directed to another volume.

The VFY utility spools a listing of the results to the line printer when the validity check is completed.

After VFY verifies the volume and outputs the normal messages, VFY reports error conditions. The VFY utility precedes all errors for a given file with a file identification line, identifying the file in error. This line is formatted as follows:

FILE ID nn,mm filename.filetype;version OWNER [g,m]

Where:

nn,mm	The nn,mm is the unique file identification number assigned to the file when the system created the file.
filename	The filename is the name of the file.

,filetype	The filetype is the type of file such as OBJ for an object file.
version	The ;version is the version number of the file.
[ggg,mmm]	The [ggg,mmm] is the file owner UIC.

The file identification line is followed by one or more of the following messages:

I/O ERROR READING FILE HEADER-ERROR CODE-32

Failed to read the file header for the specified file ID.

BAD FILE HEADER

Software checks on the validity of the file header indicate that the header is corrupted.

MULTIPLE ALLOCATION n,m

The specified (double precision) logical block number is allocated to more than one file. If this error occurs, the system makes a second pass to determine which of the files share a multiple-allocated block. The second pass is taken after all file headers are checked. Using this information, you can determine which, if any, of the files can be saved and delete the remaining files with the DELETE command. After the files are deleted, run the VFY utility again to ensure that all the multiple-allocated files are deleted. These multiple blocks must be deleted before any lost or erroneously free blocks can be recovered. Blocks that appear to be allocated but that are not contained in any file are defined as lost blocks.

BLOCK IS MARKED FREE n,m

The specified logical block number is allocated to the indicated file but is not marked as allocated in the storage allocation map. To correct the error, run VFY with the /UP switch. If the VFY utility displays the BLOCK IS MARKED FREE n,m error with the MULTIPLE ALLOCATION error messages, you must first remove the multiple-allocated blocks.

BAD BLOCK NUMBER n,m

The specified block number was found in the header for the file, but the block number is out of range for the device. The BAD BLOCK NUMBER n,m error message indicates a corrupted file header.

FILE IS MARKED FOR DELETE

A system failure occurred while the specified file was being deleted. The deletion was not completed and the file header still exists.

HEADER MAP OUT OF SYNC

The error message indicates a corrupted file header.

The VFY utility follows the last error message for the file by a summary line for the file. The summary line format appears as follows:

SUMMARY: MULT=nn, FREE=nn, BAD=nn,

Where:

MULT	The MULT value is the number of multiple block allocations
FREE	The FREE value is the number of blocks marked free that should have been allocated
BAD	The BAD value is the number of bad retrieval pointers in the file header.

To determine whether any blocks are lost, examine the last two lines of output from the validity check, which indicate the free space on the volume. The first of these two lines shows the space available according to the index file (that is, the number of blocks not in use as specified by the index file). The last line shows the space available according to the storage allocation map. Assuming there are no other errors, these two figures should agree. If the index file indicates that more blocks are free than does the storage allocation map, then those blocks are lost. The lost blocks appear to be allocated, but they are not used by any file. To recover lost blocks, run VFY with the /RE switch.

9.10.2.2 Delete (/DE) – If a file is marked for delete but the deletion process is not completed, you can either restore the file – if you still need it – or you can delete the file to recover the space that it occupied. This situation occurs only if the system crashes during file processing.

1. Restoring a File

   Run VFY specifying the /DE switch to reset the marked for delete indicators in file headers. Once the delete indicator is reset, run VFY specifying the /LO switch to scan the entire file structure.

   The deletion process may have processed partially and a portion at the end of the file may be missing. This condition can be detected by a directory listing obtained using the DIRECTORY/FULL command.

2. Deleting a File

   Files that are marked for delete can be deleted directly with the DELETE command, once they are identified with the validity check. The system issues the following error message since the file system denies the existence of files already marked-for-delete:

   FAILED TO MARK FILE FOR DELETE-NO SUCH FILE

   However, the file is completely deleted.

Once files have been restored or deleted, run VFY with the /RE switch to assure the consistency of the storage allocation map on the volume.

9.10.2.3 Free (/FR) – The free switch (/FR) displays the available space on a volume. The message appears as:

```
dev: HAS nnnn.    BLOCKS FREE, nnnn.
BLOCKS USED OUT OF nnnn.
```

9.10.2.4 List (/LI) – The list switch (/LI) lists the entire index on a file-by-file basis. The output for each file specifies the file number, file sequence number, filename, and owner. A typical index file listing appears as:

```
VFY>DK0:/LI
LISTING OF INDEX ON DK0:

FILE    ID 000001,000001 INDEXF.SYS;1     OWNER [1,1]
FILE    ID 000002,000002 BITMAP.SYS;1     OWNER [1,1]
FILE    ID 000003,000003 BADBLK.SYS;1     OWNER [1,1]
FILE    ID 000004,000004 000000.DIR;1     OWNER [1,1]
FILE    ID 000005,000005 CORIMG.SYS;1     OWNER [1,1]
FILE    ID 000006,000006 001001.DIR;1     OWNER [1,1]
FILE    ID 000007,000007 001002.DIR;1     OWNER [1,2]
FILE    ID 000010,000010 EXEMC.MLB;1      OWNER [1,1]
FILE    ID 000011,000011 RSXMAC.SML;1     OWNER [1,1]
FILE    ID 000012,000012 NODES.TBL;1      OWNER [1,1]
FILE    ID 000013,000036 QIOSYM.MSG;311   OWNER [1,2]
FILE    ID 000014,000037 F4PCOM.MSG;1     OWNER [1,2]
```

9.10.2.5 Lost (/LO) – The lost switch (/LO) scans the entire file structure looking for files which are not in any directory, and thus are lost in that they cannot be referenced by filename. A list of the files is produced, and if the lost file directory [1,3] exists on the volume, all the files are entered in the directory.

9.10.2.6 Read Check (/RC) – The read check switch (/RC) checks to ensure that every block of every file on a specified volume can be read. Since read check is a read-only operation, the volume can be write-protected when you mount the volume.

For the fastest possible read check, the maximum block factor should be used when you mount the volume. The maximum block factor is specified by the optional parameter [:n]. The default for maximum block factor is five. If an error is encountered, each block of the portion in error is reread individually to determine which data block or blocks cannot be read. If an error is detected, a file identification line is listed in the following format:

FILE ID nn,nn filename.typ;ver. blocks used/blocks allocated

Following this line, an error message is listed. If a blocking factor other than 1 is in use, an error message in the following form is issued:

ERROR STARTING AT VBN n1,n2 LBN n1,n2 - ERROR CODE -err

Following the first error message, there should be one or more error messages indicating the exact block(s) in error. The system displays the second error message in the following form:

ERROR AT VBN n1,n2 LBN n1,n2 - ERROR CODE -err

If an ERROR STARTING AT line is displayed without one or more ERROR AT lines, a multi-block read operation on the selected device has failed, but the data blocks appear to be individually readable.

If the VBN of the unreadable block listed in the ERROR AT line is beyond the block-used-count, the data portion of the file is all right.

The negative number printed after the ERROR CODE message is -4 to indicate a device parity error. Other error codes are contained in Appendix A in this guide.

9.10.2.7 Rebuild (/RE) – The rebuild switch (/RE) recovers lost blocks. Multiple-allocated blocks must be removed from the file structure before the rebuild can be run. These blocks are removed by deleting the files which access the blocks.

The scratch file should be on another volume. If this is impossible, the volume must be dismounted immediately after VFY terminates. (Failure to do this may result in partial updating of the storage allocation map.) Then the volume should be mounted again, and the scratch file deleted by using the delete command. VFY issues a detailed message in this case, specifying the name of the scratch file to be deleted.

9.10.2.8 Update (/UP) – The update switch (/UP) allocates all blocks that appear to be available but are actually allocated to a file. Files with multiple-allocated blocks must be deleted from the file structure before the update can be run.

The scratch file should be on another volume. If this is impossible, the volume must be dismounted immediately after VFY terminates. (Failure to do this may result in partial updating of the storage allocation map.) Then the volume should be mounted again, and the scratch file must be deleted manually. VFY issues a detailed message, in this case specifying the name of the scratch file to be deleted. The message is:

VFY -- TO COMPLETE THE STORAGE MAP UPDATE DISMOUNT THE VOLUME IMMEDIATELY. THEN MOUNT IT AND DELETE THE FOLLOWING FILE:

[ufd] filespec

where:

[ufd] is the UFD.

filespec is the name of the file to be deleted.

9.10.3 Displaying Error Messages

The VFY utility displays the error messages presented in this section if you respond to the VFY prompt with an incorrect command line.

VFY -- COMMAND SYNTAX ERROR

Description: The command you entered does not conform to the command syntax rules given in Section 9.10.1.

Suggested User Response: Re-enter the command line with the correct syntax rules.

VFY -- FAILED TO ALLOCATE SPACE FOR TEMP FILE

Description: The volume you specified for the temporary scratch file is full.

Suggested User Response: Use the DELETE command to remove the unnecessary files and then return to the VFY utility.

VFY – FAILED TO ATTACH DEVICE

or

VFY – FAILED TO DETACH DEVICE

or

VFY – ILLEGAL DEVICE

Description: Your file specification contains an illegal device name.

Suggested User Response: Re-enter the command line with the correct device specified. Valid device names are listed in Chapter 8.

VFY -- ILLEGAL SWITCH

Description: The switch you specified is not valid or you used a switch illegally.

Suggested User Response: Re-enter the command line with the correct switch. Valid switches are listed in Section 9.10.2.

VFY – I/O ERROR ON INPUT FILE

or

VFY – I/O ERROR ON OUTPUT FILE

Description: You specified a device that:

1. Is off-line
2. Is dismounted
3. Has a hardware failure.

Suggested User Response: Determine which of the conditions caused the error message. Correct the error condition and re-enter the command line.

VFY – NO DYNAMIC MEMORY AVAILABLE - PARTITION TOO SMALL

Description: The GEN partition is too small to run the VFY utility.

Suggested User Response: Run VFY in at least an 8k GEN partition.

VFY – OPEN FAILURE ON BIT MAP

or

VFY – OPEN FAILURE ON INDEX FILE

or

VFY -- OPEN FAILURE ON LISTING FILE

or

VFY – OPEN FAILURE ON TEMPORARY FILE

Description: You violated one of the following conditions:

1. You specified a volume which is not mounted.
2. You specified a nonexistent directory.
3. You specified a nonexistent file.
4. You used a file specification whose volume, directory, or file protection access rights prohibit your access.

Suggested User Response: Determine which of the conditions caused the error message. Correct the error condition and re-enter the command line.

If VFY cannot access the error message file, VFY reports errors using the following format:

VFY -- ERROR CODE nn.

where:

nn The value of nn is one of the error codes listed in Table 9-2. Refer to the error message description to determine the corrective action.

Table 9-2 VFY Error Codes

Error Codes	VFY Error Message
1	ILLEGAL DEVICE
2	OPEN FAILURE ON BIT MAP
3	OPEN FAILURE ON TEMPORARY FILE
4	FAILED TO ALLOCATE SPACE FOR TEMP FILE
5	FAILED TO DETACH DEVICE
6	FAILED TO ATTACH DEVICE
7	COMMAND SYNTAX ERROR
8	I/O ERROR ON INPUT FILE
9	I/O ERROR ON OUTPUT FILE
10	ILLEGAL SWITCH
11	OPEN FAILURE ON LISTING FILE
12	OPEN FAILURE ON INDEX FILE
13	NO DYNAMIC MEMORY AVAILABLE – PARTITION TOO SMALL

CHAPTER 10
FORMAT CONVENTIONS

10.1 COMMAND DESCRIPTIONS

Command descriptions include the following information, as required:

1. The name of the command.
2. A brief statement of the command's purpose or action.
3. The general format of the command, showing all elements of the command. See Section 10.2 for information on command format.
4. The possible prompts that the command can issue to request additional information, such as file specifications or tasknames omitted from the command.
5. Descriptions of the command parameters, such as file specifications.
6. Descriptions of the qualifiers for the command. Qualifiers are key words whose first character is a slash (/). Generally, there are two categories of qualifiers: command qualifiers and parameter qualifiers.

   Command qualifiers are those which appear immediately after the command name, before any parameters that the command may contain. They influence the overall command action. Many commands allow or require multiple qualifiers.

   Parameter qualifiers influence only the parameter whose specification they immediately follow.
7. Whatever additional notes may be needed to describe the syntax and action of the command.

10.2 GENERAL FORMAT NOTATIONS

In describing general command formats, notation conventions are as follows:

1. Command names are shown in upper-case letters. Batch command names also have a leading dollar sign ($).
2. Parameters are shown in lower-case letters, and specify the type of information that you must provide. Parameters used in the descriptions are as follows:

Parameter	Information Required
file-spec	File specification.
in-file-spec out-file-spec	The parameters in-file-spec and out-file-spec are used only when needed to indicate the order of the parameters (as in a COPY command).

task-name	The name of an executing task.
function	A secondary keyword required in some commands to further define the action of the command.
device-name	A device designation; this can be either physical device name or a logical device name.
logical-name	A device designation; this must be a logical device name.
ufd	A User File Directory.
user-id	A User Identification Code or User Name.

The above list is not complete. Descriptions of some specialized parameters are included in the description of the command in which they appear or are self-explanatory.

3. Qualifiers are depicted in two forms:
   - A slash followed by a string of upper-case letters, indicating the actual characteristics of the qualifiers.
   - A slash followed by the lower-case word "qualifier," indicating that any of several qualifiers are valid.
4. Colons, periods, commas, semicolons, dollar signs, and slashes are part of the elements in which they appear, and are required where shown.
5. Square brackets [ ] indicate that the material enclosed within is optional.
6. Ellipses ... indicate that the immediately preceding parameter is repeatable; i.e., multiple values are allowed for the parameter. Separate multiple values by commas.

10.3 ISSUING COMMANDS

You communicate with the system by issuing commands. A command consists of a command name which describes the action the system is to take (COPY or LOGIN, for example), often accompanied by one or more parameters. Parameters either describe the items on which the command is to act or further define the function of the command.

Both command names and parameters can have qualifiers. Qualifiers are appended as a suffix to modify or further define the command name or parameters.

Commands can be entered at an interactive terminal only when the system is prompting >. Some commands (EDIT and BASIC, for example) invoke a program that accepts its own set of commands, valid only while that program is running. In turn, system commands are not valid while that program is running; you must first return control to the system. The descriptions of EDIT and BASIC in Part II describe how to terminate the invoked program's execution.

10.3.1 Command Structure

A command consists of a command name followed by a set of parameters. (A batch command also contains a dollar sign and an optional label.) The command name specifies the type of action for the system to perform, and the parameters indicate those entities on which the command will perform the action.

A command name or parameter can include a set of qualifiers that modify or complete its meaning. Qualifiers are appended directly to the command name or parameter with no embedded spaces.

The general format of a command is as follows:

```
[$[label:]] command-name [/qualifier...] parameter [/qualifier...]...
```

You can either supply the command name followed by the parameters on one line or enter the parameters in response to prompts. In both batch and interactive mode, when two or more parameters are on one line, they must be separated by at least one space or tab.

A command may require more than one line A hyphen (-) as the last character on the line continues the command onto the next line. Following the carriage return, the system prompts:

```
DCL>
```

An exclamation mark (!) line indicates the start of a comment. The comment text appears after the exclamation mark, and the rest of the line is treated as commentary.

10.3.2 Command Names

Every command begins with a command name that describes the action the command is to perform. You need not enter the complete command name to have it function correctly, only enough of its leading characters to uniquely identify it. No command name requires more than four characters for unique identification, and in some cases one character suffices.

For example, DEASSIGN and DEALLOCATE can be abbreviated to DEAS and DEAL respectively, but further abbreviation would make the command ambiguous. However, TYPE can be abbreviated to T, because no other command begins with that letter.

NOTE

To ensure compatibility with further versions of TRAX (which may include new commands), you should use at least four characters to identify commands in batch and indirect command files.

10.3.3 Parameters

A parameter either describes a value that a command uses when executing, or it further defines the action of the command. At least one space or tab must separate the first parameter from the command name; parameters are then separated from each other by one or more spaces (and/or tabs).

If you do not enter all the parameters that the system requires to execute the command, the system prompts until all required parameters are entered.

As an example, the COPY command is used to make new copies of files. It requires the name of the file to be copied and the name of the new file to be created. If only the command name, COPY, is entered, the system prompts for the name of the existing file by typing FROM?. It then prompts for the name of the file to be created by prompting TO?.

The COPY command can be entered in any of the following ways.

```
>COPY OLDFILE.DAT NEWFILE.DAT

>COPY OLDFILE.DAT
TO? NEWFILE.DAT

>COPY
FROM? OLDFILE.DAT
TO? NEWFILE.DAT
```

10.3.3.1 **Optional Parameters** – TRAX DCL never prompts for optional parameters. You must supply optional parameters on the same line with required parameters. This means that you must respond to each parameter prompt appearing on your terminal. For example:

```
>SHOW
FUNCTION? QUEUE ALL
  PRINT QUEUES
    PRINT
    LPQO
    TEST
    BAPO
  BATCH QUEUES
    BATCH
    TRXKIT
    SURVEY
    CHRIS
```

10.3.3.2 **Parameter Lists** – Some commands allow a parameter to be replaced by a list of parameters. For example, in a DELETE command, a single file specification can be replaced by a list of file specifications. When a parameter is a list of items, the items are separated by commas. Extra spaces are ignored.

Example:

```
>DELETE ABC.CBL;*,    AB.OBJ;1
```

10.3.4 Qualifiers

A qualifier consists of a character string, recognizable by the system, with a slash (/) as its first character. Its purpose is to modify or further specify the meaning of a command name or a parameter.

Qualifiers are directly appended to the qualified element, so that an element and all of its qualifiers form a string with no embedded spaces. Any qualifier may be abbreviated if it contains enough characters to distinguish it from any other possible qualifiers for that command. Any qualifier can be uniquely abbreviated to four leading characters.

10.3.4.1 Command Qualifiers – Command qualifiers modify the action of a command. For example, consider the following COBOL command string:

```
>COBOL/NOOBJECT COBSRC.CBL;1
```

Normally (i.e., that is, by default) the COBOL command produces an object file. In the example string, the /NOOBJECT qualifier overrides the default action, and no object file is produced.

Some commands have no qualifiers, while others have many possible qualifiers. Multiple qualifiers are permitted. For example:

```
>COBOL/NOOBJECT/LIST:COBLST.TMP  COBSRC.CBL;1
```

Some command qualifiers include variables; the /LIST qualifier, for example, can specify a file to receive a listing. There is no prompting for qualifiers or for qualifier variables.

10.3.4.2 Parameter Qualifiers – In some commands, parameters such as file specifications can have qualifiers. Parameter qualifiers further specify parameters that require special treatment. In the following APPEND command, the /RELATIVE qualifier informs the system that the input file DAT.TMP;1 has relative file organization.

```
>APPEND DATA.TMP;1/RELATIVE  UPDATE.DAT
```

10.3.5 Underline Convention

To improve readability, some DCL words include an underline character join ing two or more English words. For example:

```
CREATE/VOLUME-LABEL        SEVERE-ERROR
```

When such DCL words are abbreviated the underline is treated the same as an alphabetic character. Thus the following are all valid abbreviations for the qualifier VOLUME-LABEL:

VOLUME-L
VOLUME-
VOLU

The following abbreviations are not valid:

VOLUME-LBL (interior characters omitted)
VOLUME-LAB (hyphen not valid)

10.4 TERMINAL KEYBOARD FUNCTIONS

You type the input text one line at a time, terminating each line with a carriage return (RETURN). The system either prints the terminal input on the terminal printer or displays it on the screen of a display unit.

Function keys can be used to format a line (Space Bar, TAB), to edit a line (DELETE), or to access the uppermost of two characters that appear on a key (SHIFT, SHIFT LOCK). Typing a carriage return (RETURN) causes the system to process the current line.

Table 10-1 describes the function keys, as they appear on the LA36 and VT52 support terminals, and the effects of their use.

The CTRL key produces a variety of functions when pressed simultaneously with any one of several letter keys.

The combination of CTRL and another character key is called a control character. In this manual a control character is written as "CTRL/x" where x represents a variable character key.

The effect of a control character sometimes depends on the activity that the terminal is currently supporting.

Table 10-2 lists all the control characters supported under TRAX and their associated functions.

10.5 CORRECTING INPUT ERRORS

Before terminating a line, you can correct typing errors by using the DELETE key or change the line completely by using CTRL/U. However, once a command has been terminated (and thus input to the system) it cannot be corrected; the system will perform, or attempt to perform, the operation you specified. File information can be corrected by editing.

10.5.1 Deleting Individual Characters

The DELETE key deletes the most recent character on the current line for each pressing of the key. DELETE has no effect when the current line is empty.

Table 10-1 Keyboard Functions

Key	Description
CTRL	Used in combination with several 1-key letter keys to produce a variety of functions.
DELETE	Deletes the last character typed at the terminal, and further continuous characters if the key is pressed repeatedly.
ESC	Terminates a line of input without moving the carriage or cursor.
LINE FEED	Physically moves the paper or rolls the screen image upward, without influencing the system in any way.
RETURN	Terminates the current input line and enters the line into the system; the carriage or cursor advances to the first character position of the next line.
SHIFT	Prints or displays the uppermost of two characters appearing on a key typed while SHFFT is held down.
SHIFT LOCK or CAPS LOCK	Alternately locks and unlocks SHIFT mode on alphabetic characters. This key does not affect nonalphabetic characters.
SPACE BAR	Advances carriage or cursor one space at a time.
TAB	Causes the carriage or cursor to move to the next tab stop on the line. A line conventionally contains tab stops every eight character positions.

Table 10-2 Control Key Functions

Key	Description
CTRL/C	CTRL/C typed either as the first character in the line or when the terminal is producing output causes the system to prompt for command input. If the last character entered at the terminal was CTRL/S, CTRL/C also performs the function of CTRL/Q.
CTRL/I or TAB	Advances the carriage or cursor to the next horizontal tab stop on the line. The system establishes tab stops at every eight characters in the line.
CTRL/K	Causes a vertical tab by performing four line feeds.
CTRL/L	Performs eight line feeds.
CTRL/O	Alternately suppresses and resumes the display of output at the terminal. The system discards characters directed to a terminal that has disabled the display of its output.
CTRL/Q	CTRL/Q typed after a CTRL/S resumes output suspended by the previous CTRL/S.
CTRL/R	Typing CTRL/R before typing a line terminator causes the system to retype the current line on a new line, omitting any depleted characters. If the current line is empty, CTRL/R performs a carriage return and line feed.
CTRL/S	Typing CTRL/S while the terminal is receiving output suspends additional output until you type CTRL/Q or CTRL/C. The suspended output is merely delayed, not lost (see the description of CTRL/Q). The combined functions of CTRL/Q and CTRL/S are convenient when using a display terminal that transmits faster than you desire.
CTRL/U	Typing CTRL/U before typing a line terminator causes the previously typed characters to be deleted back to the beginning of the line. The system responds with a carriage return and line-feed so that the line can be re-typed. CTRL/U is echoed as CU.
CTRL/Z	Is a break character indicating end-of-file. Use it when the system is expecting input as a signal to indicate that you are finished typing in data. Most system utilities will bring all processing to an orderly termination and exit in response to this function.

On a hard-copy terminal, each deleted character is echoed. The string of deleted characters is enclosed between an initial and a final backslash (\). These backslashes are generated by the system for visual reference, and are not included in the data. The final backslash is printed when a new text character is typed in place of DELETE. Backslashes and deleted characters are omitted in the case when CTRL/R is used to make a copy of the line as typed so far.

On a video terminal, such as a VT52, each deleted character is removed from the screen, and the cursor returns to where it was before the character was typed.

For example, to change ACCDE to ABCDE, the user presses DELETE four times to override the CCDE. On a hard-copy terminal the string now appears as

```
ACCDE\EDCC
```

The user then enters the correct sequence BCDE. On the hard-copy terminal, the string now appears as

```
ACCDE\EDCC\BCDE
```

On a display unit the screen will show the string

```
ABCDE
```

In both cases ABCDE is the string accepted and sent to the computer when the line is terminated.

10.5.2 Deleting a Line

CTRL/U deletes all characters on the line, prints CU, and performs a carriage return. The user can then enter the text correctly.

For example, if you type ACCDEFGHI, but meant to type B for the first C, pressing the DELETE key eight times would be tedious and the result confusing on a hard-copy terminal. It would be easier to press CTRL/U and start again. The latter method would appear as follows:

```
ACCDEFGHI   CU
ABCDEFGHI
```

After using the DELETE key to correct a line and before terminating the line, you can ensure that the final result is what you want by typing CTRL/R before pressing carriage return. This displays the connected line contents before it becomes computer input.

Further corrections can be made at this point if necessary.

10.6 ABBREVIATIONS

When you type a system keyword (such as a command, or qualifier, or fixed parameter value), you need only type enough leading letters to uniquely identify it. However, the characters that you do include must match those of the corresponding characters in the full name. The system does not attempt to resolve invalid names by dropping characters from the right-hand end. If, for example, you are typing a COBOL command:

COBOL	is a complete command name.
COB	is a valid abbreviation of COBOL.
CO	is ambiguous, because COPY begins with the same letters.
CBL	is invalid, because no command begins with the letters CBL.

Any keyword can be abbreviated to the first four letters and be recognizable. Some keywords can be abbreviated to a single letter. Nevertheless, when creating batch procedures or indirect command files for long-term use, you should consider limiting abbreviations to four characters; this will ensure compatibility with future enhancements.

CHAPTER 11

ALPHABETICAL LISTING OF COMMANDS

11.1 ABORT

The ABORT command terminates the execution of a running task or command. Aborting a task forces the system to terminate the task or command. To effect the termination, the system:

- Perform I/O rundown
  I/O for all nonfile-structured devices are cancelled. I/O for file-structured devices are allowed to complete and then the files are deaccessed. All attached devices are detached.
- Displays the abort message.

Upon completion of the ABORT, the system displays the task name on the originating terminal.

A privileged user can abort any task in the system, while nonprivileged users can only abort tasks they execute.

Format:

ABORT [name]

Command Qualifiers:	Default:
/COMMAND	/COMMAND
/TASK	
/DUMP	

Prompt:

COMMAND NAME? name

Command Parameter:

name
The name of either the task or command to be aborted. If you do not provide a name and you use the /TASK qualifier, the system attempts to abort the task, TTnn, where nn is your terminal number. If you do not provide a name to abort a command, the system displays the COMMAND NAME? prompt. You must supply the name of the command that you wish to abort. See the command qualifiers for further information.

Command Qualifiers:

/COMMAND
Abort a DCL command such as DIRECTORY. The system assigns the task name xxxTnn to the command that you use where xxx are the first three characters of the command and nn is your terminal number.

/TASK
Abort a user task. Tasks started with the command, RUN file . . . , can be aborted by:

1. Specifying ABORT/TASK on the originating terminal
2. Specifying ABORT RUN on the originating terminal
3. Specifying ABORT/TASK name on any privileged terminal

The indirect command file task is aborted by:

1. Specifying ABORT/TASK AT. on the originating terminal
2. Specifying ABORT/TASK AT.Tnn on any privileged terminal.

   The value on nn is the terminal number where the task was entered.

See the SHOW TASKS command to display active task names.

/DUMP
Requests a post-mortem dump of either the aborted task or command.

Examples:

1. ABORT DIRECTORY

   TASK "DIRT03" TERMINATED
   ABORTED VIA DIRECTIVE OR MCR

   Terminates the currently active DIRECTORY command originating from terminal 3 (TT3:). Multiple command names are not permitted with the ABORT command.
2. RUN/TASK:TEST FILE.TSK
   .
   .
   .
   ABORT/TASK TEST

TASK "TEST12" TERMINATED
ABORTED VIA DIRECTIVE OR MCR

Aborts the currently active user task called TEST.

3. ABORT/TASK AT.

TASK "AT.12" TERMINATED
ABORTED VIA DIRECTIVE OR MCR

Aborts the indirect file processor task from the issuing terminal (TT2:).

11.2 ALLOCATE

The ALLOCATE command establishes a specified device as a private device and denies other nonprivileged users access to the device.

A nonprivileged user must allocate a device before mounting it. For efficient resource management, devices should be deallocated when they are no longer needed. (Refer to the DEALLOCATE command for its use.) Only privileged users and the device owner can deallocate a device. The system automatically DEALLOCATES private devices when the owner logs off (LOGOUT) the system.

Format:

ALLOCATE device-name

Command Qualifier:	Default:
TERMINAL:TTnn:	TERMINAL:TI0:

Prompt:

DEVICE? device-name

Command Parameter:

device-name
The device name of the device to be allocated.

Command Qualifier:

/TERMINAL:TTnn:
The system allocates the device name to the specified terminal. Only the tasks initiated from that terminal and privileged tasks can access the device.

Example:

1. ALLOCATE DB2:

   Allocate the DB2: disk to your terminal, T10:. The system and other nonprivileged users are not permitted to use DB2:.
2. ALLOCATE/TERM: TT22: DB2:

   The system allocates DB2: to terminal TT2:

11.3 APPEND

The APPEND command copies one or multiple sequential files, or the records of one relative or indexed file, to the end of an existing sequential file.

Format:

APPEND input-file-spec[/file-qualifier] output-file-spec

File Qualifiers:	*Default:*
/SEQUENTIAL	/SEQUENTIAL
/RELATIVE	
/INDEXED	
/KEY:NUMBER:n	

Prompts:

FROM? input-file-spec[/file=qualifier] . . .
TO? output-file-spec

Command Parameters:

input-file-spec, . . .
The file specification of the input file or files. If multiple file specifications are given, the entire set of specifications must be enclosed in parentheses.

output-file-spec
The file specification of a sequential file to which the records of the input files will be appended.

All file specifications must include a file name and a file type.

File Qualifiers:

/SEQUENTIAL
Specifies that the input file or files has sequential organization. This is the default.

/RELATIVE
Specifies that the input file has relative organization.

/INDEXED
Specifies that the input file is organized as an indexed file.

/KEY:NUMBER:n
Optionally specifies the record access key for an indexed file. If n-1, it calls for access on the primary key defined for the input file. If n=2, it specifies access on the first alternate key; n=3 specifies the second alternate key, and so on, up to the number of keys defined for the input file.

NOTES:

1. The output file must have sequential organization.
2. Wildcards are allowed on input files with sequential organization only. The order of appending multiple files specified by wildcard is their order of appearance in the directory and can be seen in advance by issuing a DIRECTORY command using the same wildcard specification.
3. If the input file is organized indexed or relative, the APPEND command must include the appropriate qualifier.

Example:

This example appends the contents of FILE 1.DAT to the end of FILE 2.DAT.

```
>TYPE FILE1.DAT
THIS IS FILE 1.
>TYPE FILE2.DAT
THIS IS FILE 2.
>APPEND FILE1.DAT
TO? FILE2.DAT
>TYPE FILE2.DAT
THIS IS FILE 2.
THIS IS FILE 1.
>
```

11.4 ARCHIVE

The ARCHIVE command performs volume or file backup and restoration. The system assumes file-archiving unless you specify the /VOLUME qualifier. The system performs file or volume backup unless you specify the /RESTORE qualifier.

11.4.1 Archiving Volumes

Volume-archiving copies disk files either to another disk or to magnetic tape. Before you specify the ARCHIVE command, you must discount the source disk volume and the destination volume. The system can reallocate and consolidate the disk area if you use the merge option.

If the destination volume is on a disk, the resultant copy is functionally identical to the source disk and can be immediately mounted and accessed. If the destination volume is on a magnetic tape, the resultant magnetic tape volumes – must be restored to a disk to be usable.

Formats:

ARCHIVE/VOLUME input_device-name:[volume-id] output_device-name:[volume-id]

ARCHIVE/RESTORE/VOLUME input_device-name:[volume-id] output_device-name:[volume-id]

Command Qualifiers:	Defaults:
/DENSITY:n	n = 800
/NOMERGE	/MERGE
/REWIND	
/SPLIT_DENSITY	

Prompts:

FROM? input_device-name:[volume-id]

TO? output_device-name:[volume-id]

Command Parameters:

input_device-name:[volume-id]
When backing up a volume, you can specify either magnetic tape or disk. when restoring, you must specify a magnetic tape.

NOTE
When you back-up a disk, the system creates an immediately usable backup disk.

Multiple input_device-names are specified in the format:

input_device-name:, . . . , input_device-name:

When specifying a magnetic tape, the volume-id is verified. When using multiple device names, specify the volume identification after the last device name.

output_device-name:[volume-id]
When backing up a volume you can specify either magnetic tape or disk. When restoring a volume, you must specify a disk. You specify multiple output_device-names in the same format as given for the input device.

When specifying a magnetic tape, the volume-id must be provided. The volume-id may be up to 6-alphanumeric characters.

Command Qualifiers:

/DENSITY:n
The magnetic tape, specified as the output_device-name, is set to either 800 or 1600 bits per inch (bpi).

/[NO]MERGE
The system copies the files exactly as they appear on the disk when you specify /NOMERGE.

When you specify /MERGE, the noncontiguous files and extensions are concatenated into contiguous data blocks. The disk-housekeeping, disk storage, and disk access time are reduced.

NOTE
You must renew any system images which you did not save prior to issuing the ARCHIVE command. Refer to the RENEW function of the SETUP utility for more information.

/REWIND
Rewind the magnetic tape(s) specified as either the input_device-name(s) or output_device-name(s) before commencing the desired operation.

/SPLIT_DENSITY
Except for the first two blocks of each volume, either read or write the TU16 magnetic tape volume(s) at a density of 1600 bpi. The first two blocks of each volume are 800 bpi.

Examples:

1. ARCHIVE/VOLUME/DENSITY:1600 DB0: MM0:,MM1:FRED

   Copy the DB0: device to the volume set on TU16 and call it FRED.
2. ARCHIVE/RESTORE/VOLUME/DENSITY:1600 MM0:,MM1: FRED DB0:

   Restore DB0: from the TU16 volume set called FRED.

11.4.2 Archiving Files

File-archiving permits you to back up a single disk file or a collection of disk files to either disk or magnetic tape. The system creates the files on the destination volume with the same filenames and filetypes. Unlike volume-archiving, the source and destination volumes must be mounted.

NOTE
You must restore the disk or magnetic tape backup file(s) to disk before you can access them.

Formats:

ARCHIVE input-file-spec, . . . output-file-spec

Command Qualifiers:	Default:
/LOG[:file-spec]	
/REPLACE	
/CREATION:dd-mm-yy	
/REVISION:dd-mm-yy	
/BEFORE	
/AFTER	
/REWIND	

File Qualifier:	Default:
None	

ARCHIVE/RESTORE input-file-spec, . . . output-file-spec

Command Qualifiers:	Default:
/LOG[:file-spec]	
/REPLACE	
/ARCHIVE:dd-mm-yy	
/OWNER:[ggg,mmm]	

Input-File Qualifier:	Default:
/SELECT:(file-spec-list)	

Prompt:

FROM? input-file-spec, . . .
TO? output-file-spec

Command Parameters:

input-file-spec, . . .
When backing up files, the file specification must include a disk device name. Multiple file specifications are separated by a comma and enclosed in parentheses.

output-file-spec
When restoring files, the file specification must include a disk device name. The filename and filetype must be specified as wild cards when the file specification includes a disk. When the file specification includes a magnetic tape, you must give a file specification and no wild cards are permitted. Multiple file specifications are separated by a comma and enclosed in parentheses.

Command Qualifiers:

/LOG[:file-spec]
The system produces a log file that contains a summary of what actually occurred during the execution of the command. When you do not specify a file specification, the system displays the information on the entering terminal.

/REPLACE
When the output file specification includes a disk, the system supersedes output files with input files when the filenames are identical.

When the output file specification does not include a disk, the system ignores the qualifier.

/CREATION:dd-mm-yy
The system backs up the disk files created either before, on, or after the specified date. The timing depends upon whether you also specify the /BEFORE or /AFTER qualifier. When you specify neither qualifier, the files on that date are affected. The date format:

dd-mm-yy

requires the day (dd), the month (mm), and the last 2 digits of the year (yy) to be separated by the hyphen. You must use the wild card attribute as the input filename and filetype.

/REVISION:dd-mm-yy
The system backs up the disk files accessed either before, on, or after the specified date. The timing depends upon whether you also specify the /BEFORE or /AFTER qualifier. When you specify neither qualifier, the files on that date are affected. The date format is the same as described for the /CREATION qualifier. You must use the wild card attribute as the input filename and filetype.

/BEFORE
You specify this qualifier to redefine the /CREATION or /REVISION date. The system backs up those files created or accessed before the specified date.

/AFTER
You specify this qualifier to redefine the /CREATION or /REVISION date. The system backs up those files created or accessed after the specified date.

/REWIND
Prior to preserving the file(s), rewind the magnetic tape specified in the output file specification.

/ARCHIVE:dd-mm-yy
Restore those files backed-up on the specified date. The date format is the same as given in the /CREATION qualifier. You must use the wild card attribute as the input filename and filetype.

/OWNER[:ggg,mmm]
Restore those files backed-up from the specified directory. Use the wild card attribute as the input filename and filetype to restore the latest version of those dated files.

Input-File Qualifier:

/SELECT:(file-spec-list)
Restore the listed files from a single magnetic tape file.

Examples:

1. ARCHIVE *.CBL DB1:*.*

   Archive all your COBOL source files from the system disk to the DB1: backup disk.

   NOTE
   The files are not usable on the DB1: disk; the files must be restored with the ARCHIVE/RESTORE command before the files can be used.

2. ARCHIVE/RESTORE DB1:*.CBL *.*

   Restore all your COBOL backup source files to the system disk.

11.5 ASSIGN

Use the ASSIGN command to define a logical name to a physical device name, to redirect all the I/O from one device to another device, and to assign a processor to a print or batch queue.

11.5.1 Assigning Logical Names

Use the ASSIGN command when you want to establish an I/O path between a logical name in a task you have written and a physical device. The assignments can be made on a login, local, and global basis.

Use the SHOW ASSIGNMENTS command to display the assignments. Refer to the DEASSIGN command to delete the assignments.

Format:

ASSIGN device-name: logical-name

Command Qualifiers:	Default:
/LOCAL	/LOCAL
/GLOBAL	
/LOGIN	
/TERMINAL:TTnn:	

Prompts:

DEVICE? device-name:
LOGICAL NAME? logical-name

Command Parameters:

device-name:
The device name of the physical device or previously assigned logical name to be assigned.

logical-name
A logical name uses the same syntax as a physical device name; it consists of two alphabetic characters and an 1- or 2- digit octal number, followed by a colon(:). The two alphabetic characters can either be equivalent to a standard device name such as DB: or it can be two letters picked at random such as XY:. You can use logical names that are identical to physical device names even though they do not refer to the same physical device. In such a case, the system uses the logical name and ignores the physical device name.

Command Qualifiers:

/LOCAL
Assign logical names to commands and tasks initiated from this terminal.

/LOGIN
Assign at login the logical names to commands and tasks.

/TERMINAL:TTnn:
Assign logical names to all commands and tasks initiated from the TTnn: terminal. The /TERMINAL qualifier can be used in conjunction with /LOGIN and /LOCAL qualifiers.

/GLOBAL
Assign logical names to all commands and tasks in the system.

Examples:

1. ASSIGN/LOCAL/TERMINAL:TT5: DB2: DB1:

   Locally assign to terminal TT5: the DB2: disk to logical name, DB1:. This command can be issued from any privileged terminal.
2. ASSIGN/GLOBAL LP1: LP0:

   Globally assign the LP1: line printer to LP0: line printer.

11.5.2 Redirecting I/O

Use the privileged /REDIRECT qualifier of the ASSIGN command to redirect all I/O requests from one physical device to another physical device. Currently queued I/O is not affected. An attached device, a device with a mounted volume, and TI0: cannot be redirected.

Format:

ASSIGN/REDIRECT old-device-name new-device-name

Command Qualifier:	Default:
/REDIRECT	

Prompts:

FROM? old-device-name

TO? new-device-name

Command Parameters:

old-device-name

The physical device name or logical name to be redirected. If you use the SHOW DEVICES command, the device names listed in the left-hand column are the devices that can be redirected.

new-device-name

The device name to receive the redirected I/O from the old-device-name. If you use the SHOW DEVICES command, the new-device-name is the device displayed to the right of the old-device-name.

Command Qualifiers:

/REDIRECT

You must use the /REDIRECT qualifier to change the I/O from one device to another device.

Example:

1. ASSIGN/REDIRECT TT1: TT2:

   Redirect all output for TT1: to TT2:
2. ASSIGN/REDIRECT CO0: TT3:

   .
   .
   .

   ASSIGN/REDIRECT CO0: TT0:

Temporarily redirect the console output to your terminal, TT3:, to observe any error messages, such as the line printer not ready, which the system directs to CO0:.

11.5.3 Assigning a Processor to a Queue

Use the privileged ASSIGN/QUEUE command to establish a path permitting jobs in the specified queue to be passed to the specified processor. The processor and queue must not be marked for deletion and they must be the same type (batch or print).

Format:

ASSIGN/QUEUE queue-name processor-name

Command Qualifiers:	Default:
/QUEUE	

Prompt:

QUEUE NAME? queue-name
PROCESSOR NAME? processor-name

Command Parameters:

queue-name
The print or batch queue to be assigned to the processor. A queue name may be from 1- to 6- characters and must be unique in the queue file.

processor-name
The print or batch processor to be assigned to the queue. You must specify the processor name using the following format:

ssPnn

Where:

1. The two letters, ss, are either LP for a print processor or BA for a batch processor
2. The two letters, nn, are either a 1- or 2- digit octal number.

   For a print processor, the value of nn corresponds to the physical device name of the printer, such as LPP0 for the LP0: line printer.

   For a batch processor, the value of nn is a unique octal number, such as BAP0 for the first batch processor.

Command Qualifiers:

/QUEUE
The required /QUEUE qualifier assigns a processor to a queue.

Example:

1. ASSIGN/QUEUE BATCH BAP0

   The batch processor, BAP0, is assigned to the batch queue, BATCH.

11.6 BASIC

The BASIC command invokes the BASIC-PLUS-2 compiler and places the terminal in BASIC-PLUS-2 control mode.

Format:

BASIC

NOTES:

1. The BASIC command has no qualifiers or parameters. After you enter the BASIC command, the following information appears:

   BASIC 2

   This indicates that you are in BASIC-PLUS-2 mode and must enter only commands appropriate to that mode when handling files.

2. To exit BASIC-PLUS-2 and return to TRAX, enter the following command in response to a BASIC 2 prompt.

   EXIT

3. See BASIC -PLUS-2 documentation for information about the BASIC-PLUS-2 programming language and control commands.

11.7 COBOL

The COBOL command compiles one or more COBOL source program files.

Format:

COBOL[/qualifiers] file-spec [, . . .]

Command Qualifiers:	Default:
/LIST[:file-spec]	/NOLIST
/NOLIST	
/OBJECT[:file-spec]	/OBJECT
/NOOBJECT	
/SWITCHES:(values)	

Prompt:

FILE? file-spec [, . . .]

Command Parameter:

file-spec, . . .
Specifies a COBOL source program to be compiled. If a file-spec does not include a file type. .CBL. is assumed.

Command Qualifiers:

/OBJECT[:object-file-spec]
/NOOBJECT
Specifies that an object file be produced and named as indicated by object object-file-spec. /OBJECT is the default qualifier. The default file name is the name of the first source file. The default file type is .OBJ. /NOOBJECT specifies that no object file is to be produced.

/LIST[:list-file-spec]
/NOLIST
/LIST specifies that the listing be produced and named according to list-file-spec. The default file name is the name of the source file. The default extension is .LST.

/NOLIST specifies that no listing file be produced. This is the listing default qualifier.

/SWITCHES: (/values)
Passes optional switches directly to the compiler in keyword form. The switch values are:

ERR:n	CREF
ACC:n	SYM:n
MAP	NORUN
LOD	RUN
CVF	HELP
USW:n. . .:m	TST

Each switch value must be preceded by a slash. For details regarding these switches, see the TRAX COBOL Language User's Guide.

Example:

This command compiles the source file COBPRG. CBL. The object file name defaults to COBPRG. OBJ, and the file OUT. LST contains the listing.

11.8 COPY

The COPY command performs any of these functions, depending on the qualifiers used.

1. Copies a single file such that the output file has the same organization as the input file.
2. Creates a sequential file consisting of a concatenated set of sequential files.
3. Copies a set of files to a corresponding set of files. This is called parallel copying.
4. Creates a sequential file copy consisting of the records from a single sequential, indexed, or relative organized file.

Format:

1. For single or parallel file copying:

COPY [/qualifiers] input-file-spec [file-qualifier] output-file-spec

Command Qualifiers:

/CONTIGUOUS
/BLOCKSIZE:n
/OWN

File Qualifiers:

/SEQUENTIAL
/RELATIVE
/INDEXED [/KEY:number:n]

2. For concatenating sequential files into one file:

COPY [/qualifiers] (input-file-spec [/SEQUENTIAL] [, . . .])
output-file-spec

Command Qualifiers:

/CONTIGUOUS
/BLOCKSIZE:n
/OWN

Prompts:

FROM? input-file-spec [/file-qualified] [. . . .]
TO? output-file-spec[

Command Parameters:

input-file-spec
The file specification of the input file. Each file specification must include a file name and a file type.

output-file-spec
The file specification of the output file.

Command Qualifiers:

/CONTIGUOUS
Specifies a contiguous output file. If the /CONTIGUOUS qualifier is omitted, the output file is not necessarily contiguous.

/BLOCKSIZE:n
Specifies a blocksize to be used when copying files to and from magnetic tape.

/OWN
Specifies that the owner of the output file will be the UFD under which it resides.

File Qualifiers:

/SEQUENTIAL
Specifies that the input file has sequential organization.

/RELATIVE
Specifies that the input file has relative organization.

/INDEXED
Specifies that the input file has indexed organization.

/KEY:NUMBER:n
Optionally specifies the record access key for an indexed file. If n=1, it calls for access on the primary key defined for the input file. If n=2, it specifies access on the first alternate key; n=3 specifies the second alternate key, and so on, up to the number of keys defined for the input file.

NOTES:

1. If copying disk-to-disk to obtain a sequential output file from an /INDEXED or /RELATIVE input file, you must indicate the organization of the input file by means of a file qualifier. If you omit the /RELATIVE or /INDEXED qualifier, the output file organization is the same as the input file. When copying files to magtape, you must specify the organization of the input file.

2. Wildcards are allowed for sequential input files when producing a single, sequential output file. When wildcards appear in the input-file-spec but not in the output-file-spec, the input files are concatenated in the output file. Order of copying depends solely on the order of their appearance in the directory.

3. Wildcards are allowed in the output-file-spec when the directories of the input-file-spec and the output-file-spec are different. Both the file name and file type components of the output file must be represented as wildcards. In this case, each input file is copied into a separate output file with identical file name and file type.

Example:

1. Copy the file RANDOM.DAT from the directory [350,230] into the current default directory. The copy of the file is unchanged.

```
>COPY [350,230]RANDOM.DAT *.*
```

2. Copy all files with the file type .CBL from the current directory to the directory [40, 41].

```
>COPY *.CBL [40,41]
```

   This operation requires write permission in directory [40, 41].

11.9 CREATE

Use the CREATE command to establish either a file or user file directory. When creating a file, you assign a filename and filetype. The system places the filename, filetype, and system-supplied file identifier in the directory which contains entries for each file. These entries point to the location of each file. When accessing the file, you supply the filename and filetype, which the system associates with the file identifier. The file identifier in turn points to the location of the file header which contains a list of the extent(s) that make up the file. Refer to Chapter 8 in this guide for more information.

11.9.1 Creating Files

The CREATE command creates an empty file. If the file has sequential organization, you may enter text into it as follows:

1. In interactive mode, you enter the formatted command and then type RETURN. On succeeding lines, type the data that you want to place in the file. Type CTRL/Z to indicate the end of the data.
2. In batch mode, the text to be placed in the file follows the command. Any batch command terminates the data file unless the CREATE command includes the qualifier /DOLLARS, in which case only the batch command $EOD can terminate the data.

Files specified as organized /RELATIVE or /INDEXED cannot accept data at the time of creation.

Format:

CREATE [/qualifiers] file-spec

Command Qualifiers:	Default:
/ALLOCATION:n	n-0
/BUCKETSIZE:m	m-1
/CONTIGUOUS	
/DOLLARS	
/FORMAT:record-type	VARIABLE=0
FIXED:N	
VARIABLE[:n]	
CONTROLLED[:n]	
/PROTECTION:code	[RWED, RWED, RWED, R]
/RELATIVE	
/SEQUENTIAL	/SEQUENTIAL
/INDEXED/KEY:value	/SEQUENTIAL

Prompt:

FILE?	file-spec

Command Parameter:

file-spec
The file specification for the new file must include a file name and a file type.

Command Qualifiers:

/ALLOCATION:n
Specifies n blocks of initial allocation for the file.

/BUCKETSIZE:m
Allowed only with indexed and relative files; specifies a unit of allocation of m blocks for each bucket. In TRAX, m may be a maximum of 16.

/CONTIGUOUS
Specifies contiguous space allocation for the file.

/DOLLARS
Specifies that the data be entered into the created file contains dollar signs in record position 1. The data entered must be terminated with a $EOD command.

/FORMAT:record type
Specifies the record type of the file.

The following record types are available:

FIXED:n
Specifies fixed length records n bytes long; n is required.

VARIABLE [:n]
Specifies variable length records. The n parameter defines the maximum length of the record; it is required if /RELATIVE is specified but is otherwise optional.
CONTROLLED [:n]
Specifies variable length records with a fixed control field. The n variable defines the maximum length of the record, including the fixed control field; it is required if /RELATIVE is specified but is otherwise optional. In all instances, the size of the fixed control field defaults to 2 bytes.

/PROTECTION:code
Protects the file specified in the code parameter.

/RELATIVE
Specifies relative organization for the file.

/SEQUENTIAL
Specifies sequential organization. This is the default.

/INDEXED
Specifies indexed organization. A /KEY qualifier is also required if /INDEXED is used.

/KEY:value
Specifies the access information for an indexed file. The value parameter may contain the following:

NUMBER:n
Specifies the level of the key field. If n=1, it indicates a primary key, n=2 indicates the first alternate, and so forth.
POSITION:n
Specifies the starting character of the key field.
SIZE:n
Specifies length of the key field.
(NUMBER, POSITION, and SIZE are required for each key value.)
UPDATE
Specifies that the key field is subject to change during the update process.
NOUPDATE
Converse of UPDATE, required on primary keys.
DUPLICATE
Specifies that the record may include duplicate keys. This is implicit if UPDATE is specified.
NODUPLICATE
Converse of DUPLICATE. Illegal with UPDATE.

Table 11-1 shows the legal combinations of UPDATE and DUPLICATE with primary and alternate keys.

Table 11-1 Valid Key Parameter Combinations

Key Type	UPDATE DUPLICATE	UPDATE NODUPLICATE	NOUPDATE DUPLICATE	NOUPDATE NODUPLICATE
Primary	Error	Error	Not supported	Default
Alternate	Default	Error	Allowed	Allowed

NOTES

1. The file-spec must contain a file name and a file type. If an existing version number is not specified, the highest existing version number plus one is used.
2. If sequential organization is specified or defaulted, you can include text in the file as follows:

Interactive	Batch File
>CREATE	$CREATE/DOLLARS file-spec
FILE?	file-spec
contents of file	contents of file, possibly including dollar signs
CTRL/Z	$EOD

3. If indexed or relative organization is specified, no data is accepted to fill the file.
4. If /INDEXED is specified, a primary key is also required. If any other organization is specified, /KEY is not permitted.
5. The qualifiers /ALLOCATE, /CONTIGUOUS, and /PROTECTION are only applicable when creating an empty file and not when filling the file with data.
6. The code option /PROTECTION specifies up to four categories of protection: SYSTEM, OWNER, GROUP, and WORLD. Up to four types of access can be specified for each category: READ (R), WRITE (W), EXTEND (E), and DELETE (D). The order of the access type codes R, W, E, and D is fixed. For example:

```
/PROTECTION:(SY:RWED,OWNER:RWED,GROUP:RE)
```

Examples:
This example creates the file ABC.TXT. and accepts lines from the terminal until you type CTRL/Z.

```
>CREATE
FILE? ABC.TXT
THIS IS THE CONTENT OF THE NEWLY-CREATED FILE.
^Z

>
```

This example creates file ACCOUNT.NDX as an indexed file with one key of reference which appears in the first byte of each record and is 10 bytes long.

```
>CREATE/INDEXED/KEY:(NUMBER:1,SIZE:10,POSITION:1)
FILE? ACCOUNT.NDX
>
```

In this example of batch usage, the $CREATE command creates a file that contains batch commands. The file created, COMMAND.CMD, begins with a $COBOL command and ends with a $RUN command. Since the records to be placed in COMMANDS.CMD contain dollar signs in record position 1, the /DOLLARS qualifier is necessary on $CREATE to identify all information up to the $EOD command as data.

```
$JOB
$CREATE/DOLLARS COMMANDS.CMD
$COBOL A.CBL
$LINK A.OBJ,[1,1]COBLIB/LIB,[1,1]RMSLIB/LIB
$RUN A
$EOD
$COPY MYFILE.CBL NEWFILE.CBL
$EOJ
```

/PROTECTION:code
Establishes the access rights for the directory file.

/VOLUME LABEL:volume-id
The volume-id is a name associated with each volume to verify the correct volume is used. When the incorrect volume-id is specified, the command is ignored. The volume-id consists of an alphanumeric string, 1 to 12 characters long.

Example:
This example creates the User File Directory [200, 34] on DB11:

```
>CREATE/DIRECTORY DB1:[200,34]
```

11.9.2 Creating a Directory

The CREATE/DIRECTORY command creates a user file directory (UFD) on the specified disk and enters the UFD name into the master file directory (MFD) on that disk. The disk must have been subjected to INITIALIZE and MOUNT commands before the CREATE/DIRECTORY command can be issued.

A nonprivileged user can create a UFD on a private volume, otherwise the UFD must be created by a privileged user.

Format:

CREATE/DIRECTORY [device-name] [ggg,mmm]

Command Qualifiers:	Default:
/ALLOCATION:n	
/PROTECTION:(code)	See Command Qualifier
/VOLUME_LABEL:volume-id	

Prompt:

DEVICE AND/OR DIRECTORY? device-name [ggg,mmm]

Command Parameters:

device-name
The device name of the disk to contain the directory.

When you do not specify a device name, the system creates the user file directory on the currently defaulted system disk.

[ggg,mmm]
The user file directory to be created. When you do not specify a UFD, the system uses the currently defaulted UFD.

The group number, ggg, and the member number, mmm, are octal values from 0 to 377.

Command Qualifiers:

/ALLOCATION:n
Initially allocate _ directory entires (rounded up to the next multiple of 32).

/PROTECTION:(code)
Establish the access rights for the directory file. Where the code is (RWED, RWED,RWED,RWED) to allow all accesses to all groups.

The code is positional and contains four groups: SYSTEM, OWNER, GROUP, and WORLD. Each group contains four positional access rights: read, write, extend, and delete. When a group is not specified, its access rights are not changed. When an access right is not specified, the group is not given that particular file access. Thus RWE and RD are acceptable, while DWR is not acceptable.

The code default is based on the volume protection code (usually [RWED, RWED,RWED,R]). The volume protection code is defined when initializing the volume. See the INITIALIZE command for more information.

/VOLUME_LABEL:volume-id
The volume-id is a name associated with each volume to verify that the correct volume is used. When the incorrect volume-id is specified, the com-command is ignored.

The volume-id may be up to 6-alphanumeric characters for magnetic tape volumes and up to 12-alphanumeric characters for all other volumes.

Example:

1. CREATE/DIRECTORY DB1:[200,34]
   Create a [200,34] user file directory on DB1:.

11.10 $DATA

The $DATA command indicates the beginning of a batch data block. A data block is necessary whenever you must supply data to a task running under batch control.

In a batch file, any line that does not begin with a dollar sign is treated as data. Thus in many cases you can omit the $DATA command.

The $DATA command is required only when you need to use one of its qualifiers, as described below.

Format:

$DATA [/qualifier]

Command Qualifiers:

/DOLLARS
/NOCOPY

Prompt:

None.

Command Parameters:

None.

Command Qualifiers:

/DOLLARS
Alerts the system that lines of data may begin with dollar signs. Without this qualifier, lines that begin with dollar signs are treated as commands and thus terminate the data block. When this qualifier is present, all information is treated as data until an $EOD command is encountered.

/NOCOPY
Specifies that the data block to follow not be included in the log file for the batch job.

NOTE

In general, you need preface a data block with a $DATA command only if you need to use the /DOLLARS qualifier, the /NOCOPY qualifier, or both.

Example:
The batch job includes a $RUN command that requires input data. The data includes some lines that begin with dollar signs. The data block is not included in the Log File.

```
$JOB
$RUN PROCESS
$DATA/NOCOPY/DOLLARS
INCOME
$76.05
$346.55
$5.80
SPENT
$84.00
$4.89
$EOD
$EOJ
```

11.11 DEALLOCATE

Use the DEALLOCATE command to release a private device which permits other users to access the device. Nonprivileged users can deallocate devices that they allocate. A privileged user can deallocate any private device. The system automatically deallocates private devices when the owner logs off the system.

When an allocated disk contains a checkpoint file, you cannot deallocate the disk as long as the checkpoint file exists.

Format:

DEALLOCATE device-name

Command Qualifier:	Default:
None	

Prompt:

DEVICE? device-name

Command Parameter:

device-name
The physical device name of the device to be deallocated.

Command Qualifier:

None

Example:

1. DEALLOCATE DB2:

   Deallocate the DB2: disk. The system and other private users are now permitted to use DB2:.

11.12 DEASSIGN

11.12.1 Deassigning Logical Names

The DEASSIGN command removes the specified logical name from the device name of the specified level such as local, global, and login. There is no automatic deassignment for either global or login assignments when you log off the system. Each of these assignments must be explicitly deassigned by a privileged user. Logical assignments can be deassigned explicitly or automatically when logging off the system.

Format:

DEASSIGN logical-name

Command Qualifiers:	Default:
/LOCAL	/LOCAL
/GLOBAL	
/ALL	
/LOGIN	
/TERMINAL:TTnn:	

Prompt:

LOGICAL NAME? logical-name

Command Parameter:

logical-name
The logical name is limited to two alphabetic characters followed by one or two octal numbers and a colon.

Command Qualifiers:

/LOCAL
Delete logical names from commands and tasks initiated from the same terminal. Both privileged and nonprivileged users can specify the qualifier.

/GLOBAL
Delete global logical names. Only privileged users can specify the /GLOBAL qualifier.

/ALL
Delete all logical names. Only privileged users can specify the /ALL qualifier. The logical name parameter must be omitted when /ALL is specified. The /ALL qualifier can be used in conjunction with any other DEASSIGN command qualifier.

/LOGIN
Delete the login assignment of the logical name.

/TERMINAL:TTnn:
Delete the local assignment of the logical name for the TTnn: terminal. Only privileged users can specify the /TERMINAL qualifier. The /TERMINAL qualifier can be used in conjunction with the /LOCAL and /LOGIN qualifier.

Examples:

1. DEASSIGN DB1:

   Deletes the local assignment of DB1: associated with the issuing terminal.
2. DEASSIGN/LOCAL/ALL

   Deletes all the local assignments associated with the issuing terminal.

11.12.2 Deassigning a Processor from a Queue

Use the privileged DEASSIGN/QUEUE command to dissociate a processor and queue. The processor and queue remain listed in the queue file. When the processor is marked for delete, the DEASSIGN command is already in force. Further jobs cannot be dispatched from the queue to the processor until the association is restored with an ASSIGN command.

Format:

DEASSIGN/QUEUE queue-name processor-name

Command Qualifier:	Default:
/QUEUE	

Prompt:

QUEUE NAME? queue-name
PROCESSOR NAME? processor-name

Command Parameters:

queue-name
The print or batch queue to be deassigned from the processor. A queue name may be from 1- to 6- characters and must be unique in the queue file.

processor-name
The print or batch processor to be deassigned from the queue. You must specify the processor name using the following format:

ssPnn

where:

1. The two letters, ss, are either LP for a print processor or BA for a batch processor.
2. The two letters, nn, are either 1- or 2- digit octal number.

   For a print processor, the value of nn corresponds to the physical device name of the printer, such as LPP0 for the LP0: line printer.

   For a batch processor, the value of nn is a unique octal number, such as BAP0 for the first batch processor.

Command Qualifier:

/QUEUE
The /QUEUE qualifier is required to deassign a processor from a queue.

Example:

1. DEASSIGN/QUEUE BATCH BAP0

   Deassign the batch processor, BAP0, from the batch queue, BATCH.

11.13 DELETE

11.13.1 Deleting Files

Use the DELETE command to delete the specified file(s) from the directory and releases the occupied space. Before you can delete any file, you must have delete access to the file. See the DIRECTORY/FULL command to display file access rights.

Format:

DELETE file-spec,...,file-spec

Command Qualifier: Default:
None

Prompt:

FILE? file-spec

Command Parameter:

file-spec
The file specification must be fully defined (that is, an explicit version) or have a wild card attribute.

Command Qualifier:

None

Example:

1. DELETE TEST.TSK;*
   Delete all versions of TEST.TSK.

11.13.2 Deleting a Queue, Processor, or Queue Job

Use the privileged /QUEUE and /PROCESSOR qualifiers of the DELETE command to remove a queue, processor, or job entry from the queue file. The queue manager cannot accept a job for a queue when the queue is either deleted or marked for delete. When a queue is deleted, any processors assigned to it are automatically deassigned.

When deleting an idle processor, assignments are broken immediately, when the processor is busy, it is marked for delete and is deleted when the current job is completed or aborted.

Formats:

DELETE name

Command Qualifiers:	Default:
/QUEUE	
/PROCESSOR	

Parameter Qualifiers:	Default:
/ERASE	
/JOB:[[ggg,mmm]]job-name	

Prompt:

None

Command Parameters:

name
The queue name or processor name to be deleted. A queue name may be from 1- to 6- alphanumeric characters and must be unique in the queue file. You must specify the processor name using the following format:

ssPnn

Where:

1. The two letters, ss, are either LP for a print processor or BA for a batch processor.
2. The two letters, nn, are either a 1- or 2- digit octal number.

   For a print processor, the value of nn corresponds to the physical device name of the printer, such as LPP0 for the LP0: line printer.

   For a batch processor, the value of nn is a unique octal number such as BAP0 for the first batch processor.

The special queue name, ENTRY:(n,m), deletes a job entry in the queue file. The job entry is identified in the queue file as a pair of numbers separated by a comma and enclosed in parentheses. Since the system uses ENTRY as a queue name, you must specify the entire word. The system accepts an abbreviated form of ENTRY as another queue name.

Job entry identifiers are displayed when you specify a full listing with the SHOW QUEUE command.

Command Qualifiers:

/QUEUE
The system deletes the queue, name, when you specify the /ERASE parameter qualifier or deletes the specified job when you use the /JOB parameter qualifier.

/PROCESSOR
Delete the processor, name.

Parameter Qualifiers:

/ERASE
Use /ERASE as a queue name qualifier. The requirement of this qualifier protects against inadvertent deletion of queues.

/JOB:[[ggg,mmm]]job-name
Use the /JOB qualifier, with the /QUEUE qualifier, to delete a job entry. The job entry and its files are deleted from the queue file. (The batch or print files are not deleted.) When the job entry is currently being processed, the queue manager issues a STOP processor command, aborts the current job and deletes it, and issues a START processor command to process the next job. The /JOB qualifier requires user identification code from the originating job, [ggg,mm], and the job name. By default, the queue manager assigns the job name as the first six characters from the first filename entry. You can specify the job name with the /JOB command qualifier associated with the PRINT or SUBMIT commands.

When the queue file contains one or more identical job names with the same user identification code, the system deletes the first job. All the other jobs remain unaffected. In this case it is better to use the ENTRY:(n,m) form of the command to delete the specific job.

Examples:

1. DELETE/QUEUE BATCH/ERASE

   Delete the batch queue, BATCH.
2. DELETE/PROCESSOR LPP0

   Delete the print processor, LPP0.
3. DELETE/QUEUE ENTRY:(123,676)

   Delete the (123,676) job entry in the queue file.

11.14 DIRECTORY

The DIRECTORY command displays the directory information of specified files or the contents of your current default directory.

Format:

```
>DIRECTORY [/qualifiers] [file-spec [,...]]
```

Command Qualifiers:	Default:
/FULL	/BRIEF
/BRIEF	
/SUMMARY	
/FREE	
/PRINT	
/OUTPUT: file-spec	
/ATTRIBUTES	

Command Parameter:

file-spec
Specifies the directory entries to be listed. If omitted, all directory entries are listed.

Command Qualifiers:

/BRIEF
The entry display contains only the file specification, block count, and creation date. This is the default. See Example 1.

/FULL
A complete directory entry listing is displayed. See Example 2.

/SUMMARY
Only the total number of blocks allocated to all files in the directory is displayed.

/FREE
The free space available either on the system device or the specified device is displayed.

/ATTRIBUTES
Gives a description of the specified files that includes the full RMS attributes. See Example 3.

/OUTPUT:file-spec
Forces the display to be placed in a file according to the file-specification.

/PRINT
Causes the display to appear on the line printer.

NOTES

1. If no files are specified, a directory list of your current default UFD on your default device is given.
2. One or more file-specs may be given.
3. If no filetype is specified, a wildcard for file-type is assumed. If no file version is given, the highest version number is assumed.

Examples:
The following examples of DIRECTORY commands show the type of information you can expect in response to different qualifiers. The same file specification is used in each case.

1. This is a /BRIEF (default) directory listing.

```
DIRECTORY DB0:[40,40]
18-JUL-78 18:25

A.LST;1                 1.          18-JUL-78 14:14
A.ODL;1                 1.          18-JUL-78 14:14
A.OBJ;1                 2.          18-JUL-78 14:14
A.TSK;1                 27.      C  18-JUL-78 14:14
A.CBL;2                 1.          18-JUL-78 14:15

TOTAL OF 32./32. BLOCKS IN 5. FILES
```

2. This is a /FULL directory listing.

```
>DIRECTORY/FULL A.*


DIRECTORY DB0:[40,40]
18-JUL-78 18:26

A.LST;1                 (14271,6)       1./1.         18-JUL-78 14:14
  [40,40]  [RWED,RWED,RWED,R]
A.ODL;1                 (15040,23)      1./1.         18-JUL-78 14:14
  [40,40]  [RWED,RWED,RWED,R]
A.OBJ;1                 (15233,56)      2./2.         18-JUL-78 14:14
  [40,40]  [RWED,RWED,RWED,R]
A.TSK;1                 (15432,7)       27./27.    C  18-JUL-78 14:14
  [40,40]  [RWED,RWED,RWED,R]
A.CBL;2                 (17211,10)      1./1.         18-JUL-78 14:15
  [40,40]  [RWED,RWED,RWED,R]

TOTAL OF 32./32. BLOCKS IN 5. FILES
```

3. This is an /ATTRIBUTES directory listing.

```
>DIRECTORY/ATTRIBUTES A.*

SY0:[40,40]A.LST;1      FILE ORGANIZATION:      SEQUENTIAL
CREATED: 18-JUL-1978 14:14
FILE PROTECTION:        [RWED,RWED,RWED,R]
RECORD FORMAT:          VARIABLE
RECORD ATTRIBUTES:      CARRIAGE RETURN
FILE ATTRIBUTES:
        ALLOCATION= 1   EXTEND QUANTITY=0

SY0:[40,40]A.ODL;1      FILE ORGANIZATION:      SEQUENTIAL
CREATED: 18-JUL-1978 14:14
FILE PROTECTION:        [RWED,RWED,RWED,R]
RECORD FORMAT:          VARIABLE
RECORD ATTRIBUTES:      CARRIAGE RETURN
FILE ATTRIBUTES:
        ALLOCATION= 1   EXTEND QUANTITY=0

SY0:[40,40]A.OBJ;1      FILE ORGANIZATION:      SEQUENTIAL
CREATED: 18-JUL-1978 14:14
FILE PROTECTION:        [RWED,RWED,RWED,R]
RECORD FORMAT:          VARIABLE
RECORD ATTRIBUTES:
FILE ATTRIBUTES:
        ALLOCATION= 2   EXTEND QUANTITY=0
```

```
SY0:[40,40]A.TSK;1      FILE ORGANIZATION:      SEQUENTIAL
CREATED:.18-JUL-1978 14:14
FILE PROTECTION:        [RWED,RWED,RWED,R]
RECORD FORMAT:          FIXED=512
RECORD ATTRIBUTES:
FILE ATTRIBUTES:
        ALLOCATION= 27  EXTEND QUANTITY=0
                        CONTIGUOUS

SY0:[40,40]A.CBL;2      FILE ORGANIZATION:      SEQUENTIAL
CREATED: 18-JUL-1978 14:15
FILE PROTECTION:        [RWED,RWED,RWED,R]
RECORD FORMAT:          VARIABLE
RECORD ATTRIBUTES:      CARRIAGE RETURN
FILE ATTRIBUTES:
        ALLOCATION= 1   EXTEND QUANTITY=0

>
```

11.15 DISMOUNT

Use the DISMOUNT command to logically disconnect a volume from the file system. Once issued, the command immediately inhibits additional file access by writing inhibit information on the volume. (As explained in the MOUNT command, the system verifies that file access is permitted before each access.) The system then suspends dismounting the volume while files are being accessed. The system issues a message to the CO0: terminal when the dismount operation is complete.

Privileged users can dismount any volume, while nonprivileged users can dismount only devices that they allocated. For efficient resource management, volumes should be dismounted when they are no longer needed. The system dismounts private volumes when a user logs off the system.

Format:

DISMOUNT device-name [volume-id]

Command Qualifier:	Default:
None	

Prompts:

DEVICE? device-name

VOLUME LABEL? volume-id

Command Parameters:

device-name

The logical or physical device name of the device to be dismounted.

volume-id
The volume-id is a name associated with each volume. The volume-id is mandatory for magnetic tape volumes and optional for all the other volumes. When the volume-id is specified, the system verifies that the correct volume is used. The command is ignored when an incorrect volume-id is specified. The volume-id may be up to 6-alphanumeric characters for magnetic tape volumes and up to 12-alphanumeric characters for all other volumes.

Command Qualifier:

None

Example:

1. DISMOUNT DB0: VOLNAME

   Dismount the disk, DB0:, and verify that the volume name is VOLNAME.

11.16 EDIT

The EDIT command invokes the editor to edit or create the specified file.

Format:

EDIT file-spec

Prompt:

FILE? file-spec

Command Parameters:

file-spec
The specification of the file to be edited. The file specification must include a file name and a file type.

Command Qualifiers:

None

NOTES

1. If you do not provide a version number, the highest existing version is used. If a file does not exist as specified, a new file is created with version number 1.
2. Details of the use of the editor may be found in the DEC EDITOR Reference Manual.

Example:
The following sequence initiates an edit session on the file EASY.CBL.

```
>EDIT
FILE? EASY.CBL
*
```

*The asterisk is a prompt for an editor command. When you want to terminate the edit session, enter the editor command EXIT. The DCL prompt (>) will appear on the terminal.

11.17 $EOD

The $EOD (End of Data) command terminates a data stream initiated by a $DATA command, or the input to a file created by a $CREATE/DOLLARS command. The command may only be given in batch mode. A data stream that is not initiated by a $DATA command does not require an $EOD command for termination.

Format:

$EOD

NOTE

The command has no parameters or qualifiers.

Example:
This example uses $EOD to terminate a data block. The /DOLLARS qualifier instructs the system to accept the following lines of text as input to the file rather than batch commands to be processed.

```
$CREATE/DOLLARS TRAN.DAT
$UPDATE DATA FOR 27FEB
A601-450
$35.42
$102.99
T79-132
$824.09
$EOD
```

11.18 $EOJ

The $EOJ (End of Job) command terminates a batch job, dismounting and deallocating any allocated devices.

Format:

$EOJ

NOTES

1. The command has no parameters or qualifiers.
2. The $EOJ command is the last command in a batch job command stream. An $EOJ command is implied at the end of a batch command file if one is not included explicitly.

Example:
The $EOJ command ends the batch job and is analogous to a LOGOUT command ending an interactive terminal session.

```
$JOB
$MOUNT DB0: MAR27A
$RUN TEST
$DISMOUNT DB0:
$EOJ
```

11.19 HELP

Use the HELP command to display information about the commands and the associated qualifiers. When no parameters are specified, the system displays a complete list of commands on the requesting terminal. When a qualifier is displayed with two asterisks (**), further information is available by specifying the qualifier.

Format:

HELP [[command] [qualifier]]

Command Qualifier:	Default:
None	

Prompt:

None

Command Parameters:

command
The command and its qualifiers, if any, are displayed. When a qualifier is displayed with two asterisks (**) after it, more information is available on the qualifier.

qualifier
Displays the keywords available for the qualifier as applied to the command.

Command Qualifier:

None

Examples:

1. HELP
   The following commands are available

ABORT	ALLOCATE	APPEND	ARCHIVE
ASSIGN	BASIC	COBOL	COPY
CREATE	DEALLOCATE	DEASSIGN	DELETE
DIRECTORY	DISMOUNT	EDIT	INITIALIZE
LIBRARIAN	LINK	LOGIN	LOGOUT
MACRO	MERGE	MESSAGE	MOUNT
PRINT	PURGE	RENAME	RUN
SET	SHOW	SORT	START
STOP	TYPE	UNLOCK	

   – For more information type 'HELP' followed by the command.

2. HELP INITIALIZE

```
INITIALIZE[/QUALIFIER[S]]
                              DEVICENAME  VOLUMELABEL
          EXTENSION:N
          DENSITY:1600
                  800
          PROTECTION:[RWED,RWED,RWED,RWED]
          INDEX:BEGINNING
                MIDDLE
                END
                BLOCK:N
          HEADERS:M
          MAXIMUM:P
          [NO]VERIFIED
          VOLUME-PROTECTION:
              [RWED,RWED,RWED,RWED]
          OWNER:[UFD]
          WINDOW:Q
OR, INITIALIZE/QUEUE QUEUENAME/QUALIFIER
                                        BATCH
                                        PRINT
OR, INITIALIZE/PROCESSOR
    PROCESSORNAME/QUALIFIER
                        FORMS:N
                        FLAG-PAGE:N
                        LOWERCASE
                        UPPERCASE
```

Display the functions available with the INITIALIZE command.

3. HELP SET TERMINAL

```
SET TERMINAL[:TTN]   OPTION
                     TYPE:[NO]SCOPE
                     LOWERCASE
                     UPPERCASE
                     [NO]PRIVILEGED
                     [NO]REMOTE
                     [NO]SLAVE
                     [NO]HOLD-SCREEN
                     [NO]ESCAPE-SEQUENCE
                     SPEED:(N,M)
```

Display the keywords available with the SET TERMINAL command.

11.20 $IF

The $IF command specifies alternative action if a specified status condition occurs on a command. It is used only in batch jobs.

Format:

$IF status-level THEN action

Prompts:

None

Command Parameters:

status-level
One of the following:

SUCCESS
WARNING
ERROR
SEVERE ERROR

action
One of the following:

GOTO label
CONTINUE
STOP

Command Qualifiers:

None

NOTES

1. The status-level resulting from the execution of the command preceding the $IF command is checked. If that status-level is equal to the status-level given in the $IF command, the THEN clause is executed. Otherwise the THEN clause is not executed, and the batch job continues with the next command in the file.
2. The label of the $GOTO phrase must be the label of a command appearing after the $IF command.

Example:
The following $IF command causes the batch job to terminate immediately if the preceding command results in a status of ERROR. Otherwise, the job proceeds sequentially.

```
$IF ERROR THEN STOP
```

11.21 INITIALIZE

Use the INITIALIZE command to create a file-structured volume or to name a queue or processor to the batch and print facilities.

11.21.1 Initializing a Volume

Produce a file-structured volume on disk and magnetic tape. The command destroys all existing files on the volume. The system creates a master file directory (MFD) on the disk and creates a volume label and dummy file on the magnetic tape.

Format:

INITIALIZE device-name volume-id

Command Qualifiers:	Default:
/DENSITY:n	n = 800
/EXTENSION:n	
/HEADERS:m	
/INDEX:location	
/MAXIMUM:p	
/OWNER:[ggg,mmm]	
/PROTECTION:[code]	See the command qualifier.
/[NO]VERIFIED	
/VOLUME_PROTECTION:[code]	See the command qualifier.
/WINDOW:a	

Prompts:

DEVICE? device-name

VOLUME LABEL? volume-id

Command Parameters:

device-name
The device name of the device to be initialized.

volume-id
The volume-id is a name associated with each volume. The volume-id may be up to 6-alphanumeric characters for magnetic tape volumes and up to 12-alphanumeric characters for all other volumes.

The volume-id is requested by other commands, such as the MOUNT command, to insure the proper volume is used.

Command Qualifiers:

/DENSITY:n
Specifies the bit packing density (in bits per inch, bpi) of the magnetic tape to be initialized. Acceptable values are either 800 bpi or 1600 bpi. When not specified, 800 bpi is used.

/EXTENSION:n
After a task has exhausted the space allotment for a file, the system extends the allocated file area by the n number of blocks. Each block contains 512(10) bytes. Files are extended by tasks after exhausting the allocated file space and more space is required. The value of n is expected to be decimal format. For a task to extend a file, the task must be permitted write and extend access.

The value of n can be changed when the volume is mounted.

/HEADERS:m
The number of file headers to allocate initially in the index file. The value of number is expected to be in decimal format.

/INDEX:location
Position the index file on the volume at the specified location.

BEGINNING
Place the index file at the beginning of the volume.
MIDDLE
Place the index file at the middle of the volume.
END
Place the index file at the end of the volume.
BLOCK:n
Place the index file at the n block of the volume.

/MAXIMUM:p
Specifies the maximum number of files permitted on the volume. The value of number is expected to be in decimal format.

/OWNER:[ggg,mmm]
Specifies the UIC of the owner of the volume. The group number, ggg, and the member number, mmm, are octal values from 0 to 377. The square brackets, [ ], are required syntax.

/PROTECTION:(code)
Specify the default file protection code. When the system creates a file, the system assigns a default file protection code to it. You can change the code with the SET PROTECTION command. The code contains four groups: SYSTEM, OWNER, GROUP, and WORLD. Each group contains four positional access rights: read, write, extend, and delete. When a group is not specified, its access rights are not changed. When an access right is not specified, the group is not given that particular file access. The code default is (SY:RWED,OW:RWED,GR:RWED,WO:R) for disk volumes and (SY:RWED,OW:RWED) for magnetic tape volumes.

/VERIFIED
/NONVERIFIED
Include bad block processing in the volume initialization. When specifying VERIFIED, the system reads the bad block file created by the BAD function of the SETUP utility.

When specifying NOVERIFIED, the system accepts block specifications from the terminal. The program prompts for bad blocks with the display:

```
INI>BAD=
```

Bad blocks may be entered by specifying either of the following octal formats:

nnnnn	A single block.
nnnnn,mmm	A contiguous series of mmmm blocks beginning at nnnnn. A null line (carriage return) terminates bad block input.

/VOLUME_PROTECTION:(code)
Specify the volume protection code for tasks to access the volume. Refer to the /PROTECTION:(code) description for the format. The code default is (SY:RWED, OW:RWED,GR:RWED,WO:RWED) for disk volumes and (SY:RWED,OW:RWED,GR:R) for magnetic tape volumes.

/WINDOW:a
The (decimal) number of mapping pointers to be allocated for file windows.

Examples:

1. INITIALIZE DB0: VOLLABEL

   Initialize the disk, DB0:, with the volume-id of VOLLABEL.
2. INITIALIZE/INDEX:END/OWNER:[1,4] DB1: DCLVOL2

   Initialize DCLVOL2 on DB1: with the index file located at the end of the volume. The owner UIC is [1,4].

11.21.2 Initializing a Queue or Processor

Use the privileged /QUEUE qualifier of the INITIALIZE command to either enter a queue name into the queue file or activate a batch or print processor. When you initialize a processor, it is started but not assigned to a queue. When you initialize a queue, the system starts the queue. Before jobs can be processed, the system requires that you assign a processor to the queue. Before you initialize a print processor, you must first initialize the associated auto-spooled queue. For instance, you must initialize the LPQ0 auto-spooled queue before you can initialize the associated LPP0 print processor. When a print processor is initialized, the processor name is derived from the name of an existing printer device in the system such as LPP0 is the processor for the LP0: printer. This printer is set SPOOLED to facilitate auto-spooling, and the processor is automatically assigned to the associated auto-spooling queue.

Formats:

INITIALIZE/QUEUE queue-name

Command Qualifier:	Default:
/QUEUE	
Parameter Qualifiers:	**Default:**
/BATCH	
/PRINT	/PRINT

INITIALIZE/PROCESSOR processor-name

Command Qualifier:	Default:
/PROCESSOR	
Parameter Qualifiers:	**Default:**
/FORMS:n	n=0
/FLAG_PAGE:n	n=1
/LOWERCASE	
/UPPERCASE	

Prompt:

None

Command Parameter:

queue-name
A queue name may be from 1- to 6- characters and must be unique in the queue file.

Two reserved queue names, BATCH and PRINT, are the default queues for the SUBMIT and PRINT command, respectively. When you specify these queue names, the parameter qualifier can be omitted. In addition, the ALL and ENTRY queue names are not permitted due to the syntax requirements for other queue manipulation commands.

processor-name
You must specify the processor name using the following format:

ssPnn

where:

1. The two letters, ss, are either LP for a print processor or BA for a batch processor.
2. The two letters, nn, are either a 1- or 2- digit octal number.

   For a print processor, the value of nn corresponds to the physical device name of the printer, such as LPP0 for the LP0: line printer.

   NOTE
   Before you can initialize a print processor, you must initialize the associated auto-spooled queue.

   For a batch processor, the value of nn is a unique octal number, such as BAP0 for the first batch processor.

Command Qualifiers:

/QUEUE
The /QUEUE qualifier is required to initialize a queue which the system enters into the queue file and starts the queue.

/PROCESSOR
The /PROCESSOR qualifier is required to initialize a processor which the system activates and makes it known to the queue manager.

Parameter Qualifiers:

/BATCH
The qualifier specifies the queue as a batch queue. The qualifier can be omitted when the queue name is BATCH.

/PRINT
The qualifier specifies the queue as a print queue. The qualifier can be omitted when the queue name is PRINT.

/FORMS:n
Specifies the default form the print processor is to process initially. The value of n can be from 0 to 255; the default is 0.

/FLAG_PAGE:n
Specifies the number (n) of flag pages to precede each job printed by the particular processor; where n can be 0, 1, or 2. When you do not specify a value for :n, the system use n=1.

/LOWERCASE
Specifies the character set of the printer to be uppercase and lowercase characters.

/UPPERCASE
Specifies the character set of the printer to be only uppercase characters.

Examples:

1. INITIALIZE/QUEUE BATCH

   Initialize a batch queue with the name BATCH.
2. INITIALIZE/QUEUE EXPRES/BATCH

   Initialize a batch queue with the name EXPRES.
3. INITIALIZE/PROCESSOR BAP0

   Initialize a batch processor with the name BAP0.
4. INITIALIZE/QUEUE LPQ1
   INITIALIZE/PROCESSOR LPP1/FLAG:2

   Initialize the LPQ1 print processor and initialize the LPP1 print processor with two flag pages.

11.22 $JOB

The $JOB command is used only in batch mode and marks the beginning of a batch job. It is the batch mode equivalent of an interactive LOGIN command.

Format:

$JOB [/qualifier] jobname [uic]

Command Qualifier:	Default:
/TIME=xx	No time limit

Prompts:

None

Command Parameters:

job-name
Specifies the name by which the batch job will be identified in the batch log.

uic
Specifies the User Identification Code. This is a privileged parameter that enables the batch job to log in under a different account than the one from which the job was submitted.

Command Qualifier:

/TIME=xx
Specifies the maximum number of minutes in wall clock time that the batch job is allowed to run. This parameter is optional; if omitted, the system assumes no time limit.

11.23 LINK

The LINK command invokes the TRAX linker to convert object modules into executable task images. It produces output as directed by command qualifiers. For further information, see the *TRAX Linker Reference Manual.*

Format:

LINK [/qualifiers] [file-spec [/file-qualifiers],...]

Command Qualifiers:	Default:
/BASIC	
/CHECKPOINT:SYSTEM :TASK	/CHECKPOINT:S
/NOCHECKPOINT	
/CROSS REFERENCE	
/DEBUG [:debug-file-spec]	
/[NO] DUMP	/NODUMP
/[NO] FULL SEARCH	
/MAP: map-file-spec [/FULL] /NARROW /SHORT /WIDE	
/NOMAP	/NOMAP
/OPTIONS [:file-spec]	
/OVERLAY [overlay-file-spec]	
/[NO] RECEIVE	/RECEIVE
/SEQUENTIAL	
/SYMBOLS [symbol-file-spec]	
/[NO] SYMBOLS	/NOSYMBOLS
/TASK: task-file-spec	TASK=default-file-spec
/NOTASK	

File Qualifiers:	Default:
/[NO] CONCATENATED	/CONCATENATED
/DEFAULT LIBRARY: file-spec	
/LIBRARY [:module-list]	
/[NO] MAP	See qualifier description.
/SELECT SYMBOLS	See qualifier description.

Prompt:

FILE? file-spec [/file-qualifiers] , . . .

Command Parameter:

file-spec
Specifies an input file containing object modules. It must not be present if the command qualifier /OVERLAY is specified.

If the file name is given with no file type, the default file type of .OBJ is used for an object file and .OLB for a library. File specifications for symbol table files must include a file type .STB and specifications for an overlay description file must include file type .ODL.

Command Qualifiers:

/BASIC
Identifies the input file as a command file produced by issuing the BUILD command to the BASIC-PLUS-2 compiler. The Linker decodes the command file and the task image file according to information supplied in the command file.

The /BASIC qualifier is valid only if the input file was generated this way.

/CHECKPOINT [:keyword]
/NOCHECKPOINT
Identifies the Linker task as checkpointable when /CHECKPOINT is specified. The optional keyword is SYSTEM or TASK; this specifies where the checkpoint space is allocated. TASK requests checkpoint space within the task image file, and SYSTEM requests system checkpoint space. The qualifier /CHECKPOINT:SYSTEM is the default.

You should avoid specifying /NOCHECKPOINT, as this option seriously degrades overall system performance. See Note for explanation.

/CROSS REFERENCE
Requests that a symbol cross-reference listing be appended to the memory allocation (MAP) file; thus the /MAP qualifiers must also be present for this qualifier to be effective.

If /CROSS REFERENCE is not specified, no cross-reference listing is produced.

/DEBUG [:debug-file-spec]
Specifies incorporation of a debugging aid in the task image file. If debug-file-spec is omitted, the system standard debugging aid is used. You can incorporate a different debugging aid by specifying a debug-file-spec. The user generated debugging aid must be in object module format.

See the *TRAX Linker Reference Manual* for further information including a debugging aid.

/DUMP
/NODUMP
/DUMP requests a post-mortem dump if your task is terminated abnormally.

/NO DUMP is the default.

/FULL SEARCH
/NOFULL SEARCH
Specifies a full search of all co-tree overlay segments for a matching definition or reference, when processing modules from the default object module listing.

NOFULL SEARCH is the default.

/MAP [:map-file-spec [/map-file-qualifier]]
/NOMAP
Instructs the linker to produce a memory allocation file (with file type .MAP) when linking the task image file.

If you specify /MAP without a map-file-spec, the memory allocation file is spooled directly to the line printer. It remains on your file directory taking the task file name and the file type .MAP until it is deleted after printing.

If you include the map-file-spec, you may omit the file type field and the linker will use the file type .MAP.

/NOMAP is the default if /MAP is not specified.

The following file qualifiers may be applied to the map-file-spec.

/FULL
The Linker will include all modules in the memory allocation file, even those which explicitly or by default have the NOMAP input file qualifier.
/NARROW
The Linker produces a map listing 72 characters wide, suitable for printing on an output terminal.

/SHORT
Tells the Linker to include only the segment headings in the memory allocations file
/WIDE
Produces a map 132 characters wide, suitable for printing on a line printer. When /MAP is specified, this is the default file qualifier.

/OPTIONS[:file-spec]
Provides or prompts for Linker option input. See *TRAX Linker Reference Manual* for detailed information on Linker options.

If no file-spec argument is present, the Linker prompts for Linker option input lines as follows:

```
OPTIONS?
```

This prompt continues after each line of option input that you enter, until you type a line that ends with a slash (/), as follows:

```
OPTIONS?  /<CR>

OPTIONS?  :  OPTION-input/<cr>
```

When the file-spec is included, the linker treats that file as a series of option input lines. Interactive prompting for options does not occur. The default file type for the input file is .CMD.

/OVERLAY[:overlay-file-spec]
Specifies an Overlay Description Language (ODL) file that will govern the creation of the task image file.

Only an overlay description file is allowed with this qualifier. See the *TRAX Linker Reference Manual* for information on overlay descriptions.

/[NO]RECEIVE
Enables the resultant task to receive direct messages via the executive SEND directive.

/RECEIVE is the default. To disable the feature, the /NO RECEIVE qualifier is required.

/SEQUENTIAL
Causes the task image to be constructed from the object files in the order stated in the Link command string. If /SEQUENTIAL is not present in the command string, the Linker records the object program files alphabetically, not sequentially.

See the *TRAX Linker Reference Manual* for further description of task image storage allocation in detail.

/SYMBOLS[symbol-file-spec]
/NOSYMBOLS
/SYMBOLS specifies creation of a symbol table definition file by the Linker. If symbol-file-spec is present, the file type is optional; if the type is absent, it defaults to .STB.

If symbol-file-spec is absent, the first input file name becomes the file name, with .STB the default file type.

/NOSYMBOLS is the default qualifier.

/TASK [:task-file-spec]
/NOTASK
Specifies the name of the task image file. If task-file-spec is present, the file type is optional; if file type is absent, it defaults to .TSK. If file-spec is absent, the first input file name becomes the file name of the task image file, with .TSK the default file type.

/TASK is the default. If /NOTASK is used, the linker processes the input for unresolved symbol references but does not produce a task image file.

File Qualifiers:

/[NO] CONCATENATED
/CONCATENATED specifies processing of all modules in the input file to form the task image, and is the default.

/NOCONCATENATED causes the Linker to process only the first object module, regardless of the number present.

The /LIBRARY qualifier overrides this qualifier.

/DEFAULT LIBRARY:file-spec
Specifies the default library file to be used for resolving undefined global symbol references. This overrides the default system library LB0=[1,1] SYSLIB.OLB.

If the specified library is empty, the default library reverts to the system library.

/LIBRARY:[(]module[, . . .)]
Identifies the associated file (that is, the input file specification modified by this qualifier) as an object module library file. /LIBRARY is required for any input library file, and its use is prohibited for any other type of file.

If no modules are specified, the Linker searches the library file to resolve undefined global symbol references. The Linker extracts any and all modules that resolve undefined references and includes them in the task image file.

If you specify module names, the file is defined as a library file (file type .OLB) of relocatable object modules, and the modules named are copied for inclusion in the task image.

The module names are defined at assembly time. You may specify up to eight modules, and only those specified are included in the task image.

To direct the Linker to search a library file for both global symbol references and selected modules needed in the task image, you must include both forms of the qualifier, using separate file specifications.

/[NO] MAP
Specifies inclusion of this file in the memory allocation map.

If /NOMAP is specified, no details of modules contained in the file will appear in the memory allocation map or cross-reference listing. /NOMAP, when qualifying an input file, is overridden by the command qualifier FULL SEARCH.

For a system library file, resident libraries, and common areas, /NOMAP is the default qualifier. For user supplied object module input files, /MAP is the default qualifier.

/SELECT SYMBOLS
Instructs the Linker to search the file only for those global symbols for which an undefined reference exists. The Linker uses only the required symbol definitions.

This qualifier is useful when an input file is the symbol table (file type .STB) output of another LINK command, because it reduces the size of the symbol table search and improves system performance.

If /SELECT SYMBOLS is absent, all global symbols from the input file are included in the task image file; that is the default condition.

If the /LIBRARY or /CONCATENATED qualifier is in effect, /SELECT SYMBOLS is active for each module of the input file.

NOTE

Checkpointing is recommended as good programming practice. If a task (called Task A) is running and then a task of higher priority (called Task B) enters the system. Task A can be interrupted if it has been defined as checkpointable. The current state of Task A is recorded in the selected checkpoint area. When its required system resources become available again, it is reinstalled and resumed in the state that existed at the time of the interrupt.

Examples:

The file specification INTEREST.CMD contains a BASIC-PLUS-2 program. The file type .CMD is added to the file name by default.

```
>LINK/BASIC INTEREST
```

After execution of this LINK command, an executable task image called INTEREST.TSK is ready.

The following command directs the task OVERLAY.TSK to be created and the map file OVERLAY.MAP to be generated and spooled. The optional input specifies that DB0: will be assigned to LUN 8. The task will be built from the overlay descriptor file OVERLAY.ODL and will include the standard debugging aid.

```
>LINK/DEBUG/OVERLAY:OVERLAY/OPTIONS/MAP
OPTIONS?ASG=DB0:8
OPTIONS/?
```

11.24 LOGIN

Initiate a user session at a terminal. A valid user-id must be given to ensure that an authorized user is accessing the system. The system records information (on CO0:) about who is logging into the system and when the log-in occurred. When the user created LOGIN.CMD file exists, the system then executes the file from the log-in user file directory.

The system grants access to the terminal until a LOGOUT command is issued.

Format:

LOGIN [user-id [password]]

Command Qualifier:	Default:
None	

Prompts:

USERID? user-id

PASSWORD? password

Command Parameters:

user-id

The user-id is either an octal code or the logical name associated with the UIC. The forms of the user-id are:

1. [ggg,mmm]

   Using this form permits the system always to display the login text.
2. ggg,mmm

   Using this form permits the system always to display the login text.
3. ggg/mmm

   This form supresses the login text message after the first time you log in to the system during a 24-hour period.
4. name

   Using this form permits the system always to display the login text.

The group number, ggg, and the member number, mmm, are octal values from 1 to 377.

When name is specified, the effect is the same as the octal format.

password
A 1- to 6-character alphanumeric string. Associated with each user-id is a secret password. The correct password must be specified to gain access to the system. The password is not displayed when input in response to the PASSWORD? prompt.

Command Qualifier:

None

Example:

1. LOGIN JONES SAM
   GOOD MORNING
   05-Jul-78 11:56 LOGGED ON TERMINAL TT7:

   The user, Jones (with the password, Sam), requests access to the system. The system responds with a system identification followed by a greeting. The greeting is either GOOD MORNING, GOOD AFTERNOON, or GOOD EVENING depending upon the time of day.

11.25 LOGOUT

Terminates user access to the system. When you log out as a privileged user, the system does not abort privileged tasks nor release public resources. When logging out as a nonprivileged user, the system aborts tasks and releases allocated resources. The terminal then becomes available to other users.

Format:

LOGOUT

Command Qualifier:	Default:
None	

Prompt:

None

Command Parameter:

None

Command Qualifier:

None

Example:

1. LOGOUT
   HAVE A GOOD MORNING
   05-Jul-78 10:04 TT5: LOGGED OFF

   When a nonprivileged user issues the LOGOUT command, the system aborts active tasks initiated by the user, dismounts the user private volumes, and deallocates the user private devices.

   Depending on the time of day, the system displays:

   HAVE A GOOD MORNING
   HAVE A GOOD AFTERNOON
   HAVE A GOOD EVENING

11.26 MACRO

The MACRO command assembles one or more MACRO source files into a single relocatable binary object module.

Format:

MACRO [/qualifiers] file-spec [file-qualifiers] + . . .

Command Qualifiers:	Default:
/LIST [: list-file-spec]	
/NOLIST	/NOLIST
/OBJECT [: object-file-spec]	
/NOOBJECT	/OBJECT
/[NO]CROSS_REFERENCE	/NOCROSS_REFERENCE
/SWITCHES (: switch-list)	

File Qualifiers:	Default:
/PASS:n	
/LIBRARY	

Prompts:

FILE? file-spec [/file-qualifiers] + . . .

Command Parameter:

file-spec
Specifies a file that contains MACRO source code. Multiple input file-specifications must be concatenated with a plus (+) sign. Specifications must include a file name. If the file type is omitted, MAC is assumed, unless /LIBRARY is used, .MLB is assumed. No wildcards are allowed.

Command Qualifiers:

/NOLIST
Specifies that an assembly listing is not to be generated. This is the default.

/LIST [: list-file-spec]
Specifies that an assembly listing will be generated. If list-file-spec is given, then that file is not spooled; otherwise the listing is printed. The default file name is the name of the source file in the list, and the default file type is .LST.

/NOOBJECT
Specifies that an object module is not generated.

/OBJECT [: object-file-spec]
Specifies that an object module is to be generated. This is the default. The default name given to the object module file is taken from the last source file name in the list and is given the filetype .OBJ. This default name may be overridden by supplying the optional object-file-spec.

/[NO]CROSS_REFERENCE
/Specifies whether a cross reference listing is to be appended to the listing file. This implies use of the LIST qualifier. The default is /NOCROSS_ REFERENCE.

/SWITCHES (:switch-list)
Enables you to pass to MACRO the standard listing options you wish to use.

Switch-list has the form:

switch1:arg1 . . . switch:argn

If switch is /LI or /NL, the arguments are:

SEQ	LOC	BIX	BEX	SRC	COM
MD	MC	ME	MEB	CND	LD
TOC	SYM	TTM			

If switch is /EN or /DS, the argumets are:

ABS	AMA	CDR	FPT	GBL
LC	LSB	PNC	REG	

They are defined in the *TRAX MACRO Reference Manual.*

File Qualifiers:
The file qualifier may be one or both of these.

/PASS:n	Specifies that the file is only to be assembled during the pass specified (n may be either 1 or 2).
/LIBRARY	Specifies that the file is a macro library file. /LIBRARY is not allowed on the last source file in the list. The default file type is MLB.

NOTE

1. Library files must appear in a fixed order with respect to the source.

Example:
This command assembles the input files B, C, and D.MAC (using the necessary macros defined in A.MLB), creating the object module OBJMOD.OBJ and a listing file D.LST which will be spooled.

```
>MACRO/LIST/OBJECT:OBJMOD
FILE? A/LIBRARY+B+C+D
```

11.27 MERGE

The MERGE command merges records currently in one file with the records of another existing file; the receiving file must have relative or indexed organization.

Format:

```
MERGE [/LOG [:log-file-spec] input-file-spec [/qualifier] ]
      output-file-spec [/qualifier]
```

Command Qualifiers:	Default:
/LOG [log-file-spec]	See qualifier description.
Input-file Qualifier:	
/SEQUENTIAL	/SEQUENTIAL
/RELATIVE	
/INDEXED [/KEY:NUMBER:n]	
Output-file Qualifier:	
/INDEXED	One is required.
/RELATIVE	

Prompts:

FILE? Input-file [/qualifier]

INTO? output-file/qualifier

Command Parameters:

Input-file-spec
Specifies the file containing the records to be merged

Output-file-spec
Specifies the file that receives the new records.

Command Qualifier:

/LOG [:log-file-spec]
The qualifier is optional; if specified, a log of all error messages is created during the merge sequence. An example is a listing of all records taken from the input-file that could not be merged, due to a duplicated key being detected when duplicate keys are not supported. If the file specification is omitted, the file name defaults to T10: and the log is printed in the OUTPUT stream.

Input-File Qualifier:

/SEQUENTIAL
Specifies a sequential file. This is the default.

/RELATIVE
Specifies a relative file.

/INDEXED
Specifies an indexed file. If specified, the /KEY qualifier is also required.

/KEY:NUMBER:n
The order of the record extraction may be specified by the use of the KEY qualifier. This is meaningful only when the /INDEXED qualifier is used.

Output-File Qualifier:
An output-file qualifier is required.

/INDEXED
Specifies an indexed structured file.

/RELATIVE
Specifies a relative structure file.

NOTE

Wildcards are not permitted in either file specification parameter.

Example:
This example merges all records from the sequential file PROLL.1SEQ to the indexed file PROLL2.NDX.

```
>MERGE
FILE? PROLL1.SEQ
INTO? PROLL2.NDX/INDEXED
```

11.28 MESSAGE

Display a message at the specified support terminal(s). Two bells are sounded at the target terminal. The message is preceded by the date and the originating terminal number.

Nonprivileged users are restricted to sending messages to one terminal.

Format:

MESSAGE [message]

Command Qualifiers:	Default:
/ALL	/ALL
/TERMINAL:TTnn	
/LOG	

Prompt:

TERMINAL? TTnn

MESSAGE? message

Command Parameter:

message
The message is any combination of alphanumeric and control characters. Up to 68 characters can be specified. The string is terminated by the carriage return.

Command Qualifiers:

/ALL
Send the message to all the terminals.

/TERMINAL:TTnn
Send the message to terminal TTnn. The terminal must be logged into the system for the message to be displayed. The system ignores the command when the terminal is not logged into the system.

/LOG
Send the message to all logged-in terminals.

Examples:

1. MESSAGE/ALL SYSTEM WILL SHUT DOWN 4PM TO 6 PM

   On the receiving terminals the message appears as:

   05-Jul-78 10:13 FROM TT1: TO ALL
   SYSTEM WILL SHUT DOWN 4PM TO 6 PM

2. MESSAGE/TERMINAL:TT3 LOAD DISK1 ON DB3:

   On the receiving terminal the message appears as:

   05-Jul-78 10:55 FROM TT2: TO TT3:
   LOAD DISK1 ON DB3:

11.29 MOUNT

Use the MOUNT command to logically connect a volume to the file system. The system also ensures the device is on line. The system writes information on the volume permitting subsequent I/O access. Before each file access to the volume, the system verifies that I/O access is permitted.

Privileged users can mount volumes on public, nonpublic, and private devices; while nonprivileged users can mount a volume on devices that they allocate.

For efficient resource management, volumes should be dismounted when they are no longer needed. (Refer to the DISMOUNT command for its use.)

Format:

MOUNT device-name volume-id

Command Qualifiers:	Default:
/ACP:ddnACP	
/EXTENSION:blocks	blocks = 0
/PROTECTION:code	See Command Qualifier
/OVERRIDE:option(s)	
/OWNER:[ggg,mmm]	See Command Qualifier
/UNLOCKED	
/SHOW	
/WINDOW:m	m = 0
/DENSITY:bpi	bpi = 800

Prompts:

DEVICE? device-name

VOLUME LABEL? volume-id

Command Parameters:

device-name
The physical device name or logical name of the device to be mounted.

volume-id
The volume-id is a name associated with each volume. The volume-id is mandatory for magnetic tape and disk volumes unless the /OVERRIDE qualifier is specified. When the volume-id is specified, the system verifies the correct volume is used. The command is ignored when an incorrect volume-id is specified. The volume-id may be up to 6-alphanumeric characters for magnetic tape volumes and up to 12-alphanumeric characters for disk.

Command Qualifiers:

/ACP:DdnACP
The /ACP qualifier creates and assigns the specific ancillary control processor task to the disk drive, where:

Dd	is either DB, DM, or DR
n	is the unit number of the disk drive.

If you do not use the /ACP qualifier, the system uses the default F11ACP ancillary control processor to access the Ddn disk drive.

/EXTENSION:blocks
The /EXTENSION qualifier overrides the extension specified in the INITIALIZE command. If this qualifier is not specified, see the INITIALIZE/EXTENSION description for the default. The value of blocks is expected to be in decimal format.

/PROTECTION:code
Change the file and volume protection access. The codes consist of four categories and each category contains four specifiers. The categories are: SYSTEM, OWNER, GROUP, and WORLD. The specifiers are: R, W, E, and D for read, write, extend, and delete, respectively. The specifiers are positional. Thus,

```
/PROTECTION:(SYSTEM:RWED,OWNER:RWED,GROUP:R,
WORLD:R)
```

permits all access to the SYSTEM and the OWNER; while GROUP and WORLD are only permitted read access. The /PROTECTION:(OWNER:DEWR) command qualifier is not permitted since the order is incorrect.

When either /PROTECTION is not used or specific classes within the code are not given, the default values are taken from the volume. See the INITIALIZE command for the volume default values.

/OVERRIDE:option(s)
The /OVERRIDE qualifier permits several MOUNT options to be ignored. The options are separated by comma and enclosed in parentheses.

The options and the functions are:

IDENTIFICATION	Do not verify the volume label.

NOTE
When IDENTIFICATION is specified, no other /OVERRIDE option can be given.

LABEL	Do not verify the magnetic tape label.
EXPIRATION_DATE	Override the expiration date on the magnetic tape volume.

/OWNER:[ggg,mmm]
Change the user identification code (UIC) of the owner of the volume. The group number, ggg, and the member number, mmm, are octal values from 0 to 377. The square brackets, [ ], are required syntax.

/UNLOCKED
Permits read/write access to SY:[0,0] INDEXF.SYS file which is normally locked permitting read-only access.

/SHOW
Displays the volume information at the issuing terminal.

/WINDOW:m
Overrides the number of mapping pointers allocated for file windows set up at volume initialization. The range of m is from 0 to 9.

/DENSITY:bpi
Set the magnetic tape density to either 1600 or 800 bits per inch (bpi).

Examples:

1. MOUNT DB0: VOLNAME

   Mount the disk, DB0:, and verify that the volume name is VOLNAME.
2. MOUNT/PROTECTION:(SYSTEM:RW,GROUP:RWED) DB4: SYS001

   Mount disk, DB4:, and verify the volume name is SYS001. Permit the system read and write access and permit the group read, write, extend, and delete access. The owner and world access are unchanged.

11.30 $ON

The $ON command specifies an action to be taken if a subsequent batch command returns an error status equal to or greater than a specified level.

Format:

$ON status-level THEN action

Prompt: None.

Command Parameters:
Status-level is one of the following:

WARNING
ERROR
SEVERE ERROR

Action is one of the following:

CONTINUE
GOTO label
(Label is an alphanumeric string and must appear, together with a colon, in front of a subsequent command.)

STOP

Command Qualifiers: None

NOTES

1. $ON ERROR STOP is assumed by default at the beginning of a batch job.
2. The THEN action is taken if a subsequent command returns a status level equal to or greater than the status level specified in the $ON command.
3. An $ON command remains in force until superseded by another $ON command, until an ON condition is met, or until end-of-job, whichever occurs first.
4. A $ON command can be suspended by a $SET NOON command and can be later reinstated by a $SET ON command.

Example:
The $ON and $MACRO commands are executed. If the assembly is completed with nothing worse than a warning, the job proceeds to $LINK. If the linking is completed with nothing worse than a warning, the job proceeds to $RUN. If any of these commands produces a status-level of ERROR or SEVERE ERROR, the job is stopped; in this case, all remaining commands in the file are skipped.

```
$JOB
$ON ERROR STOP
$MACRO MYPROG
$LINK MYPROG
$RUN MYPROG
$EOJ
```

11.31 PRINT

The PRINT command causes one or more files to be printed on the line printer. It defines a printing job to be placed in the print queue.

Format:

PRINT [/qualifiers] file-spec [/qualifiers], . . .

Command Qualifiers:	Default:
/[NO] DELETE	/NODELETE
/COPIES:n	/COPIES:1
/QUEUE:queue-name	/QUEUE:PRINT
/UPPERCASE	/UPPERCASE
/LOWERCASE	
/[NO] ORIGINAL	/NOORIGINAL
/[NO] WIDE	/NOWIDE
/PAGES:n	Pages unlimited
/JOB:jobname	JOB: First six characters of first filespec
/PRIORITY:n	/PRI:50
/FORMS:n	/FORMS:0
/LENGTH:n	No implied form feeds
/[NO] RESTART	
/[NO] FLAG_PAGE	/NOFLAG_PAGE
/AFTER: (dd-mmm-yy hh:mm)	Present time

File Qualifiers:

/[NO] DELETE
/COPIES:n
/[NO] ORIGINAL

Prompt:

FILE; file-spec [/qualifier], . . .

Command Parameter:

file-spec
Specifies a file to be printed. If no file type is included in the file specification, the default file type is .LST.

Command Qualifiers:

/[NO] DELETE
Instructs the system to delete all files after printing. The default is /NODELETE.

/COPIES:n
Specifies the number of file copies to be printed. The value of n is an integer from 1 to 32 (decimal), with a default of 1.

/UPPERCASE
Specifies that an uppercase-only printer is sufficient for printing the job.

/LOWERCASE
Specifies that a printer capable of printing both upper and lower case characters is needed for this job.

/QUEUE:queue-name
Specifies the name of the queue in which the job is to be placed. If this qualifier is omitted, the job is placed in the queue named PRINT.

/[NO]WIDE
Specifies whether a wide printer is required. A wide printer has at least 132 characters per line.

/PAGES:n
Specifies the maximum number of pages that the job may produce. Default is unlimited.

/JOB:jobname
Specifies the name of the job to be placed in the queue. If omitted, the system will assign a job name based on the first six characters of the first file name.

/PRIORITY:n
Specifies the queue priority level of the print job. The argument n must be an integer in the range 1 to 250, and 250 is the highest priority.

/FORMS:n
Specifies the forms attribute of a print job. The FORM option complements the LENGTH option in defining the basic vertical boundaries and margins of an individual form.

The FORMS option indicates, directly or indirectly, the size of the form. Usually, this is the number of print lines between perforations. The default forms attribute is n=0, which indicates that form-feed processing will be handled by the printer. Values of n from 1 to 255 indicate that the software will handle form-feed processing. Values of n from 1 to 66 denote the actual number of lines by default, although they can be redefined by the installation. Values of n greater than 66 require installation definition.

/LENGTH:n
Specifies the number of lines that can be printed on a form page. If, while processing the print job, form-feed characters are not found in the file within n lines of the last form feed, a form feed is generated. Thus, if the FORMS option indicates a form size of 66 lines, and LENGTH specifies that no more than 60 lines may be printed per form page, the combination of options implies a bottom margin of six lines. By default, the system generates no form feeds; this is equivalent to specifying /LENGTH:0.

/[NO]ORIGINAL
Indicates whether or not to make temporary copies of files that exist on private volumes. /ORIGINAL requests that no copy be made.

/[NO]RESTART
Specifies whether a job can be restarted from the beginning following an interrupt, such as running out of paper.

/[NO]FLAG_PAGE
/NOFLAG_PAGE suppresses the flag page before each file in the job. /NOFLAG_PAGE is the default.

/AFTER: (dd-mmm-yy hh:mm)
Specifies the date and the time after which the job will become eligible for dispatching to some print processor.

File Qualifiers:

/[NO]DELETE
Specifies whether or not the file is to be deleted after printing.

/COPIES:n
Specifies the number of list copies to be produced for the file.

/[NO]ORIGINAL
Indicates whether or not to use a temporary copy of the file. /ORIGINAL directs that no copy be made.

NOTES

1. If /[NO]DELETE, /COPIES:n, or /[NO]ORIGINAL is specified as both a command qualifier and a file qualifier, the file qualifier overrides the command qualifier for that file. Any of these qualifiers given at command level sets a default qualifier that applies to all files unles overridden by a file qualifier.
2. The /QUEUE qualifier is optional. If it is omitted, the job is added to the default print queue, named PRINT.

Example:
The following command prints two copies each of every file in the directory having the file name RESULT. It also prints four copies of the file PRIME.DAT. The job is to be queued on the queue NIGHT and will not run until after 6 p.m.

```
>PRINT/COPIES:2/QUE:NIGHT
FILE? RESULT.*,PRIME.DAT/COPIES:4
```

11.32 PURGE

Use the PURGE command to remove the specified file(s) from the directory and release the occupied space. Before you can purge any file, you must have delete access to the file. See the DIRECTORY/FULL command to display file access rights and SET PROTECTION command to change the file access rights.

Format:

PURGE file-spec,...,file-spec

Command Qualifier:	Default:
/KEEP[:n]	n = 1

Prompt:

FILE? file-spec,...,file-spec

Command Parameter:

file-spec
The file specification must omit the version field.

Command Qualifier:

/KEEP[:in]
When the n qualifier is specified, the latest n versions are saved. The system locates the highest version number and calculates the remaining versions to keep. In making this calculation, the system assumes that the version numbers are downward contiguous. (That is, if the highest version is 17 and you wish to keep 3 files, the system keeps versions 17, 16, and 15, even if version 16 or 15 was previously removed. The default value of 1 is assigned when n is either not specified or you specify n as 0.

Examples:

1. PURGE TEST.TSK

   Purge all versions of TEST.TSK and retain the latest version.
2. PURGE/KEEP:3 ASDF.TMP

   Purge the ASDF.TMP files. The system retains the three highest numeric versions of the file when the volume contains three or more versions. When the volume contains less than three versions, they are all retained.

11.33 RENAME

The RENAME command renames an existing file.

Format:

RENAME old-file-spec new-file-spec

Prompts:

OLD? old-file-spec
NEW? new-file-spec

Command Parameters:

old-file-spec
Specifies an existing file. The file specification must include a file name and a file type.

new-file-spec
Specifies the new name for the existing file. The file specification must include a file name and a file type.

Command Qualifiers:

None

NOTES

1. Both specifications must have the same device (since files may not be renamed across devices).
2. Wildcards are allowed in the file type and file version fields of each file specification. Wildcards appearing in one file specification must appear in the corresponding fields of the other file specification.
3. If a version number is omitted from new-file-spec, the version number of the old-file-spec is used by default.
4. You can RENAME a file into another UFD, protection conditions permitting. As a general rule, if you are authorized to create a file in a UFD, you are also authorized to rename files stored under that UFD. Also, you can use RENAME to change the UFD of a file.

Examples:
This example renames the fourth version of the OLD.TMP to NEW.TMP.

```
>RENAME
OLD? OLD.TMP;4
NEW? NEW.TMP;1
```

11.34 RUN

The RUN command permits the system to install a task, run it, and when the task finishes, remove it from the system. If you wish to terminate an active task before it is completed, use either CTRL/Z for an interactive task or the ABORT command for any task. To display active task names, use the SHOW TASKS command.

Format:

RUN file-spec

Command Qualifier:	Default:
/TASK:task-name	See Command Qualifier

Prompt:

FILE? file-spec

Command Parameter:

file-spec
When the filename, in the file specification, is preceded by the $ symbol, the system searches the system file directory – [1,54] – for the file. When the filename is not preceded by either the $ symbol or a UFD, the system searches the default user file directory for the file. The default filetype is ".TSK".

Command Qualifier:

/TASK:task-name
Specifies a task name to assign when the system installs the task. The name may be up to 6-characters. The default name is TTnn where nn is the terminal number of the requesting terminal.

Examples:

1. RUN TSK1

   Install and execute the latest version of the task image file, TSK1.TSK, from the default user file directory. As mentioned earlier, the system names a task TTnn when you do not use the /TASK qualifier to assign a task name.
2. RUN/TASK:TEST DEMO.TSK

   Install and execute the latest version of the task image file, DEMO.TSK, from the default user file directory and assign it a task name of TEST.

11.35 SET

The SET command allows the user to alter dynamically a number of system-wide and local terminal characteristics. Nonprivileged users can alter the terminal characteristics of the terminal where they are logged in, the devices which they allocate, and local aspects of the operating environment. Privileged users can alter any terminal characteristic, device, and system-wide characteristic.

11.35.1 Enable/Disable Logins

Use the SET LOGINS command to enable/disable user access to the system. No one can log into the system after a privileged user issues a SET NOLOGINS command. Logging into the system is permitted after a privileged user issues the SET LOGINS command.

Format:

SET [NO] LOGINS

Command Qualifier: Default:

None

Prompt:

FUNCTION? [NO] LOGINS

Command Parameter:

None

Command Qualifier:

None

Examples:

1. SET NOLOGINS

   User access to the system is disabled until the SET LOGINS command is issued.
2. SET LOGINS

   User access to the system is now permitted.

11.35.2 File Protection

Change the protection access rights of the file(s) that you specify. Protection access rights can only be changed by the file owner or a privileged user. Display the protection access rights and file ownership with the DIRECTORY/FULL command.

Formats:

SET PROTECTION file-spec code
SET PROTECTION (file-spec,...,file-spec) code

Command Qualifier:	Default:
None	

Prompts:

FUNCTION? PROTECTION
FILE? file-spec
PROTECTION? code

Command Parameters:

file-spec
When more than one file specification is given, they must be enclosed within parentheses and delimited by a comma.

code
The code contains four groups: SYSTEM, OWNER, GROUP, and WORLD. Each group may possess four access rights: Read, Write, Extend, and Delete. (That is, to permit complete access to the world, specify "WORLD:RWED".) When a group is not specified, its access rights are not changed. When an access right is not specified, the group is not given that particular file access. For instance if you specify code as (SY:RWED,GR:RWED,OW:RWED,WO:RWED), full access is granted to all users.

NOTE
Full access is granted to all users when only the world is specified as: WORLD:RWED.

Command Qualifier:

None

Examples:

1. SET PROTECTION A.TMP SYSTEM:RWED

   Alter the system access rights for the file, A.TMP, to permit read, write, extend, and delete access. The group, owner, and world access rights are unchanged.
2. SET PROTECTION A.TMP SYSTEM:RWED,GROUP:R

   Alter the system access rights of the file, A.TMP, to permit read, write, extend, and delete access to the system and read access to the group. The owner and world access rights are unchanged.

11.35.3 Default Device and Directory

Change the default for either the user file directory code or the system device at the issuing terminal. The defaults can be displayed with the SHOW DEFAULT command. Use the SHOW DEFAULT command to display the current defaults.

Format:

SET DEFAULT [device-name] [[ufd]]

Command Qualifier: Default:

None

Prompts:

FUNCTION? DEFAULT
DEVICE AND/OR DIRECTORY? device-name [ufd]

Command Parameters:

device-name
The name of the device to be specified as the default system device.

ufd
The user file directory is specified in the format:

[ggg,mmm]

where, the group number, ggg, and the member number, mmm, are octal values from 0 to 377.

Command Qualifier:

None

Example:

1. SET DEF DB1:

   The default system device is changed to DB1: on the issuing terminal.

11.35.4 Device Attributes

The SET DEVICE command permits changing various attributes, such as device buffer width. Use the SHOW DEVICES command to display device attributes.

Privileged users can change characteristics for any device, while non-privileged users can only change devices that they allocate.

Format:

SET DEVICE:device-name option

Command Qualifier: Default:
None

Prompts:

FUNCTION? DEVICE
DEVICE? device-name
ATTRIBUTE? option

Command Parameters:

device-name
The name of the device to be set.

option
The option can be any one of the following items:

WIDTH:value
Applies to terminals and printers, specifying the buffer width. The format of value is either octal or decimal. Specify decimal in the nn format that is, 72 is octal; while 72. is a decimal.
[NO]WRITECHECK
Some devices, such as disk and magnetic tape, permit a read after write to verify the write phase. Specifying WRITECHECK, permits this verification.
[NO]PUBLIC
When a device is to be shared, the device is specified as PUBLIC. A public device cannot be allocated.

Command Qualifier:

None

Examples:

1. SET DEVICE:LP0: WIDTH:132.
   Set the buffer size of LP0: to 132(10) characters.
2. SET DEVICE:DB0: NOWRITE
   Disable write checking on DB0.

11.35.5 Terminal Attributes

Change terminal characteristics.

Format:

SET TERMINAL [:terminal-name] option

Command Qualifier:	Default:
None	

Prompts:

FUNCTION? TERMINAL
ATTRIBUTE? option

Command Parameters:

terminal-name
The name of the terminal to be set. When no terminal name is specified, TI0: is assumed. Specify the terminal name in the format, TTnn: (where nn is the terminal number).

Privileged users can change any terminal, while nonprivileged users can only change terminals that they allocate.

option
The option can be any one of the following items:

TYPE:term
Specify the type of terminal. Where term can be either SCOPE or NOSCOPE.
LOWERCASE
Recognize lowercase and uppercase characters on input.
UPPERCASE
Recognize uppercase characters. Lowercase characters are converted to uppercase on input.
[NO]PRIVILEGED
Establish the privilege status of the terminal. A privileged terminal is permitted to use privileged commands.
[NO]SLAVE
Establish the slave status of the terminal. A slaved terminal cannot enter unsolicited input. Terminal input to the system is recognized when requested from the task.
[NO]REMOTE
The option declares the terminal is connected to a modem for access through dial-up lines.

[NO]HOLD
Establish the terminal hold screen mode. When in hold screen mode, the terminal displays a full screen of data each time the scroll key is pressed. This is useful when displaying a file on a scope terminal.
[NO]ESCAPE
Either enable or disable the recognition of the escape sequence.
SPEED:(xmt,rec)
Set the terminal transmit and receive baud rates to the same speed. The terms transmit and receive are in reference to the ability of the terminal to transmit and receive. In addition to informing the system, the terminal must be set to the specified baud rates. Although the system recognizes baud rates of: 110, 150, 300, 600, 1200, 2400, 4800, and 9600, certain speeds and combinations of speeds are not permitted on each terminal.

The parentheses and comma are part of the required format.

Command Qualifier:

None

Examples:

1. SET TERMINAL:TT3: TYPE:SCOPE

   Sets terminal 3 to respond as a scope.
2. SET TERMINAL:TT4: SPEED:(300,300)

   Sets terminal 4 to transmit at 300 baud and receive at 300 baud.

11.35.6 Print and Batch Queue/Job Status

Use the SET command to modify print and batch job attributes. The attributes are initially assigned by the PRINT and SUBMIT commands.

Format:

SET QUEUE queue-name option[,...,option]
SET QUEUE ENTRY:(n,m) option[,...,option]

Command Qualifier:	Default:
None	

Prompt:

QUEUE NAME OR ENTRY? queue-name
OPTIONS? option

Command Parameters:

queue-name
The queue name can be either the name of a queue or an entry number. If you specify a queue name, such as PRINT, you must also specify the JOB option to define the job uniquely.

The job entry is identified in the queue file as a pair of numbers separated by a comma and enclosed in parentheses. Since the system uses ENTRY as a queue name, you must specify the entire word. The system accepts an abbreviated form of ENTRY as another queue name.

Job entry identifiers are displayed when you request a full listing with the SHOW QUEUE command.

option
The option can be any one or more of the following items:

JOB:[ggg,mmm]job-name
Use this option to specify the job to be affected. When ENTRY is specified, this option is not permitted.
UPPERCASE
Specify that the print job requires only a printer with an uppercase character set.
LOWERCASE
Specify that the print job requires an uppercase and lowercase printer.
[NO]WIDE
The WIDE option indicates a 132 character printer is to be used. The NOWIDE option indicates an 80 character printer is to be used.
PAGES:n
Specifies that the print job should be aborted if it exceeds n pages.
PRIORITY:n
Change the queue priority of the job to n; where n can be from 0 to 250.
FORMS:n
Specifies that the print job is to be dispatched to a printer processing FORM n.
LENGTH:n
Specifies to the system to generate a form feed character if one is not found within n lines of the most recent form-feed character.
[NO]RESTART
Specifies whether the job is restartable from the beginning if the job is stopped.
[NO]FLAG_PAGE
Specifies whether to precede the files in a print job with flag pages.
HOLD
Place the job in the hold list. The job remains in the hold list until explicitly released, even though other conditions would indicate that it should be processed.

RELEASE
Place the job in the waiting job list. The job is processed next when the job reaches the top of the waiting job list.
AFTER:(dd-mmm-yy hh:mm)
Make the job eligible for processing after the specified date and time are reached. Where:

dd	is the day of the month;
mmm	is the 3-letter abbreviation of the month;
yy	is the last 2 digits of the year;
hh	is the hour, based on a 24-hour clock;
mm	is the minutes.

Command Qualifier:

None

Examples:

1. SET QUEUE BATCH JOB:[1,1]TEST,PRIORITY:120
   The batch job, [1,1]TEST, is given a queue priority of 120.
2. SET QUEUE ENTRY:(112,756) FORMS:1
   Change the queue entry, 112,756, to print form 1.

11.36 SHOW

The SHOW commands display various system-wide information at the requesting terminal.

11.36.1 Time of Day and Date

Displays the time of day and date.

Format:

SHOW TIME

Command Qualifier:	Default:
None	

Prompt:

FUNCTION? TIME

Command Parameter:

None

Example:

1. SHOW TIME
   09:34:55. 05-Jul-78

11.36.2 Default Device and Directory

Display the default system device name and user file directory.

Format:

SHOW DEFAULT

Command Qualifier:	Default:
None	

Prompt:

FUNCTION? DEFAULT

Command Parameter:

None

Command Qualifier:

None

Example:

1. SHOW DEFAULT
   DB0: [27,200]

   The user default system device is the DB0: disk and the default user file directory is [27,200].

11.36.3 Device Assignments

Display either the global or local assignments.

Format:

SHOW ASSIGNMENTS [:range]

Command Qualifier:	Default:
None	

Prompt:

FUNCTION? ASSIGNMENTS

Command Parameter:

range
The range can be either GLOBAL or LOCAL. When neither is specified, LOCAL is assumed. Specifying GLOBAL displays local and global assignments, while LOCAL displays the terminal, TTnn:, and the LOGIN assignments.

Command Qualifier:

None

Example:

1. SHOW ASSIGNMENTS

   ```
   SY0:    SY0:    LOGIN T10: – TT2:
   US1:    MM0:    LOCAL T10: – TT2:
   ```

   The command was entered on the TT2: terminal. The SY0: logical name is assigned to the system device – SY0: – from which the system is booted. The system device, SY0:, is also a logical name assigned when the user logged in to the system. (The SHOW DEVICES command displays the SY0: login logical name assigned to a physical device, such as DB0:).

   The US1: logical name is assigned to the MM0: magnetic tape as a local assignment.

2. SHOW ASSIGN: GLOBAL

   ```
   US1:    DP0:    LOCAL TI0: – TT2
   TO0:    MM0:    LOCAL TI0: – TT33
   SY0:    SY0;    LOGIN TI0: – TT2:
   SY0:    SY0:    LOGIN TI0: – TT16:
   SY0:    SY0:    LOGIN TI0: – TT31:
   SY0:    SY0:    LOGIN TI0: – TT35:
   SY0:    SY0:    LOGIN TI0: – TT45:
   SY0:    SY0:    LOGIN TI0: – TT40:
   MP0:    DB0:    GLOBAL
   IN0:    DB0:    GLOBAL
   QU0:    DB0:    GLOBAL
   ```

11.36.4 Device Attributes

Display the symbolic names and status of the devices. The display is made on the entering terminal.

Format:

SHOW DEVICES [option]

Command Qualifier:	Default:
None	

Prompt:

FUNCTION? DEVICES

Command Parameter:

option
The option can be any of the following words. The system displays all the options when no option is specified.

[NO] PUBLIC
Display only those devices allocated as either PUBLIC or NOPUBLIC

TYPE device-name:
Display only the devices specified by the device name, such as DB:.

[NO] WRITECHECK
Display only those devices in either the WRITECHECK or NOWRITE-CHECK mode.

WIDTH: device-name
Display the buffer size of the specified device.

Command Qualifier:

None

Examples:

1. SHOW DEVICE WIDTH:LP0:
   BUF = LP0:00132.

   Displays the buffer size of LP0: as 132 decimal characters.
2. SHOW DEVICES

```
DB0:    PUBLIC MOUNTED LOADED
DB1:    PUBLIC LOADED
DB2:    LOADED
MM0:    OFFLINE UNLOADED
MM1:    LOADED
LP0:    DB0: SPOOLED LOADED
LP1:    OFFLINE LOADED
TT0:    LOADED
TT1:    [22,101] – LOGGED ON
TT2:    LOADED
TT3:    LOADED
TT4:    LOADED
CO0:    TT0:
CL0:    LP0:
SP0:    DB0:
LB0:    DB0:
SY0:    DB0:
```

The device names in the left-hand column can be either a logical name or a physical device name. Beginning with TI0:, the list at the bottom of the SHOW DEVICES example displays the standard logical names used by some system tasks. The typical usage of each of these logical names is:

TI0:	terminal input
CO0:	console output
CL0:	console listing
SP0:	spooling
LB0:	library input
SY0:	system input/output device

The following definitions describe the messages that appear in the right column. More than one message can appear on the same line. A device name in the left column with no messages in the right column indicates that, although the system tables contain entries for this device, the host configuration does not contain the related device driver. Also, a device name in the right column with no other messages indicates a redirected device such as:

SY0: DB0:

1. MOUNTED
   The MOUNTED message indicates that a volume is logically connected to the file system.
2. LABEL = name
   The LABEL = name message indicates the mounted volume identification when the SHOW DEVICES command is specified on a privileged terminal. The message is omitted on non-privileged terminals.
3. PUBLIC
   The PUBLIC message indicates that the device is a shared device that you cannot allocate for private use.
4. MARKED FOR DISMOUNT
   The MARKED FOR DISMOUNT message indicates that the system dismounts the volume when the system completes the current file access.
5. OFFLINE
   The OFFLINE message indicates that the device cannot be selected by the system.
6. [uic] LOGGED ON
   The [uic] LOGGED ON message indicates that the user identified by [uic] is logged into the system at this terminal.
7. LOADED
   The LOADED message indicates that the device drive is loaded and the device is available for access.
8. UNLOADED
   The UNLOADED message indicates that the device driver is loadable, but not currently loaded.

9. SPOOLED
   The SPOOLED message indicates the device in the left column is a spooled device. When you output files from a private device to a spooled device, the system temporarily stores the files on the device specified in the right column. The system transfers the files to the spooled device when it is available.
10. PRIVATE
    The PRIVATE message indicates that the device in the left column is allocated to the user who logged into the terminal associated with the message.

11.36.5 Terminal Attributes

Display a list of terminals with the specified characteristic.

Format:

SHOW TERMINAL option

Command Qualifier: Default:

None

Prompts:

FUNCTION? TERMINAL

ATTRIBUTE? option

Command Parameter:

option
The option can be any one of the following items:

TYPE: term
Specify the type of terminal. Where term can be either SCOPE or NOSCOPE.

LOWERCASE
Display those terminals that recognize uppercase characters.

UPPERCASE
Display those terminals that recognize lowercase and uppercase characters.

[NO] PRIVILEGED
Display the terminals in the privileged mode.

[NO] SLAVE
Display the terminals in the slave mode.

[NO] REMOTE
Display those terminals in the remote mode.

[NO] HOLD
Display the terminals in the hold screen mode. When in hold screen mode, the terminal displays a full screen of data each time the scroll key is pressed. This is useful when displaying a file on a scope terminal.

[NO] ESCAPE
Display those terminals in the escape sequence mode.

SPEED: Tnn
Display in the transmit:receive format the terminal that you specify. The terms transmit and receive refer to the ability of the terminal to transmit and receive.

When Tnn is not specified, the speed of the issuing terminal (TI0:) is shown.

Command Qualifier:

None

Examples:

1. SHOW TERMINAL SPEED:TT1:
   SPEED = TT1:2400:2400

   Display the transmit and receive baud rates of terminal 1.
2. SHOW TERMINAL TYPE:SCOPE

   CRT = TT1:
   CRT = TT2:
   CRT = TT3:
   CRT = TT4:
   CRT = TT5:
   CRT = TT6:

   List all SCOPE or cathode ray terminals.

11.36.6 Partitions
Display a description of each memory partition on the requesting terminal. The information shown includes:

- Partition name
- Partition base address (octal)
- Partition size (octal)
- Partition kind (main partition or subpartition)
- Partition type:

TASK	for user-controlled partition
COM	for common partition (for re-entrant libraries and common data areas)
DEV	to allow tasks to communicate with specific device registers
SYS	for system-controlled partition
(taskname)	for active tasks
DYNAMIC	for dynamically created region, or
DRIVER	for region occupied by a loaded device driver.

Format:

SHOW PARTITIONS

Command Qualifier:	Default:
None	

Prompt:

FUNCTION? PARTITIONS

Command Parameter:

None

Command Qualifier:

None

Example:

1. SHOW PARTITIONS

LDR	000000	000000	MAIN	TASK
SYSPAR	120000	010000	MAIN	TASK
FCPPAR	130000	026000	MAIN	TASK
PMDPAR	156000	020000	MAIN	TASK
SPLPAR	156000	010000	SUB	TASK
DRVPAR	176000	014000	MAIN	TASK
	176000	001600	SUB	DRIVER – DB:
	177600	001000	SUB	DRIVER – LP:
	200600	003100	SUB	DRIVER – MM:
GEN	212000	546000	MAIN	SYS
	212000	013400	SUB	(RMDEMO)
	225400	045700	SUB	(...EDI)
	302100	034200	SUB	(...AT.)
	336300	117100	SUB	(...TKB)
	455400	010000	SUB	(...SYS)
	465400	124100	SUB	(F11ACP)

11.36.7 Memory

Displays the memory allocation for the partition.

Formats:

SHOW MEMORY MAIN:part-name

SHOW MEMORY SUBPARTITION:part-name

Command Qualifier: Default:

None

Prompts:

MAIN or SUBPARTITION? MAIN or SUBPARTITION

MAIN PARTITION NAME? part-name

PARTITION NAME? part-name

Command Parameters:

MAIN:part-name
Show the main partition. The partition name is a 1- to 6-alphabetic character string.

SUBPARTITION:part-name
Show the subpartition. The partition name is a 1- to 6-alphabetic character string.

Command Qualifier:

None

Example:

1. SHOW MEMORY MAIN:SYSPAR
   MAIN = SYSPAR:001200:000100:TASK
   Show the user-controlled main partition called SYSPAR.

11.36.8 Task Status

Display information about installed tasks on your terminal, TI0:. The information displayed is dependent upon the option selected.

Format:

SHOW TASKS status [display] [task-name]

Command Qualifier:	Default:
None	

Prompts:

FUNCTION? TASKS

ACTIVE OR INSTALLED? status

Command Parameters:

status
Specify either ACTIVE or INSTALLED. When you specify INSTALLED, the system displays both active and dormant tasks.

display
You can specify either: BRIEF, ALL, or FULL. If you specify ACTIVE and BRIEF, the default value, the system displays a list of the active task names executed from your terminal. If you specify ACTIVE and ALL, the system displays a list of all the active task names in the system.

If you specify INSTALLED and BRIEF, the system makes a brief display of all the installed task names. If you specify INSTALLED and FULL the system makes a full display of all the active and dormant tasks.

The FULL display contains the following information for each task:

- Task name
- Task control block physical address (octal)
- Partition name
- Partition control block physical address (octal)
- Partition base and limit physical addresses (octal)
- Running priority and default priority of the task
- Task status flags
- TI0: terminal physical device name
- I/O count (decimal)
- Task local event flags, and
- Task registers and processor status word (memory resident tasks only).

Flags prefixed by a minus (-) sign indicate the complementary status. That is, -CHK indicates that the task is not checkpointable.

When a task is not in memory (the OUT flag is displayed), the contents of the PC, PS, and the registers are not displayed.

STATUS	DESCRIPTION
ABO	Task is being aborted
ACP	Task is an ancillary control processor
AST	Task is processing a task-controlled software interrupt
BFX	Task is being fixed in memory
CAF	Checkpoint space allocation failure occurred
CAL	Checkpoint space is allocated in task image
CHK	Task is checkpointable
CKD	Task checkpointing is disabled
CKP	Task is checkpointed
CKR	Task checkpoint request pending
DST	AST of the task is disabled
EXE	Task is in execution
FXD	Task is fixed in memory
HLT	Task is being terminated
MCR	Task was activated by MCR
MSG	Task aborted and waiting for TKTN message
NRP	Task is mapped to nonresident partition
NSD	Task cannot receive data (no send data allowed)
PMD	Suppress task post-mortem dump on abort
OUT	Task is out of memory
PRV	Task is privileged

STATUS	DESCRIPTION
RDN	Task I/O is being run down
REM	Task is to be removed on exit
ROV	Task has resident overlays
SLV	Task is slave
SPN	Task is being suspended
SPNA	Task was suspended prior to AST
STP	Task stopped for terminal input
STPA	Task stopped prior to AST
TIO	Task is waiting for terminal input
WFR	Task is in a wait-for state
WFRA	Task was in a wait-for state before AST

task-name
When the task name is specified, only that task information is displayed, When the task name is omitted, all tasks are displayed.

Command Qualifier:

None

Examples:

1. SHOW TASKS ACTIVE BRIEF
   EDIT2
   . . . DCL
   AT.T2

   The SHOW command is issued from terminal TT2: and is a display of the active tasks initiated from terminal TT2.
2. SHOW TASKS ACTIVE ALL
   MTAACP
   F11ACP
   DB1ACP
   QMG
   LPP0
   MACT3
   EDIT4
   EDIT3
   TT2
   AT.T6

   The SHOW command is issued from any support terminal and is a display of all the active tasks in the system.

11.36.9 Print and Batch Queue/Job Status

Use the SHOW command to display print and batch job entry attributes. The attributes are initially assigned by the PRINT and SUBMIT commands.

Format:

SHOW QUEUE queue-name [option1] [option2] [option3]

SHOW QUEUE ENTRY: (n, m) [option1] [option2] [option3]

SHOW QUEUE ALL [option1] [option2] [option3]

Command Qualifier:	Default:
None	

Prompt:

None

Command Parameters:

queue-name
You can specify ALL, ENTRY, or a queue name.

Using a queue name, such as PRINT or BATCH, requests the system to display job entries which you further define with the options.

ALL
Requests the system to display job entries in all the queues which you further define with the options.

ENTRY:(n, m)
Requests the system to display a specific job entry where the values of n and m are unique values assigned to the job entry by the queue manager. Since the system uses ENTRY as a queue name, you must specify the entire word. The system accepts an abbreviated form of ENTRY as another queue name.

option1
Option1 further defines the jobs to display. The value of option1 can be:

BATCH
Use the option in conjunction with the queue name of ALL to display all the batch queue(s).

PRINT
Use this option in conjunction with the queue name of ALL to display all the print queue(s). This option is the default option when you do not specify either PRINT or BATCH.

ALL
Display all the entries in the specified queue.

option2
Option2 further defines the jobs to display. The value of option2 can be:

JOB:[[ggg,mmm]] job-name
Use the option to specify the job(s) to be displayed. When ENTRY is specified, the option is not permitted. If option1 is omitted, the system assumes the PRINT value.

USER:[ggg,mmm]
Display the queue entries for the specified user identification code [ggg,mmm] which defaults to your login UIC. If option1 is omitted, the system assumes the ALL value.

PRIORITY:n
Display the jobs at the specified priority level. The value of n can be from 1 to 250. If option1 is omitted, the system assumes the ALL value.

FORMS:n
Display the entries using the specified forms "n" attribute. If option1 is omitted, the system assumes the ALL value.

NUMBER
Display the number of entries in the queue(s). If option1 is omitted, the system assumes the ALL value.

option3
Option3 further defines the jobs to display. The value of option3 can be:

BRIEF
Display a summary report. This is the default option when you do not specify FULL or BRIEF.

FULL
Display a full report.

Command Qualifier:

None

Examples:

1. SHOW QUEUE BATCH NUMBER

   The number of entries in the batch queue, BATCH, are displayed.
2. SHOW QUEUE ENTRY:(112,756)

   Display the queue entry, (112,756).

11.37 SORT

The SORT command invokes the SORT utility of TRAX. This utility sorts the contents of an input file into a sequence indicated by the SORT command, and writes the sorted contents into an output file.

Two types of SORT are allowed, a record sort or a tag sort.

A record sort produces a reordered file by examining the specified control keys and directly copying entire records to the output file as required.

A tag sort produces a reordered file by extracting the control keys into the proper order. Then the record pointer associated with each key is used to reaccess the input file randomly to produce the sorted output file. The tag sort is possible only when the input file resides on a disk.

See the *TRAX SORT Reference Manual* for further details.

Format:

SORT [/qualifiers] input-file-spec [/file qualifiers]

Command Qualifiers:	Default:
/ALLOCATION:n	See qualifier
/BLOCKSIZE:n	/BLOCKSIZE:512
/BUCKETSIZE:n	/BUCKETSIZE:1
/[NO]CONTIGUOUS	/NOCONTIGUOUS
/DEVICE:device-name	See qualifier
/FILES:n	/FILES:5
/FORMAT:type:n	See qualifier
/KEYS:(abm.n)	See qualifier
/PROCESS:process-type	PROCESS:RECORD
/RELATIVE	
/SEQUENTIAL	/SEQUENTIAL
/SIZE:n	See qualifier
/SPECIFICATION:file-spec	See qualifier
/OUTPUT:file-spec	See qualifier
Input File Qualifier:	
/FORMAT:type	Required
/INDEXED:n	

Prompt:

FILE? input-file-spec [/file-qualifiers]

Command Parameters:

input-file-spec
Specifies the file whose contents are to be sorted. If no file type is given, .DAT is the default file type.

Command Qualifiers:

/ALLOCATION:n
Specifies the initial disk space allocation for the output file. Legal values range from 0 to 65535 (bytes). Output file allocation defaults to the input file size.

/BLOCKSIZE:n
Specifies the blocksize in bytes for any magnetic tape files that may be involved in the sort. This qualifier is valid for magnetic tapes only. The default is a 512 byte block.

/BUCKETSIZE:n
Specifies the number of 512 byte blocks per bucket in a disk output file. If this qualifier is used, the block size is 512 bytes, regardless of any /BLOCKSIZE specification. If the input and output files have the same organization, the output file defaults to the same bucketsize as for the input file. Otherwise the default bucketsize is 1.

/[NO]CONTIGUOUS
Specifies whether the disk output file allocation must be contiguous or not. In a contiguous file, each successive block is physically located between its logical predecessor and its logical successor with no filler or extraneous material separating the blocks. /NOCONTIGUOUS is the default.

/DEVICE: device-name
Specifies the device to be associated with the intermediate scratch files of the sort. This qualifier overrides device specifications for scratch files resulting from task build options.

See also the description of the /FILES qualifier that follows, and refer to the *TRAX SORT Reference Manual* for detailed information about scratch files and their use.

/FILES
Specifies the maximum number of intermediate scratch files. Default is 5. See the *TRAX SORT Reference Manual* for detailed information on scratch files.

/FORMAT:type:n
Specifies the record format and maximum record size of a file. If /FORMAT appears in the command prior to the input file specification, as a command qualifier, it qualifies the output file specification only.

The /FORMAT qualifier must always be present in the command as an input file qualifier. If /FORMAT is omitted as an output file qualifier, the /FORMAT for the input file applies also to the output file.

The type argument can be any of the following:

FIXED
STREAM
VARIABLE
UNKNOWN

The record size n is the exact record size in bytes for FIXED records, and the maximum record size in bytes for other record formats. Record size may be omitted for output files.

The output record format defaults to that of the input file.

/KEYS=(abm.n, . . .)
Specifies the key fields to control the record sequence of the output file. Up to 10 key fields, separated by commas, are allowed. The entire list of key descriptions must be enclosed in parentheses.

Each field description sequence abm.m breaks down as follows:

1. Specifies how the data shall be treated. Legal values of their interpretations are:
   - B twos complement binary
   - C alphanumeric (this is the default)
   - D One of the following:
     a. if the characters are alphabetic, numeric with the sign superimposed over the units digit, or certain slashes (/), use the value of the digits group. Here are two examples and their values:

        A2CD5 = (+) 12345 A/47J = (-) 11471

     b. if the characters represent a standard FORTRAN IV number, such as 12, -35, 42.98 or -0.76E+3, convert the number to binary for storage or evaluation
   - F 1- or 4-word floating point binary
   - I same as D, but with the sign leading and separate, so that the first byte of the field is a + or -
   - J same as I but with the sign trailing and separate

K same as D but with the sign leading and over-punched (54321, for instance, if positive, would come out as 5432A. The negative 54321 would be 5432J.)
P packed decimal format
Z ASCII zone

2. Defines the general sort order. The default is N (ascending order).
N ascending order
O opposite or descending order
m is a decimal number giving the first byte of the key field. Number from the first byte of the record which is byte 1. This item must be present.
n is a decimal number giving the length of the key field in bytes. This item must be present.

The default abm.n value is

a = C
b = N
m = first position of the field
n = length of field

/OUTPUT:file-spec
Specifies the file that will receive the sorted records. Default is the input file.

/PROCESS:process-type
Specifies which sorting process shall be used. This qualifier is illegal when the /SPECIFICATION qualifier is present. The process-type argument has two possible values:

RECORD
TAG

RECORD specifies a record sort. It produces a reordered file by directly transferring the entire record contents on examination of the record keys. This is the default.

TAG specifies a tag sort. It produces a reordered file in two stages. First, the key fields from the various records are sorted and given record pointers. Then the sorted file is created by using the sorted record pointers to access the input file records and create the full sorted file.

/RELATIVE
Specifies the organization of the output file. /SEQUENTIAL is the default.

/SIZE:n
Specifies the size of the retrieval window. The value n corresponds to the pack default set up by the /WINDOW qualifier on the INITIALIZE or MOUNT command.

/SPECIFICATION:file-spec
Specifies a file containing a set of controls for the sorting process. The /SPECIFICATION qualifier takes the place of /KEY and /PROCESS qualifiers, and offers greater flexibility in sorting files of non-uniform format.

The specification file includes the following controls.

Record selection
Alternative collating sequence
Forced keys
Variable input format
Variable output format
Process selection

The detailed description of the specification file is beyond the scope of this manual. See the *TRAX SORT Reference Manual* for further information.

Input File Qualifiers:

/FORMAT:type:n
Specifies the record format and maximum record size for the input file. This file qualifier is required.

The type of argument can be any of the following:

FIXED
VARIABLE
UNKNOWN

The n argument gives the exact record size in bytes for FIXED files, on the maximum record size for other record formats.

/INDEXED:n
Specifies indexed sequential file organization for the input file, and gives the number of access keys, n, defined for that file.

NOTES

1. Either a /KEYS or /SPECIFICATION is required in the SORT command.
2. The qualifiers /KEYS or /PROCESS must not be used if /SPECIFICATION is used.

Example:
The file DATA.DAT consists of variable length records, with no record longer than 80 bytes. A RECORD sort is performed, because /PROCESS= RECORD is the default. The sort key begins in record position 1 and is 4 bytes long. An alphanumeric ascending sort is performed. The output is placed in the next higher version of DATA.DAT.

```
>SORT/KEYS:(1,4) DATA/FORMAT:VARIABLE:80
```

11.38 START

Use the START command to start the queue manager, queue, or processor.

11.38.1 Starting a Queue or Processor

Use the privileged START command to restart a queue or processor. Queues and processors are first started when they are initialized. They can be halted with the STOP command. The START command does not affect any assignments.

Format:

START name

Command Qualifiers:	Default:
/QUEUE	
/PROCESSOR	

Parameter Qualifiers:	Default:
/ALIGN	
/FORMS:n	
/FLAG:n	
/CONTINUE	/CONTINUE
/NEXT_JOB	
/BACKSPACE:n	
/RESTART	
/TOP_OF_FILE	
/AT_PAGE:n	

Prompt:

None

Command Parameters:

name

The queue or processor to be started is specified by name. A queue name may be from 1- to 6-characters and must be unique in the queue file. You must specify the processor name using the following format:

ssPnn

where:

1. The two letters, ss, are either LP for a print processor or BA for a batch processor

2. The two letters, nn, are either a 1- or 2-digit octal number.

   For a print processor, the value of nn corresponds to the physical device name of the printer, such as LPP0 for the LP0: line printer.

   For a batch processor, the value of nn is a unique octal number, such as BAP0 for the first batch processor.

Command Qualifiers:

/QUEUE
Use the /QUEUE qualifier to restart the queue, name. The jobs in the queue become eligible for processing by any assigned processor.

/PROCESSOR
Use the /PROCESSOR qualifier to restart the print or batch processor, name. You must also consider the job to be processed. You can continue with the current job (the active job when the processor was stopped), restart the current job, or hold the current job and process the next job.

Parameter Qualifiers:

/ALIGN
The print processor is to assume the printer paper is at the top of the page.

/FORMS:n
The print processor is to process a new form. The qualifier also invokes the /NEXT_JOB qualifier. The current job, if there is one, is placed in the held jobs list. The value of n can be from 0 to 255(10).

/FLAG:n
The print processor is to print n flag pages at the beginning of a job and the files within the job. The qualifier also invokes the /NEXT_JOB qualifier. The current job, if there is one, is placed in the held jobs list.

/CONTINUE
Restart the print or batch processor where it was stopped. If there is a current job, continue the current job where it was stopped.

/NEXT_JOB
Restart the print or batch processor with the next job in the queue. The current job, if there is one, is placed in the held jobs list.

/BACKSPACE:n
Restart the print processor n pages backward and continue the current job. A page in a print file is indicated by the presence of a form feed character.

/FORWARDSPACE:n
Restart the print processor n pages forward and continue the current job. A page in a print file is indicated by the presence of a form feed character.

/RESTART
Restart the print or batch processor at the beginning of the current job.

/TOP_OF_FILE
Restart the print or batch processor at the beginning of the current file.

/AT_PAGE:n
Restart the print processor at the specified page (n) in the current file. A page in a print file is indicated by the presence of a form feed character.

Examples:

1. START/QUEUE BATCH

   The batch queue, BATCH, is restarted. There is no change to assignments. The processors assigned to the BATCH queue can now receive jobs from this queue.
2. START/PROCESSOR LPP0/NEXT

   The print processor, LPP0, is restarted at the beginning of the next job.

11.38.2 Starting the Queue Manager

Use the START command to re-enable queueing operations in the system. No other queue-related commands can be used until this command is specified. The queue manager is initially started when the system image is booted into the system. You can stop the queue manager with the STOP command.

Format:

START/QUEUE/MANAGER

Prompt:

None

Command Parameter:

None

Example:

1. START/QUEUE/MANAGER

   Restart the queue manager who was halted with the STOP command.

11.39 STOP

11.39.1 Stopping a Queue or Processor

Use the privileged STOP command to halt a queue or processor indicating that no jobs can be passed to it. The assignments are not affected.

Format:

STOP name

Command Qualifier:	Default:
/QUEUE	
/PROCESSOR	

Parameter Qualifiers:	Default:
/PAUSE	/PAUSE
/FILE_END	
/JOB_END	
/ABORT	

Prompt:

None

Command Parameters:

name

The name of the queue or processor to be stopped. A queue name may be from 1- to 6-characters and must be unique in the queue file. You must specify the processor name using the following format:

ssPnn

where:

1. The two letters, ss, are either LP for a print processor or BA for a batch processor.
2. The two letters, nn, are either a 1- or 2-digit octal number.

   For a print processor, the value of nn corresponds to the physical device name of the printer, such as LPP0 for the LP0: line printer.

   For a batch processor, the value of nn is a unique octal number, such as BAP0 for the first batch processor.

Command Qualifiers:

/QUEUE
The queue, name, is stopped as defined by the parameter qualifier. No more jobs are dispatched from the queue to a processor until the queue is restarted. The assignments of processors to queues are not affected.

/PROCESSOR
The system stops the processor, name. The queues continue to accept jobs and dispatch jobs to other assigned processors, but no jobs are passed to the processor, name.

Parameter Qualifiers:

PAUSE
Stop the print or batch processor immediately. Place the job in a state where it can be continued when the processor is restarted. The processor remains busy.

/FILE_END
Stop the print or batch processor at the end of the current file. The processor remains busy.

/JOB_END
Stop the print or batch processor at the end of the current job. The processor remains busy.

/ABORT
Stop the print or batch processor immediately. Return the current job, if there is one, to the queue and place it in the held jobs list. The processor is made idle.

Examples:

1. STOP/QUEUE BATCH

   The queue assignments remain unaffected, but no jobs can be dispatched from the BATCH queue until you specify the START/QUEUE BATCH command statement.
2. STOP/PROCESSOR LPP0/JOB_END

   The print processor, LPP0, is halted when the current job is completed.

11.39.2 Stopping the Queue Manager

Use the STOP command to halt all queueing operations. After you specify this command, most queue-related commands (including SUBMIT and PRINT) are ignored except for the STOP/PROCESSOR command. Queue operations are restarted when you issue the START/QUEUE/MANAGER command.

Format:

STOP/QUEUE/MANAGER

Command Qualifier:	Default:
/ABORT	/ABORT
/JOB_END	

Prompt:

None

Command Parameter:

None

Command Qualifiers:

/JOB_END
Stop the queues and processors at the end of the current job(s). The processor(s) remains busy.

/ABORT
Stop the queues and processors immediately. Return the current job(s), if there are any, to the queue(s) and place them in the held jobs list(s). The processors are made idle.

Example:

1. STOP/QUEUE/MANAGER

   The queue manager and all queue operations are halted until you issue a START/QUEUE/MANAGER command.

11.40 SUBMIT

The SUBMIT command accumulates one or more specified batch command files into a job and places the job in a specified batch queue.

Format:

SUBMIT [/qualifiers] file-spec [, . . .]

Command Qualifiers:	Defaults:
/QUEUE:queue-name	/QUEUE:BATCH
/PRIORITY:n	n:50
/[NO] RESTART	/RESTART
/[NO] ORIGINAL	/NOORIGINAL
/[NO] PRINT	/PRINT
/JOB:jobname	First six characters of list file-spec
/AFTER: (dd-mmm-yy hh:mm)	Current time

Prompt:

FILE? file-spec [, . . .]

Command Parameter:

file-spec
Specifies the file containing batch commands. If no file type is included, .CMD is the default.

Command Qualifiers:

/QUEUE:queue-name
Specifies the batch queue into which the job is to be placed. The default queue-name is BATCH

/PRIORITY:n

Specifies the queue priority of the job.

/[NO] RESTART
Indicates whether or not the job can be restarted from the beginning in the event that it is interrupted for some reason.

/[NO] PRINT
Specifies whether or not to print the log file for the job.

/[NO] ORIGINAL
Specifies whether or not the system should make temporary copies of files to be submitted from a private volume. /ORIGINAL indicates no temporary copies are to be made; i.e., the original copy will be submitted to the batch or print queue. This allows private volumes to be dismounted.

/JOB:job-name
Specifies a name for the batch job.

/AFTER: (dd-mmm-yy hh:mm)
Specifies a date and time after which the job shall be made eligible for submission to a batch processor.

Examples:
This command places a batch job, name BATCH1 and containing the file BATCH1.CMD into the queue named BATCH.

```
>SUBMIT
FILE? BATCH1
>
```

The job TEST shall be placed in the queue BAT and become eligible for processing after 5:30 p.m. on January 30, 1978. The files FIRST.CTL and SECOND.CTL contain the batch commands of which TEST will consist.

```
>SUBMIT/QUEUE:BATCH/AFTER:(30-JAN-78 17:30)-
DCL>/JOB:TEST -
DCL>FIRST.CTL, SECOND.CTL
>
```

11.41 TYPE

The TYPE command prints or displays the contents of one or more specified files on the issuing terminal.

Format:

TYPE file-spec[, . . .]

Prompts:

FILE? file-spec[, . . .]

Command Parameter:

file-spec
Specifies a file to be printed or displayed on the terminal or in the batch log file.

Command Qualifiers:

None

NOTE

Each file-spec must include a file name and a file type. Wildcards are permitted in the file name, file type, and file version components of the file specification.

Example:
The following command displays the contents of the file A.CBL on the terminal from which the TYPE command is issued.

```
>TYPE A.CBL
IDENTIFICATION DIVISION.
PROGRAM-ID.  SAMPLE.
REMARKS.  THIS JUST PRINTS A BRIEF MESSAGE.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. PDP-11-70.
OBJECT-COMPUTER. PDP-11-70.
DATA DIVISION.
WORKING-STORAGE SECTION.
01      END-MESSAGE PIC X(40) VALUE IS "THE TASK IS COMPLETED.".
PROCEDURE DIVISION.
MESSAGE-PRINT.
        DISPLAY END-MESSAGE.
        STOP RUN.
>
```

11.42 UNLOCK

The UNLOCK command releases a locked file for access.

Format:

UNLOCKfile-spec [, . . .]

Prompt:

FILE? file-spec[, . . .]

Command Parameter

file-spec
Specifies a file to be released from a locked condition for access. You must specify both a file name and a file type.

Command Qualifier:

None

NOTES

1. Locked files occur as the result of being stored by the system under abnormal conditions. If they are open during ABORT operation, for instance, they are stored without being normally closed. Locked files may not contain the information you expect, due to the abnormal closing.
2. Locked files are indicated in the directory listing by an L between the block count and the storage date.

Example:
The following sequence demonstrates the use of an UNLOCK command to gain access to a locked file.

```
>COBOL A.CBL

DCL>ABORT COBOL
>
14:56:25    TASK "COBT4 " TERMINATED
            ABORTED VIA DIRECTIVE OR MCR
>DIRECTORY A.*


DIRECTORY DB0:[350,230]
19-JUL-78 14:56

A.TST;1                0.            02-JUN-78 13:56
A.LST;1                1.            02-JUN-78 13:56
A.ODL;1                1.            18-JUL-78 14:01
A.OBJ;1                2.            18-JUL-78 14:01
A.TSK;1               27.       C    18-JUL-78 14:01
A.CBL;3                1.            18-JUL-78 14:03
A.TMP;2                0.          L 19-JUL-78 14:56

TOTAL OF 32./32. BLOCKS IN 7. FILES


>UNLOCK A.TMP
>DIRECTORY A.TMP

DIRECTORY DB0:[350,230]
19-JUL-78 14:59

A.TMP;2                0.            19-JUL-78 14:56

TOTAL OF 0./0. BLOCKS IN 1. FILE


>
```

APPENDIX A
SAMPLE DIALOG

This appendix presents the dialog of starting the system, starting a transaction processor, and initializing and assigning print and batch processors.

A.1 STARTING THE SYSTEM

The following dialog begins after manipulating the console switches as described in Chapter 2 in this guide. At the completion of the startup dialog, the SYSMOD utility is run to define a journal device in preparation for running a transaction processor.

```
        TRAX TRANSACTION PROCESSING SYSTEM VERSION 1.0

>
>
>
>
>
>RUN $INIT/PAR=TP1PAR
>@ <EOF>
>
Target system name? TEST

     Time? 8:01

     Date? 05-Jul-78

05-Jul-78 8:02      FROM VT1:      TO ALL:
TEST TRAX IS ON THE AIR.

     INIT has successfully finished.

>
>RUN $SYSMOD
SYSMOD  BL6-02
TRAX Modify global system parameters
Command <EXIT>? CHANGE
Old device? NONE
New device? MM1:
Command <EXIT>? (RET)
>
```

You must run the SYSMOD utility before starting a transaction processor if you intend to journal the transaction processor activities.

A.2 STARTING A TRANSACTION PROCESSOR

The following dialog can be followed to initialize and start a transaction processor and to include logging and journalling capability.

```
>RUN $TPCTRL
Command <BRIEF>? INSTALL
Transaction processor name ? TPTEST
Partition <TP1PAR>? (RET)
Trace transaction processor <NO>? YES
Write protect data base <NO>? (RET)
High-Performance RMS <YES>? (RET)
Forms definition file version <LATEST>? (RET)
Command <START>? (RET)
Transaction processor name <TPTEST>? (RET)
Enable journalling <YES>? (RET)
Enable logging <NO>? YES
Command <EXIT>? (RET)
>RUN $TPSTAT
Transaction processor Name? TPTEST
Interval between samples? 5
Number of samples? 11
Output file <TPTEST.SMP>? (RET)
>
```

In the example, the transaction processor is started with (1) statistics sampling, (2) journalling, and (3) logging enabled. The statistics utility is then activated with the TPSTAT utility.

A.3 ADDING A QUEUE AND PROCESSOR

The following dialog: (1) adds a batch queue, (2) adds a batch processor to the batch queue, and (3) adds a second auto-spooled queue and processor.

```
>INITIALIZE/QUEUE TRXTST/BATCH
>INITIALIZE/PROCESSOR BAP2
>ASSIGN/QUEUE TRXTST BAP2
      .
      .
      .
>INITIALIZE/QUEUE LPQ1/PRINT
>INITIALIZE/PROCESSOR LPP1
```

In the example, you must assign the batch processor to a batch queue before the system can pass jobs to the batch processor. The print processor, LPP1, however is automatically assigned to the auto-spooled queue, LPQ1, by the system.

APPENDIX B
TRAX SUPPORT ENVIRONMENT MESSAGES

This appendix describes the system messages created by the TRAX Support Environment commands. All commands that can be issued from the TRAX Support Environment with the exception of the BASIC, COBOL, and MACRO programming languages and the TRAX Editor error messages are listed. These error messages are described in their respective user reference manuals.

B.1 ABORT

These are the error messages created by the ABORT command. The ABORT command functions are described in the *TRAX Support Environment User's Guide,* and in Chapter 11 of this manual.

ABO – TASK MARKED FOR ABORT
An attempt has been made to abort a task which is already marked for abort.

ABO – TASK NOT ACTIVE
The specified task is not currently active.

Messages from Task Termination Notification Routine (TKTN):
TKTN displays information about task aborts, whether caused by an explicit ABORT command or some other force. The display has the format:

TASK "<taskname>" TERMINATED
<abort cause>

Following the displayed cause for the abort is a list of the task's registers at the time of the abort. The possible causes of the abort are described below.

Abort Cause Messages:

ABORTED BY DIRECTIVE OR MCR
Either TRAX or an Executive directive issued by another task caused the task to be aborted.

ABORTED VIA MCR
TRAX aborted the task and requested a post-mortem dump.

CHECKPOINT FAILURE. READ ERROR.
The task could not be read back into memory from disk after being checkpointed.

LOAD FAILURE. READ ERROR
The task could not be loaded from disk because of a hardware error.

PARITY ERROR
A parity error occurred while the task was executing. The task was fixed in memory so that the memory could not be reused by another task.

TASK EXIT WITH OUTSTANDING IO
The task exited with one or more outstanding I/O requests. Tasks should terminate all I/O operations before exiting. The system does, however, clean up all outstanding I/O.

B.2 ALLOCATE

These are the error messages created by the ALLOCATE command. The ALLOCATE command functions are described in the *TRAX Support Environment User's Guide* and in Chapter 11 of this manual.

ALL – DEVICE ATTACHED
The specified device cannot be allocated because it is attached to a running task.

ALL – PSEUDO DEVICE ERROR
The specified device is a pseudo device. Pseudo devices cannot be allocated.

ALL – PUBLIC DEVICE
The command attempted to allocate a public device. Public devices cannot be allocated.

ALL – USER LOGGED ON TERMINAL
The command attempted to allocate a terminal that has been logged-in by another user. Logged-in terminals cannot be allocated.

B.3 APPEND

These are the error messages created by the APPEND command. The APPEND command functionalities are described in **TRAX** *Support Environment User's Guide* and in Chapter 11 of this manual.

NOTE

A fatal error in the cnv utility is marked by a preceeding question mark "?". If the message has a question mark in brackets [?] the error may be either fatal or diagnostic. If the error message has no preceeding question mark the error is diagnostic. The error messages prefixed with "DSC" refer to volume archiving. The others refer to file archiving.

?cnv–DEVICE OFF LINE – device

Description:
The indicated device exists on the system but the attempt to access it has been prohibited for one of the following reasons.

1. The device is not ready.
2. No volume is mounted on the device.
3. The device is currently reserved by another job.
4. The device requires privileges for ownership and the user does not have privilege.
5. The device has been disabled.

Suggested User Action:
Determine the nature of the problem and take corrective action.

cnv–DEVICE/FILE IS FULL – device/filename

Description:
The utility cannot create an output file on the indicated device because of insufficient space or the indicated file cannot be extended due to insufficient space.

Suggested User Action:
Reenter the command using another device for output files or copy the indicated file to another device and retry the command. Optionally, delete unneeded files on the indicated device and reenter the original command line.

?cnv–FILE NOT AVAILABLE – filename

Description:
The indicated file is being accessed for exclusive use by another job.

Suggested User Action:
Periodically retry the command until the file has been released.

?cnv–ILLEGAL DEVICE – device

Description:
The indicated device does not exist.

Suggested User Action:
Reenter the command line with a corrected device specification.

?cnv–NO SUCH KEY FOR FILE – value

Description:
The specified key of reference value represents a non-existent key in an indexed file.

Suggested User Action:
Reenter the command with a correct key of reference value.

?cnv–NOT A DIRECTORY DEVICE – device

Description:
The user has issued a directory-oriented command for a device (such as a printer) that does not have directories (accounts).

Suggested User Action:
Reenter the command line without specifying an account.

APP – CANNOT FIND DIRECTORY FILE

Description:
UFD specified does not exist on this volume.

Suggested User Response:
Reenter the command line, specifying the correct UFD or the correct volume.

APP – CANNOT FIND FILE(S)

Description:
The file(s) specified in the command were not found in the designated directory.

Suggested User Response:
Check the file specifier and reentry the command line.

APP – I/O ERROR ON INPUT FILE or
APP – I/O ERROR ON OUTPUT FILE

Description:
One of the following conditions may exist:

- The device is not on-line.
- The device is not mounted.
- The hardware has failed.
- The volume is full (output only).
- Input file is corrupted.

Suggested User Response:
Determine which condition caused the message and correct that condition. Reenter the command line.

APP – NOT A DIRECTORY DEVICE

Description:
A directory-oriented command was issued to a device that does not have directories (such as a printer).

Suggested User Response:
Reenter the command line without specifying a UFD.

APP – OPEN FAILURE ON INPUT FILE or
APP – OPEN FAILURE ON OUTPUT FILE

Description:
The specified file could not be opened. One of the following conditions may exist:

- The file is protected against access.
- A problem on the physical device (e.g., device cycled down).
- The volume is not mounted.
- The specified file directory does not exist.
- The named file does not exist in the specified directory.

Suggested User Response:
Determine which condition caused the message and correct that condition. Reenter the command line.

B.4 ARCHIVE

These are the error messages that are created by the ARCHIVE command. The ARCHIVE command functions are described in the *TRAX Support Environment User's Guide* and in Chapter 11 of this manual.

NOTE

A question mark "?" preceding the bck or rst utility error messages indicates a fatal error. A question mark in brackets [?] indicates that the error may be fatal or diagnostic. If no question mark precedes the error message the error is diagnostic. The error messages prefixed with "DSC" refer to volume archiving. The others refer to file archiving.

DSC – 7 DUP. DEV. NAME
The same device was entered more than once in the command.

Reenter the command string with the devices specified only once.

DSC – 9 DEV. device: NOT IN SYSTEM
The specified device is not present in the configuration of the operating system being used.

Check the device identifier that was entered in the command string, and retry the command.

DSC – 10 DEV. device: NOT files-11
The specified input device is not formatted as a files-11 device.

Check the input device to ensure it is the one desired, and reenter the command.

DSC – 14 OUTPUT TAPE ON device: IS NOT AT BOT
The specified continuation tape is not at load point.

Remount or reset the tape at load point and reenter the command.

DSC – 18 TAPE device: NOT ANSI FORMAT
The tape is not in correct format for a DSC operation.

Check the tape and change if necessary.

DSC – 21 TAPE device: A CONTINUATION TAPE
The tape has been mounted out of sequence.

Reenter the command, specify input tapes in proper order.

DSC – 23 FAILED TO FIND HOME BLOCK device:
A read error occurred when trying to copy from the input disk. Either the disk is bad, the home block is bad, or the disk is not in files-11 format.

Check the disk in question, change disk drives if possible, and reenter the command.

DSC – 26 I/O ERROR B ON device:
The I/O error indicated by the message that follows explains why the file header on the input device could not be read. The specified file is lost.

Retry the operation after correcting the cause of the error on the input device.

DSC – 27 I/O ERROR B ON device:
The I/O error indicated by the message that follows explains why the file header on the output device could not be read. The specified file is lost.

Retry the operation after correcting the cause of the error on the output device.

DSC – 28 CODE A
The file header for the storage bit map file cannot be read.

The disk is unusable and therefore cannot be copied.

DSC – 29 I/O ERROR C ON device:
The following message explains the error that occurred while reading the specified file.

Retry the operation.

DSC – 30 I/O ERROR D ON device:
A read error, as indicated by the diagnostic message which follows, occurred when reading the name or boot block of the disk.

Retry the operation on a new drive.

DSC – 31 RELATIVE VOLUME X OF SET NOT MOUNTED
The specified tape is not on the system.

Mount the tape and reenter the command.

DSC – 36 I/O ERROR E ON device: file id
The message that follows explains the I/O error that occurred while reading the specified file header.

Retry the operation.

DSC – 37 INPUT DEVICE device: file id file number NOT PRESENT
The specified file does not have a file header in the index file; the file is not copied.

This is a warning only. If desired, the operation may be retried on a different disk drive.

DSC – 38 INPUT DEVICE device: file id file number IS DELETED
The specified file was found to be partially deleted on the input disk and was not copied.

This is a warning only. No action is required.

DSC – 39 INPUT DEVICE device: file id UNSUPPORTED STRUCTURE LEVEL
The specified input disk is not a level one (ODS1) disk and cannot be used.

Retry the operation with a level one disk.

DSC – 40 INPUT DEVICE device: file id file number FILE NUMBER CHECK
An incorrect file header was read from disk causing the specified file to be lost.

Retry the operation.

DSC – 41 INPUT DEVICE device: file id file number FILE HEADER CHECKSUM ERROR
Incorrect file header contents cause the specified file to be lost.

Retry the operation.

DSC – 42 INPUT DEVICE device: file id SEQUENCE NUMBER CHECK
The sequence number is incorrect.

Retry the operation and/or replace the disk.

DSC – 43 INPUT DEVICE device file id file number SEGMENT NUMBER CHECK
The linkage connecting file segments has been broken; the specified file is lost.

Retry the operation.

DSC – 44 DIRECTIVE ERROR
An internal error has occurred, usually the result of a system overload.

Retry the operation.

DSC – 45 I/O ERROR F ON device:
The message that follows indicates that the specified input device may cause a subsequent error.

This message is a warning only. No action is required unless another error message is displayed. If another error message is displayed, correct the cause of the error and reenter the command.

DSC – 46 I/O ERROR F ON device:
The message that follows indicates that the specified output device may cause a subsequent error.

This message is a warning only. No action is required unless another error message is displayed. If another error message is displayed, correct the cause of the error and reenter the command.

DSC – 47 I/O ERROR I ON device: file id file number virtual block number
An I/O error occurred which is explained by the message that follows which resulted in bad data being read from the specified virtual block number.

This is a warning message only. The block specified should be examined to determine the extent of the error.

DSC – 48 I/O ERROR I ON device: file id file number virtual block number
An I/O error occurred which is explained by the message that follows which resulted in bad data being read from the specified virtual block number.

This is a warning message only. The block specified should be examined to determine the extent of the error.

DSC – 49 VERIFICATION ERROR ON device: file id virtual block number
This is a warning signifying that the input and output devices did not match.

DSC – 50 BAD DATA BLOCK ON device: file id file number virtual block number
A parity error occurred when copying the blocks contents from disk. The block specified on the output disk contains erroneous data.

When the copy operation is completed, the data contained in the specified block should be examined and corrected.

DSC – 55 INPUT FILE ON device: WILL BE RESYNCHRONIZED
The tape position was lost while reading the input tape. The file specified in the message, as well as some subsequent files, may be lost. Additional error messages will probably occur.

Retry the operation from the beginning.

DSC – 57 OUTPUT FILE HEADER FULL ON device:
Too many blocks on the output disk have caused inconsistencies in file header data. The specified file is lost.

Retry the operation with a different output disk.

DSC – 58 OUTPUT FILE HEADER ON device: NOT MAPPED – file id file number
Space for the specified file header was not allocated. The file is lost.

Retry the operation; a new disk may be required.

DSC – 59 I/O ERROR G ON device:
The message that follows explains the I/O error that occurred while writing the specified file.

Retry the operation.

DSC – 60 FAILED TO READ FILE EXTENSION HEADER ON device: file id file number
When copying from the input disk, an extension header was searched for, but but not found. The remainder of the specified file was lost. A problem may exist with the input disk, or a preceding I/O error occurrence may have caused an inconsistency.

Retry the operation.

DSC – 61 FAILED TO ALLOCATE HOME BLOCK device:
The home block cannot be created on the specified disk device because it has too many bad blocks.

Replace the device and reenter the command.

DSC – 62 INDEX FILE ALLOCATION FAILURE device:
Too many bad blocks exist to allow the allocation for specified file.

Replace the disk and reenter the command.

DSC – 63 OUTPUT DISK device: IS NOT BOOTABLE
Logical block number 0 of the specified disk or tape is bad.

This is a warning only. No action is required.

DSC – 64 INVALID BAD BLOCK DATA device:
The bad block data on the output disk are invalid.

Run the BAD utility on the disk; manually enter bad block data; or reenter the command, specifying another disk.

DSC – 65 BAD BLOCK FILE FULL device:
Too many bad blocks exist on the output disk.

Replace the disk and retry the command.

DSC – 66 NO BAD BLOCK DATA FOUND device:
No bad block data exists for the specified output disk.

If bad block data is not desired, ignore the message. Otherwise, run the BAD program on the disk; manually enter bad block data; or reenter the command using a new disk.

DSC – 67 OUTPUT DEVICE device: IS A DIAGNOSTIC PACK – DO NOT USE IT
The specified output disk is a diagnostic device, and cannot be used.

Mount new output disk and reenter the command.

DSC – 68 CODE B ON device: file id file number VBN: expected x found y
The tape position was lost when reading the virtual block number specified. Some data may be lost.

Determine the extent of the error. If necessary, try the tape on another drive, or create another tape.

DSC – 69 CODE C ON device: file id file number VBN
The position of the tape was lost while reading the data file specified. Data beyond the VBN mentioned are lost.

Recreate the tape, or retry the operation on a different tape drive.

DSC – 70 CODE D ON device: file id file number expected x found y
The tape position was lost while reading the tape mentioned in the message. All of "y" and some of "x" are lost.

Retry the entire operation.

DSC – 71 FAILED TO MAP OUTPUT FILE ON device: file id file number
An inconsistency occurred when writing the specified file to the output disk. The file header did not specify the correct number of virtual blocks required to write the file and the file is lost.

Retry the operation.

DSC – 72 OUTPUT DISK device: IS TOO SMALL – BLOCKS NEEDED
The output disk is not large enough to accommodate the data to be transferred.

Retry the operation specifying a larger output disk.

DSC – 73 I/O ERROR C ON device:
The following message explains the error that occurred while reading the specified file.

Retry the operation.

DSC – 74 I/O ERROR H ON device:
The message that follows explains the I/O error that occurred while writing the specified file.

Retry the operation.

DSC – 75 I/O ERROR J ON device:
An I/O error (which follows) occurred when reading the tape labels on the specified device.

Retry the operation on a different tape drive.

DSC – 76 INPUT TAPE ON device: MUST BE AT BOT
The specified tape must be at beginning of tape or its load point. This message is also displayed during a /VE operation merely to indicate that the current volume is rewinding to enable the verify pass.

If /VE was not specified, check the tape and remount at load point.

DSC – 77 WRONG INPUT TAPE ON device: EXPECTING file id FOUND file id
The input tapes were specified out of sequence.

Check the tapes, reenter in proper order after receiving mount instructions.

DSC – 78 CODE E ON device: AFTER file id file number
This is the result of a read error from tape. When trying to read an attribute block, some other block was accessed. The file following the file specified in the error message is lost.

Retry the operation.

DSC – 79 I/O ERROR K ON device:
The message that follows explains the I/O error that occurred while reading the specified file.

Retry the operation.

DSC – 80 I/O ERROR L ON device:
The message that follows explains the I/O error that occurred while reading the file header.

Retry the operation.

DSC 81 – INPUT TAPE device: RESYNCHRONIZED AT file id file number
The tape position has been recovered. Some data preceding the file specified were lost.

This is usually received in conjunction with one or more error messages, all indicating that the input tape was either read incorrectly or recorded badly. The tape should be recreated and the operation reinitiated.

DSC – 82 TAPE FILE filelabel NOT FOUND ON device:
The input tape specified does not contain the file identified as "filelabel".

Check the filelabel and the tape, reenter when the correct tape and filelabel are specified.

DSC – 83 EXPECTED EXTENSION HEADER NOT PRESENT ON device: file id file number
A tape read error occurred causing the specified file to be lost.

If the error message was preceded by one or more I/O warning messages, the operation should be retried. If not, the input tape is bad and should be regenerated.

DSC – 84 CODE F ON device: AFTER file id file number
This is the result of a read error from tape. When trying to read a file header some other block type was accessed. The file following the file specified in the error message is lost.

Retry the operation.

DSC – 85 I/O ERROR M ON device:
The following message explains why the specified file could not be read.

Retry the operation.

DSC – 86 INDEX FILE DATA NOT PRESENT device:
When reading the input tape specified, a file other than the index file was accessed due to a tape error or an I/O error.

Recreate the tape, or retry the same tape on a different tape drive.

DSC – 87 I/O ERROR N ON device:
The message that follows explains the I/O error that occurred while restoring the index and storage map files from the specified input tape.

Retry the operation using a different input tape drive.

DSC – 88 VOLUME SUMMARY DATA NOT PRESENT device:
Either the input tape is not a DSC tape, or incomplete data are contained.

Check the tape and reenter the command.

DSC – 89 I/O ERROR O device: file id file number
The message that follows explains the I/O error that occurred while writing the specified file header.

Retry the operation.

NOTE

The DSC errors identified as I/O errors are accompanied by one or more of the following error messages to explain the type of I/O error that occurred.

BAD BLOCK NUMBER
The block does not exist on the disk; an internal DSC error has occurred; or the block is bad.

Retry the operation with a new disk and/or disk drive.

BAD BLOCK ON DEVICE
A device malfunction has occurred, or a tape was used with bad data on it resulting in a block containing incorrect information.

Retry the operation.

BLOCK CHECK
A parity error occurred indicating that bad data may have been transferred.

Retry the operation.

DATA OVERRUN
The physical tape used is larger than was expected; the tape got out of position, or is in the wrong fromat.

Make sure the tape is the right one and retry the operation.

DEVICE NOT READY
The device is not ready or not up to speed, or a blank tape has been used as an input tape.

Retry the operation after checking that the device is online and correctly mounted.

DEVICE OFFLINE
The device is not in the system.

Check the device, the device specification in the command string, and reenter the command.

DEVICE WRITE LOCKED
The disk drive is write locked.

Write enable the disk drive and reenter the command.

END OF FILE DETECTED
The tape position was lost.

Retry the operation.

END OF TAPE DETECTED
The tape position was lost.

Retry the operation.

END OF VOLUME DETECTED
The tape position was lost.

Retry the operation.

FATAL HARDWARE ERROR
A hardware malfunctions has occurred.

Retry; if error repeats call DIGITAL Field Service.

INSUFFICIENT POOL SPACE
The operating system is overloaded.

Retry the operation.

PARITY ERROR ON DEVICE
A device malfunctions or tape incompatibility has occurred.

Retry the operation.

PRIVILEGE VIOLATION
A device has been mounted as FILES-11.

TRAX users: DISMOUNT the disk and retry the operation.

UNKNOWN SYSTEM ERROR
An undefinable I/O error has occurred.

Retry the operation.

The following error messages appear only in the stand-alone version of DSC used as part of the SYSGEN procedure.

Table B-1 General Error and I/O Error Message Codes

General Error Message Codes	
Symbol	**Meaning**
Code A	Failed to read storage map header
Code B	Input data out of phase
Code C	Non-data block encountered
Code D	Input file out of phase
Code E	File attributes out of phase
Code F	File header out of phase
I/O Error Message Codes	
Symbol	**Meaning**
A	Reading index file bit map
B	Reading index file header
C	Reading storage bit map
D	Reading boot or home block
E	Reading file header
F	Input (or output device)
G	Writing index file bit map
H	Writing storage bit map header
I	Reading input device
J	In input tape labels
K	Reading file attributes
L	Reading file header
M	Reading index file data
N	Reading summary data
O	Writing file header

utl – DEVICE/FILE IS FULL – device/filename

Description:
The utility cannot create an output file on the indicated device because of insufficient space or the indicated file cannot be extended due to insufficient space.

Suggested User Action
Reenter the command using another device for output files or copy the indicated file to another device and retry the command. Optionally, delete unneeded files on the indicated device and reenter the original command line.

?utl – ERROR WITH WILDCARDS

Description:
The wild card processor has returned an error to the utility during resolution of wild cards in a file specification.

Suggested User Action
Reenter the command line. If the same condition recurs, use successive invocations of the utility and non-wild carded file specifications to achieve the original desired result.

?utl – FILE NOT AVAILABLE – filename

Description:
The indicated file is being accessed for exclusive use by another job.

Suggested User Action
Periodically retry the command until the file has been released.

?utl – FILE POSITION LOST – filename

Description:
The utility has lost its position within a container file on magnetic tape while rewinding or backspacing. The error may have been caused by hardware failure.

Suggested User Action
Determine from the output account or summary listing file the extent of the processing that was completed prior to the occurrence of the error. Reenter the command line eliminating file specifications of files successfully processed. Use a new tape volume and/or a different tape drive, the file is input to RESTORE, the utility cannot restore the data records within the indicated blocks of the original file.

?utl – I/O ERROR ENCOUNTERED ON OUTPUT FILE – filename

Description:
One of the following conditions exists:

1. The device is not on line.
2. The device is not mounted.
3. The hardware has failed.
4. The volume is full.

Suggested User Action
Rectify the condition and reenter the command line.

utl – INPUT FILE IS NOT BACKUP FILE – filename

Description:
The utility requires that the input file be a backup file. The user has specified a file that is not in backup format. For example, a file not in backup format is specified as input to ARCHIVE/RESTORE.

Suggested User Action
Reenter the command line with the correct file specification.

utl – NO SUCH FILE

Description:
No files in the UFD correspond to the wild cards of a file specification.

Suggested User Action
Obtain a listing of the files in the desired UFD. Reenter the command with the desired file specification.

?utl – PRIVILEGE VIOLATION – filename

Description:
The user does not have the privileges necessary to access the indicated file.

Suggested User Action
Have the owner of the file change its privilege specification.

utl – READ ERROR, INTEGRITY CHECK TABLE AND REWRITE DATA MAY HAVE BEEN LOST

Description:
The utility has encountered an error while attempting to read internal data maintained in a backup file or container file.

Suggested User Action
Retry the command. If the same error occurs, check summary listing file created at the time the backup or container file was created. Determine from this file which file or files cannot be completely restored.

utl – READ ERROR ON FILE ATTRIBUTES – filename

Description:
The volume is corrupted or the user does not have the necessary privileges to access the indicated file.

utl – READ ERROR ON FILE PROLOGUE – filename

Description:
The utility is unable to read the prologue (internal RMS-11 information within a file) of the specified file. The file is bypassed and processing continues.

Suggested User Action
Reenter the command specifying the subject file only. If the same error occurs, the indicated file cannot be properly accessed on the device. Use the RESTORE utility to retrieve a new copy of the file.

utl – READ ERROR OR INCONSISTENT DATA. MAY HAVE LOST FILES.

Description:
The utility has encountered an error while reading a backup or container file.

Suggested User Action
Retry the command. If the same error occurs, check summary listing file created at the time the backup or container file was created. Determine from this file which file or files may have been lost.

?utl – REWIND OR SPACE ERROR ON FILE – filename

Description:
The utility has encountered an error while rewinding or backspacing on magnetic tape.

Suggested User Action
Retry the command. If the condition occurs again, mount the volume on another drive and retry the command.

?utl – SELECT ERROR – dev

Description:
One of the following conditions exists:

1. The device is not on-line.
2. The device is not mounted.
3. The hardware has failed.

Suggested User Action
Rectify the condition and reenter the command line.

utl – UNABLE TO RESTORE SPECIAL ATTRIBUTES – filename

Description:
The utility was unable to restore the file with one or more of its original date attributes (e.g., creation date, revision date) or its original protection specification.

Suggested User Action:
Use the DISPLAY utility to determine which attributes of the file were not restored.

?utl – WRITE ERROR ON ATTRIBUTES OF FILE – filename

Description:
The volume is corrupted or the user does not have the necessary privileges to write the file.

Suggested User Action:
Verify access to file.

?utl – WRITE ERROR ON CREATE OF OUTPUT FILE – filename

Description:
One of the following conditions exists:

1. The device is not on line.
2. The device is not mounted.
3. The hardware has failed.
4. The volume is full.

Suggested User Action:
Rectify the condition and reenter the command line.

?utl – WRITE ERROR ON INTEGRITY CHECK TABLE ON OUTPUT FILE – filename

Description:
The utility is unable to write internal data integrity checking tables in the output backup file.

Suggested User Action:
If the output medium is magnetic tape, use a different tape volume and retry the command. If the output medium is disk, rename the output file so that the utility will not attempt to use the space and retry the command.

B.5 COPY

These are the error messages created by the COPY command. The functions of the COPY command are described in *TRAX Support Environment User's Guide* and in Chapter 11 of this manual.

COP – ALLOCATION FAILURE – NO CONTIGUOUS SPACE

Description:
Contiguous space available on the output volume is insufficient for the file being copied.

Suggested User Response:
Delete all files that are no longer required on the output volume, and reenter the command line.

COP – ALLOCATION FAILURE ON OUTPUT FILE or COP – ALLOCATION FAILURE – NO SPACE AVAILABLE

Description:
Space available on the output volume is insufficient for the file being copied.

Suggested User Response:
Delete all files that are no longer required on the output volume, and reenter the command line. Also, use the archive command.

COP – BAD USE OF WILD CARDS IN DESTINATION FILE NAME

Description:
A wildcard * was specified for an output filename where use of a wildcard is explicitly disallowed.

Suggested User Response:
Reenter the command line with the proper output file explicitly specified.

COP – CANNOT FIND DIRECTORY FILE

Description:
UFD specified does not exist on this volume.

Suggested User Response:
Reenter the command line, specifying the correct UFD or the correct volume.

COP – CANNOT FIND FILE(S)

Description:
The file(s) specified in the command were not found in the designated directory.

Suggested User Response:
Check the file specifier and reenter the command line.

COP – CLOSE FAILURE ON INPUT FILE or
COP – CLOSE FAILURE ON OUTPUT FILE

Description:
The input or output file could not be properly closed. The file is locked to indicate possible corruption.

Suggested User Response:
Reenter the command line. If the error recurs, run a validity check of the file structure using the verify utility (VFY) on the volume in question to determine if it is corrupted. The functions of the Verify utility are described in Chapter 11 of this manual.

COP – DIRECTORY WRITE PROTECTED

Description:
COP could not remove an entry from a directory because the device was write-protected, or because of privilege violation.

Suggested User Response:
Enable the unit for write operations or have the owner of the directory change its protection.

COP – FAILED TO ENTER NEW FILE NAME

Description:
You have specified a file that already exists in the directory file, or do not have the necessary privileges to make entries in the specified directory file.

Suggested User Response:
Reenter the command line, ensuring that the filename and UFD are specified correctly, or request COP under the correct UIC and reenter the command line.

COP – FAILED TO WRITE ATTRIBUTES

Description:
Volume is corrupted or you do not have the necessary privileges to write the file attributes.

Suggested User Response:
Ensure that COP is running under the correct UIC. If the UIC is correct, then run the validity check of the file structure verification utility (VFY) against the volume in question to determine where and to what extent the volume is corrupted.

COP – ILLEGAL "*" COPY TO SAME DEVICE AND DIRECTORY

Description:
You attempted to copy all versions of a file into the same directory that is being scanned for input files. This results in an infinite number of copies of the same file and is not allowed.

Suggested User Response:
Reenter the command line, renaming the files or copying them into a different directory.

COP – I/O ERROR ON INPUT FILE or
COP – I/O ERROR ON OUTPUT FILE

Description:
One of the following conditions may exist:

- The device is not on-line.
- The device is not mounted.
- The hardware has failed.
- The volume is full (output only).
- Input file is corrupted.

Suggested User Response:
Determine which condition caused the message and correct that condition. Reenter the command line.

COP – NO SUCH FILE(S)

Description:
The file(s) specified in the command were not found in the designated directory.

Suggested User Response:
Check the file specifier and reenter the command line.

COP – OPEN FAILURE ON INPUT FILE or
COP – OPEN FAILURE ON OUTPUT FILE

Description:
The specified file could not be opened. One of the following conditions may exist:

- The file is protected against access.
- A problem on the physical device (e.g., device cycled down).
- The volume is not mounted.
- The specified file directory does not exist.
- The named file does not exist in the specified directory.

Suggested User Response:
Determine which condition caused the message and correct that condition. Reenter the command line.

B.6 CREATE

These are the error messages created by the CREATE command. The functions of the CREATE command are described in the *TRAX Support Environment User's Guide* and Chapter 11 of this manual.

NOTE

A question mark [?] preceding the dfn utility message indicates a fatal error. A question mark in brackets [?] indicates that the error may be fatal or diagnostic. If no question mark preceeds the error message it is a diagnostic.

?dfn – DEVICE OFF LINE – device

Description:
The indicated device exists on the system but the attempt to access it has been prohibited for one of the following reasons.

1. The device is not ready.
2. No volume is mounted on the device.
3. The device is currently reserved by another job.
4. The device requires privileges for ownership and the user does not have privilege.
5. The device has been disabled.

Suggested User Action:
Determine the nature of the problem and take corrective action.

?dnf – DEVICE WRITE PROTECTED – device

Description:
The utility cannot access the indicated device for write operations.

Suggested User Action:
Check the hardware condition of the indicated device. Write enable the unit.

?dfn – DEVICE/FILE IS FULL – device/filename

Description:
The utility cannot create an output file on the indicated device because of insufficient space or the indicated file cannot be extended due to insufficient space.

Suggested User Action:
Reenter the command using another device for output files or copy the indicated file to another device and retry the command. Optionally, delete unneeded files on the indicated device and reenter the original command line.

?dfn – DIRECTORY NOT FOUND – filename

Description:
The directory does not exist on the specified device.

Suggested User Action:
Reenter the command with the correct directory specification.

?dfn – FILE ALREADY EXISTS – filename

Description:
The utility has attempted to create an output file that already exists in the output account.

Suggested User Action:
Reenter the command line using a new or corrected filename or delete the existing file and reenter the original command line.

?dfn – ILLEGAL DEVICE – device

Description:
The indicated device does not exist.

Suggested User Action:
Reenter the command line with a corrected device specification.

?dfn – ILLOGICAL DEVICE – device

Description:
The indicated device is not permitted in the context of the command line. For example, the user cannot CREATE an indexed file on magnetic tape.

Suggested User Action:
Reenter the command line with an appropriate device specification.

?dfn – PRIVILEGE VIOLATION – filename

Description:
The user does not have the privileges necessary to create the indicated file.

Suggested User Action:
Have the owner of the file change its privilege specification.

?dfn – WRITE ERROR ON CREATE OF OUTPUT FILE – filename

Description:
One of the following conditions exists:

1. The device is not on line.
2. The device is not mounted.
3. The hardware has failed.
4. The volume is full.

Suggested User Action:
Rectify the condition and reenter the command line.

The following messages pertain to the "CREATE/DIRECTORY" Command.

UFD – DIRECTORY ALREADY EXISTS
The requested UFD already existed on the volume.

UFD – FAILED TO CREATE DIRECTORY
No space existed on the volume, or an I/O error occurred.

UFD – NOT FILES-11 DEVICE
The device on which the UFD was to be created was not a Files-11 device, and therefore could not support UFDs.

UFD – WRITE ATTRIBUTES FAILURE
An error was encountered while writing the attributes of either the MFD or the newly created UFD.

UFD – WRONG VOLUME
The volume label and the label specified in the command did not match.

UFD – VOLUME NOT MOUNTED
The volume on which a UFD is to be created must be mounted before accessing the files-11 structure.

B.7 DCL

The error messages described in this section are command independent. The system prefixes the error message with a unique 3 letter code derived from the command name. For example:

MES – YOU DO NOT HAVE THE PRIVILEGE TO ISSUE THIS COMMAND

The 3 letter code "MES" is derived from the MESSAGE command. In this section the command names are substituted with "XXX" to signify that the operating task mnemonic is inserted in this position.

XXX – ILLEGAL FUNCTION

Description:
The command line contains an illegal command name.

Use the help command to get a list of valid commands.

XXX – SYNTAX ERROR

Description:
The command line has a syntax error.

Suggested User Response:
Consult the *TRAX Support Environment User's Guide* for the correct syntax. Reenter the command line.

XXX – FUNCTION NOT UNIQUE

Description:
You did not enter enough characters to uniquely identify the command.

Suggested User Response:
Consult the *TRAX Support Environment User's Guide* for the correct spelling of the command. Reenter the command line.

XXX – ILLEGAL QUALIFIER

Description:
The command line contains an illegal qualifier for the command.

Suggested User Response:
Consult the *TRAX Support Environment User's Guide* for the correct task qualifiers. Reenter the command line.

XXX – QUALIFIER NOT UNIQUE

Description:
You did not enter enough characters to uniquely identify the qualifier in the current context.

Suggested User Response:
Consult the *TRAX Support Environment User's Guide* for the correct format. Reenter the command line.

XXX – REQUIRED PARAMETER NOT SPECIFIED

Description:
A required parameter has been omitted from the command line.

Suggested User Response:
Consult the *TRAX Support Environment User's Guide* for the correct format. Reenter the command line.

XXX – INVALID PROTECTION CODE SPECIFIED

Description:
A protection code other than R, W, E or D was specified or the protection codes where not specified in the order RWED.

Suggested User Response:
Check the command line. Reenter the command line correctly.

XXX – FILE SPECIFICATION EITHER INVALID OR NOT SPECIFIED

Description:
The command line contains an invalid file specification or has been omitted.

Suggested User Response:
Check the command line. Reenter the command line corrently.

XXX – PRIMARY KEY NOT SPECIFIED

Description:
The number parameter of the key qualifier is missing. It should be 1.

Suggested User Response:
Check the command line. Reenter the command line correctly.

XXX – CONTRADICTORY QUALIFIER IN KEY SPECIFICATION

Description:
A contradictory pair of qualifiers was specified in the key specification. For example:

UPDATE/NOUPDATE –OR– DUPLICATE/NODUPLICATE

Suggested User Response:
Consult the *TRAX Support Environment User's Guide* for the correct format. Reenter the command line.

XXX – INVALID KEY QUALIFIER VALUE

Description:
A negative number or 0 was specified as the key qualifier value.

Suggested User Response:
Consult the *TRAX Support Environment User's Guide.* Reenter the command line.

XXX – REQUIRED VALUE NOT SPECIFIED FOR POSITION SIZE OR NUMBER

Description:
The numeric value for either the number, position or size qualifiers have not been specified.

Suggested User Response:
Reenter the command line.

XXX – TASK ACTIVE

Description:
You attempted to executive a command twice simultaneously on the same terminal.

Suggested User Response:
Wait till the first invocation is completed. Reenter the command line.

XXX – WILD CARDS NOT PERMITTED

Description:
Filename, type, or the version number of the file must be expressed explicitly. The wildcard default "*" cannot be used.

Suggested User Response:
Reenter the command line.

XXX – ZERO VALUE NOT VALID FOR KEY SIZE OR NUMBER

Description:
The key, size or number qualifiers must be a positive number.

Suggested User Response:
Consult the *TRAX Support Environment User's Guide.* Reenter the command line.

XXX – CONTRADICTORY QUALIFIER

Description:
The command line contains a pair of qualifiers that are contradictory. For example:

```
UPDATE/NOUPDATE
```

XXX – INVALID FILE SPECIFICATION QUALIFIER

Description:
The command line contains an invalid file specification qualifier.

Suggested User Response:
Check the command line. Consult the *TRAX Support Environment User's Guide.* Reenter the command line.

XXX – COMMAND LINE INCOMPLETE

Description:
A necessary parameter was omitted from the command line.

Suggested User Response:
Consult the *TRAX Support Environment User's Guide.* Reenter the command line.

XXX – LIBRARY INVALID ON LAST INPUT FILE

Description:
The library must be specified before all other input files.

Suggested User Response:
Consult the *TRAX Support Environment User's Guide.*

XXX – INVALID COMMAND FUNCTION

Description:
The command option is invalid. For example:

```
SET TABLE
```

Suggested User Response:
Consult the *TRAX Support Environment User's Guide.* Reenter the command line.

XXX – QUEUE MARKED FOR DELETE

Description:
The command line tried to access a queue that is marked for deletion.

Suggested User Response:
Try another queue.

XXX – PROCESSOR MARKED FOR REMOVAL

Description:
The command line tried to access a processor marked for removal.

Suggested User Response:
Try another processor.

XXX – QUEUE DIRECTORY FULL

Description:
The maximum number of queues of a given kind have already been created.

Suggested User Response:
Check queue directory. Delete non-essential queues as required.

XXX – PROCESSOR DIRECTORY FULL

Description:
The maximum number of processors of a given kind have already been created.

Suggested User Response:
Check processor directory. Delete non-essential processors as required.

XXX – REDUNDANT OPERATION

Description:
Self-explanatory. For example, stopping a queue which is already stopped.

Suggested User Response:
Check command line. Reenter the command line.

XXX – DEVICE DOES NOT EXIST

Description:
The device specified in the command line does not exist. Perhaps the unit number has been omitted.

Suggested User Response:
Check the device address with a show device command. Reenter the command line.

XXX – COMMAND PROCESSING TASK NOT IN SYSTEM

Description:
The queue manager has not been initiated and therefore no queue management commands can be processed.

Suggested User Response:
Start the queue manager.

XXX – CONFLICTING QUALIFIER

Description:
A qualifier has been specified when some other attribute of the command makes it meaningless.

Suggested User Response:

XXX – RESERVED QUEUE NAME

Description:
You have attempted to create a queue whose name is reserved by the system.

Suggested User Response:
Choose a different queue name.

XXX – ENTRY is not a job entry

Description:
You have tried to access a queue entry which is not associated with a batch or print job.

Suggested User Response:
Invoke the show queue command to ascertain the correct job entry number.

B.8 DISMOUNT

These are the error messages created by the DISMOUNT command. The functions of the DISMOUNT command are described in the *TRAX Support Environment User's Guide* and Chapter 11 of this manual.

DMO – ALREADY MARKED FOR DISMOUNT
The device-unit had been requested to be dismounted and was in the process of waiting for all accesses to the volume to complete.

DMO – CHECKPOINT FILE STILL ACTIVE
The command attempted to dismount a volume that contained an active checkpoint file. The volume cannot be dismounted until the checkpoint file has been discontinued. In order to dismount your system disk please run the shutdown utility.

DMO – HOME BLOCK CHECKSUM ERROR
The checksum in the home block and the calculated checksum did not agree. This message is usually caused by an I/O error.

DMO – HOME BLOCK I/O ERROR
An error was detected in updating the volume file sequence number in the volume home block.

DMO – NO VOLUME LIST
The command specified a magnetic tape drive for which a mounted volume list does not exist.

DMO – NOT MOUNTABLE DEVICE
The specified device was not a mountable device and therefore could not be dismounted.

DMO – NOT MOUNTED
The specified device was not mounted.

DMO – WRONG VOLUME
The volume label and the label specified in the command did not match.

B.9 INITIALIZE

These are the error messages created by the INITIALIZE command. The functions of the INITIALIZE command are described in the *TRAX Support Environment User's Guide* and Chapter 11 of this manual.

INI – BAD BLOCK FILE CORRUPT – DATA IGNORED
Although automatic bad block recognition was selected, the bad block data on the disk was not in the correct format, and was therefore ignored.

INI – BAD BLOCK FILE FULL
The disk had more than 102 bad regions on it.

INI – BAD BLOCK HEADER I/O ERROR
An error was detected in writing out the bad-block file header.

INI – BLOCK(S) EXCEED VOLUME LIMIT
The specified block (or blocks) exceeded the physical size of the volume.

INI – BOOT BLOCK WRITE ERROR
An error was detected in writing out the volume boot block.

INI – CHECKING DDnn
Not an error message. An automatic bad-block specification was proceeding, using the bad-block file provided by the Bad Block Locator utility program or, on an RK07, the factory-written file from the last track of the disk.

INI – CHECKPOINT FILE HEADER I/O ERROR
An error was detected in writing out the checkpoint file header.

INI – DATA ERROR
The command specified a bad-block number or contiguous region that was too large.

INI – DISK IS ALIGNMENT CARTRIDGE
THE LAST TRACK ON AN RK07 identified the volume as an alignment cartridge, which cannot be initialized as a files-11 volume. An alignment cartridge is specifically formatted for aligning disk read/write heads.

INI – DUPLICATE BLOCK(S) FOUND
A block that had been defined as bad was being defined as bad a second time.

INI – FAILED TO READ BAD BLOCK FILE
The command was unable to read the bad block information from the last track of an RK07 disk.

INI – HOME BLOCK ALLOCATE WRITE ERROR
In overwriting a bad-home-block area, a write error occurred.

INI – HOME BLOCK WRITE ERROR
An error was detected in writing out the volume home block.

INI – INDEX FILE BIT MAP I/O ERROR
An error was detected in writing out the index-file bit map.

INI – INDEX FILE HEADER I/O ERROR
An error was detected in writing out the index-file header.

INI – MFD FILE HEADER I/O ERROR
An error was detected in writing out the Master File Directory (MFD) file header.

INI – MFD WRITE ERROR
An error was detected in writing out a block in the Master File Directory (MFD).

INI – NO BAD BLOCK DATA FOUND
Although automatic bad-block specification was selected, no bad-block file was found on the volume.

INI – NOT FILES-11 DEVICE
The system does not support files-11 on the specified device.

INI – NULL FILE HEADER I/O ERROR
An error was detected in writing out null-file headers to the index file.

INI – STORAGE BITMAP FILE HEADER I/O ERROR
An error was detected in writing out the storage allocation file header.

INI – VOLUME MOUNTED
An attempt was made to initialize a mounted volume. Mounted volumes can not be initialized.

INI – VOLUME NOT READY
The command specified a volume that was not ready (not up to speed).

INI – VOLUME WRITE LOCKED
The command specified a volume that was write-locked and therefore could not be initialized as a files-11 device.

INI – WARNING BLOCK 0 IS BAD
Block 0 of the specified volume, the boot block, was bad. A bootable image can therefore not be placed on this volume.

B.10 LIBRARIAN

These are the error messages created by the LIBRARIAN command. The functions of the LIBRARIAN command are described in the *TRAX Support Environment User's Guide* and Chapter 11 of this manual.

LBR

Description:
Either the file is not a library file or the file is corrupted.

Suggested User Response:

- If the file is not a library file, reenter the command line with a proper library file specified.
- If the file is a proper library file, the user should run the file structure verification utility (VFY) against the volume to determine if it is corrupted. The functions of the Verify utility are described in the *TRAX System Manager and Operations Guide.*
- If the volume is corrupted, it must be reconstructed before it can be used.

LBR – DUPLICATE ENTRY POINT NAME "name" IN filename

Description:
An attempt has been made to insert a module into a library file when both contain an identically-named entry point.

Suggested User Response:
Determine if the specified input file is the correct file. If not, reenter the command line, specifying the correct input file. If the input file is the correct file, the user may delete the duplicate entry point from the library and rerun.

LBR – DUPLICATE MODULE NAME "name" IN filename

Description:
An attempt has been made to insert (without replacement) a module into a library that already contains a module with the specified name.

Suggested User Response:
Determine if the specified input file is the correct file. If the input file is correct, decide whether to delete the duplicate module from the library file and insert the new one, or replace the duplicate module by rerunning LBR with the /RP switch appended to the input file specifier.

LBR – EPT OR MNT EXCEEDED IN filename

Description:
The EPT or MNT table limit has been reached during the execution of an Insert or Replace command.

Suggested User Response:
Copy the library, increasing the table space via the COMPRESS command. Reenter the command line.

LBR – EPT OR MNT SPACE EXCEEDED IN COMPRESS

Description:
An EPT or MNT table size was specified for the output library file that is not large enough to contain the EPT or MNT entries used in the input library file.

Suggested User Response:
Reenter the command line with a larger EPT or MNT table size specified.

LBR – ERROR IN LIBRARY TABLES, FILE filename

Description:
The library file is corrupted or is not a library file.

Suggested User Response:
If the file is corrupted, no recovery is possible; the file must be reconstructed. If the file is not a library file, reenter the command line with the correct library file specified.

LBR – FATAL COMPRESS ERROR

Description:
The input library file is corrupted or is not a library file.

Suggested User Response:
No recovery is possible. The file in question must be reconstructed.

LBR – INVALID EPT AND/OR MNT SPECIFICATION

Description:
An EPT or MNT value greater than 4096(10) was entered in a CREATE or SQUEEZE function.

Suggested User Response:
Reenter the command line with the correct value specified.

LBR – INVALID FORMAT, INPUT FILE filename

Description:
The format of the specified input file is not the standard format for a macro source or object file, or the input file is corrupted.

Suggested User Response:
Reenter the command line with the correct input file specified.

LBR – NO ENTRY POINT NAMED "name"

Description:
The entry point to be deleted is not in the specified library file.

Suggested User Response:
Determine if the entry point is misspelled or if the wrong library file is specified. Reenter the command line with the entry point correctly specified.

LBR – NO MODULE NAMED "module"

Description:
The module to be deleted is not in the specified library file.

Suggested User Response:
Determine if the module name is misspelled or if the wrong library file is specified. Reenter the command line with the module name correctly specified.

LBR – OPEN FAILURE ON FILE filename

Description:
The file system, while attempting to open a file, has detected an error. One of the following conditions may exist:

- The user directory area is protected against an open.
- A problem exists on the physical device (e.g., device cycled down).
- The volume is not mounted.
- The specified file directory does not exist.
- The file does not exist as specified.
- Insufficient contiguous space to allocate the library file (compress and create only).
- Insufficient dynamic memory in Executive.

Suggested User Response:
Determine which of the above conditions caused the message and correct that condition. Reenter that command line.

LBR – OUTPUT ERROR ON filename

Description:
A write error has occurred on the output file. One of the following conditions may exist:

- The volume is full.
- The device is write-protected.
- The hardware has failed.

Suggested User Response:
If the volume is full, delete all unnecessary files and rerun LBR. If the device is write-protected, write-enable the device, and reenter the command line. If the hardware has failed, swap devices and reenter the command line or wait until the device is repaired and rerun LBR.

B.11 LINK

The functions of the LINK command are described in the *TRAX Linker Reference Manual.* The TRAX Linker produces diagnostic and fatal error messages. Error messages are printed in the following forms:

TKB – *DIAG*-error-message or
TKB – *FATAL*-error-message

Some errors are correctable when command input is from a terminal. In such a case, a diagnostic error message can be printed, the error corrected, and the task building sequence continued. If the same error is detected in an indirect file by the TRAX Linker, a correction cannot be made, and the task linkage operation is aborted.

Some diagnostic error messages merely advise the user of an unusual condition. If the user considers the condition normal to this task, he can run the task image.

The following section tabulates the error messages produced by the Task Builder. Most of the messages are self-explanatory. In some cases, the line in which the error occurred is printed.

A Software Performance Report (SPR) should be submitted to DIGITAL in cases where the explanation accompanying a message refers to a system error.

ALLOCATION FAILURE ON FILE file-name
The TRAX Linker could not acquire sufficient disk space to store the task image file, or did not have write-access to the UFD or volume that was to contain the file.

LOOKUP FAILURE ON FILE filename
invalid-line
The invalid-line printed contains a filename that cannot be located in the directory.

LOOKUP FAILURE ON SYSTEM LIBRARY FILE
The TRAX Linker cannot find the system Library (SY0:[1,1] SYSLIB.OLB) file to resolve undefined symbols.

LOOKUP FAILURE RESIDENT LIBRARY FILE
invalid-line
No symbol table or task image file can be found for the shared region.

MODULE module-name NOT IN LIBRARY
The TRAX Linker could not find the module named on the LB switch in the library.

SEGMENT seg-name HAS ADDR OVERFLOW: ALLOCATION DELETED
Within a segment, the program has attempted to allocate more than 32K. A map file is produced, but no task image file is produced.

TASK HAS ILLEGAL MEMORY LIMITS
An attempt has been made to build a task whose size exceeds the partition boundary. If a task image file was produced, it should be deleted.

TASK IMAGE FILE filename IS NON-CONTIGUOUS
Insufficient contiguous disk space was available to contain the task image. A non-contiguous file was created. After deleting unnecessary files, the /CO switch in PIP should be used to create a contiguous copy.

B.12 LOGIN

The following are the error messages created by the LOGIN command. The functions of the LOGIN command are described in the *TRAX Support User's Programmer's Guide* and Chapter 11 of this manual.

LOG – ACCOUNT FILE OPEN FAILURE
The account file was open for another user; or the disk containing the account file was not mounted. Retry command.

LOG – INVALID ACCOUNT
The name or UIC specified in the command is not stored in the account file; or the password specified does not match the name or UIC given.

LOG – LOGINS ARE DISABLED
The system was in the process of shutting down; or the command SET NOLOGIN has been issued. A user cannot log onto a terminal at these times.

LOG – MESSAGE FILE ERROR nnn.
The system could not open the file [1,2] LOGIN, TXT for a reason indicated by the FCS code nnn.

LOG – OTHER USER LOGGED ON
The issuing terminal was currently logged by another user. Only one user a at a time can be logged onto a terminal.

LOG – TERMINAL ALLOCATED TO OTHER USER
The issuing terminal has been allocated to another user. A user cannot log onto a terminal allocated to someone else.

The system was unable to allocate a system file from the specified block because of intermediate bad blocks or end of volume.

B.13 MERGE

These are the error messages created by the MERGE command. The functions of the MERGE command are described in the *TRAX Support Environment User's Guide* and Chapter 11 of this manual.

NOTE

A question mark "?" preceding the cnv error message indicates a fatal error. A question mark in brackets [?] indicates that the error may be fatal or diagnostic. If no question mark precedes the error message it indicates a diagnostic error.

?cnv – DEVICE OFF LINE – device

Description:
The indicated device exists on the system but the attempt to access it has been prohibited for one of the following reasons.

1. The device is not ready.
2. No volume is mounted on the device.
3. The device is currently reserved by another job.
4. The device requires privileges for ownership and the user does not have privilege.
5. The device has been disabled.

Suggested User Action:
Determine the nature of the problem and take corrective action.

?cnv – DEVICE WRITE PROTECTED – device

Description:
The utility cannot access the indicated device for write operations.

Suggested User Action:
Check the hardware condition of the indicated device. Write enable the unit.

cnv – DEVICE/FILE IS FULL – device/filename

Description:
The utility cannot create an output file on the indicated device because of insufficient space or the indicated file cannot be extended due to insufficient space.

Suggested User Action:
Reenter the command using another device for output files or copy the indicated file to another device and retry the command. Optionally, delete unneeded files on the indicated device and reenter the original command line.

?cnv – DIRECTORY NOT FOUND – filename

Description:
The directory does not exist on the specified device.

Suggested User Action:
Reenter the command with the correct directory specification.

cnv – DUP RCD= string

Description:
The utility could not write a record into an indexed file because duplicate key values for one or more keys in the record were not permitted. Writing the record would cause duplication. The displayed string represents the first 72 characters of the record that could not be written.

Suggested User Action:
None.

cnv – number DUPLICATE RECORDS NOT WRITTEN

Description:
The utility could not write the indicated number of records into an indexed file because duplicate key values for one or more keys were not permitted.

Suggested User Action:
None.

?cnv – FILE NOT AVAILABLE – filename

Description:
The indicated file is being accessed for exclusive use by another job.

Suggested User Action:
Periodically retry the command until the file has been released.

[?] cnv – FILE NOT FOUND – filename

Description:
The indicated file was not found in the designated UFD and device.

Suggested User Action:
Verify the file specification and reenter the command line.

?cnv – FILE READ ERROR

Description:
The utility has encountered a hardware read error on an input or output device.

Suggested User Action:
If not at TRAX Support Environment prompt level, use CTRL/Z to terminate access to utility. Check input and output devices for hardware problems.

?cnv – ILLEGAL DEVICE – device

Description:
The indicated device does not exist.

Suggested User Action:
Reenter the command line with a corrected device specification.

?cnv – INPUT AND OUTPUT RECORD FORMATS DO NOT CORRESPOND

Description:
The user is attempting to write records from one file to another. However, the input file records are variable and the output file records are fixed.

?cnv – INPUT AND OUTPUT RECORD SIZES DO NOT CORRESPOND

Description:
The user is attempting to write records from one file to another. However, one of the following conditions exists:

1. Both files have fixed format records but the fixed size differs.
2. Both files have variable format records but the maximum size of the input file is greater than the maximum size of the output file.

Suggested User Action:
Redefine the output file and retry the command.

?cnv – MAXIMUM RECORD EXCEED – filename

Description:
No more records can be written into the indicated relative file because of the file's maximum number of records attribute.

Suggested User Action:
Create a new relative file through a MACRO-11 program. Specify an appropriate MRN (maximum record number) attribute, Rerun the utility.

?cnv – NO SUCH KEY FOR FILE – value

Description:
The specified key of reference value represents a non-existent key in an indexed file.

Suggested User Action:
Reenter the command with a correct key of reference value.

?cnv – PRIVILEGE VIOLATION – filename

Description:
The user does not have the privileges necessary to access the indicated file.

Suggested User Action:
Have the owner of the file change its privilege specification.

?cnv – RECORD TOO BIG – filename

Description:
A record from the indicated input file exceeds the maximum record size attribute of the output file.

Suggested User Action:
Use the DEFINE utility to create a new file with an appropriate maximum record size.

B.14 MOUNT

These are the error messages created by the MOUNT command. The functions of the MOUNT command are described in the *TRAX Support Environment User's Guide* and Chapter 11 of this manual.

MOU – ACP NOT IN SYSTEM
The task specified as ACP or default ACP was not installed in the system.

MOU – ALREADY MOUNTED
The specified device-unit was already mounted.

MOU – DEVICE ATTACHED[-dev:]
The device-unit specified in the command was attached by a task and could not be mounted. For attempts to mount one or more magnetic tapes, the message includes a specific device-unit.

MOU – DEVICE OFFLINE [-dev:]
The device specified in the command, although generated into the system, was not physically present in the host configuration. If the offline device is a magnetic tape drive, the message includes the device-unit.

MOU – FILE HEADER READ ERROR
Mount could not read either the index file header or the storage allocation file.

MOU – HOME BLOCK READ ERROR
An I/O error was detected in trying to read the home block. This message usually indicates that the volume is not ready. Wait until it is ready and reissue the command.

MOU – MOUNT ERROR FROM ACP xxx.
The ACP detected an error while trying to mount the volume set. See the See Appendix A for a definition of these codes.

MOU – NOT MOUNTABLE DEVICE
The specified device was not supported as a files-11 device (including ANSI magnetic tape) or a network device.

MOU – OTHER VOLUME MOUNTED[-dev:]
An attempt was made to mount a volume on a device that already had a mounted volume. The message specifies the device-unit if it is a tape drive.

MOU – STORAGE BIT MAP FILE READ ERROR
An I/O error was encountered while reading the storage allocation file.

MOU – WRONG VOLUME
The volume label and the label specified in the command did not match.

MOU – VOLUME STRUCTURE NOT SUPPORTED
TRAX did not support the files-11 structure level of the volume being mounted.

B.15 RENAME

These error messages are created by the RENAME command. The functions of the RENAME command are described in *TRAX Support Environment User's Guide* and Chapter 11 of this manual.

REN – CANNOT FIND DIRECTORY FILE

Description:
UFD specified does not exist on this volume.

Suggested User Response:
Reenter the command line, specifying the correct UFD or the correct volume.

REN – CANNOT RENAME FROM ONE DEVICE TO ANOTHER

Description:
You attempted to rename a file across devices.

Suggested User Response:
Reenter the command line, renaming the file on the input volume; then, enter another command to transfer the file to the intended volume.

REN – DIRECTORY WRITE PROTECTED

Description:
REN could not remove an entry from a directory because the device was write-protected, or because of privilege violation.

Suggested User Response:
Enable the unit for write operations or have the owner of the directory change its protection.

B.16 SET

These are the error messages created by the SET command. The functions of the SET COMMAND are described in the *TRAX Support Environment User's Guide* and Chapter 11 of this manual.

SET – DEVICE NOT VARIABLE SPEED MULTIPLEXER
An attempt was made to set the baud rate for a terminal that was not attached to a DZ11 multiplexer.

SET – DEVICE NOT TERMINAL
An attempt was made to set terminal characteristics for a nonterminal device.

SET – INVALID SPEED
The multiplexer line specified does not support the requested speed; or the command specified unequal receive and transmit speeds for a DZ11. The DZ11 does not support split speeds.

SET – LINE NOT DZ11
The command attempted to set to remove a line that was not attached to a DZ11 multiplexer.

B.17 SORT

The functions of the SORT command are described in the *TRAX Sort Reference Manual.* The error messages generated by the SORT command are displayed in two formats.

The first format is:

SORT ERROR – CODE nn
The following is a list of the SORT error codes and a brief explanation of their meaning:

SORT ERROR – CODE 00

Description:
No errors.

SORT ERROR – CODE 01

Description:
Device input error.

SORT ERROR – CODE 02

Description:
Device output error.

SORT ERROR – CODE 03

Description:
OPEN(IN) failure.

SORT ERROR – CODE 04

Description:
OPEN(OUT) failure.

SORT ERROR – CODE 05

Description:
Size of current record is greater than maximum size.

SORT ERROR – CODE 06

Description:
Not enough work area.

SORT ERROR – CODE 07

Description:
RETRN was called after it had exited with a negative error code (end of sort).

SORT ERROR – CODE 10

Description:
SORT routine called out of order. (The order of the calls should be RSORT, RELES, MERGE, RETRN, ENDS.)

SORT ERROR – CODE 11

Description:
Sort already in progress. (To do a second sort, ENDS must be called to clean up the first sort.)

SORT ERROR – CODE 12

Description:
Key size is not positive. Sorts detected a zero or negative key size in its calling parameter.

SORT ERROR – CODE 13

Description:
Record size is not positive.

SORT ERROR – CODE 14

Description:
Key address is not even. (The keys must start at an even address because SORT uses word moves.)

SORT ERROR – CODE 15

Description:
Record address is not even.

SORT ERROR – CODE 16

Description:
Scratch records will be too large (the size of the keys plus the size of the largest record must be less than 37776 octal).

SORT ERROR – CODE 17

Description:
Too few scratch files are given (a minimum of 3 scratch files must be specified).

SORT ERROR – CODE 20

Description:
Too many scratch files are given (a maximum of 10 scratch files may be specified).

SORT ERROR – CODE 21

Description:
End-of-string record was detected where none was expected.

SORT ERROR – CODE 22

Description:
Unexpected end-of-file.

SORT ERROR – CODE 23

Description:
SORT found a record larger than expected.

SORT ERROR – CODE 24

Description:
Record length is not standard for SORTT, SORTA, SORTI.

The second format is:

SRT – control-phase:?message [-RMS-status-code]

where:

control-phase	is the SORT phase in control at the time the error occurred. These values are: C - command decoder M- merge P - presort
message	is a one-line brief explanation of what happened.
RMS-status-code	is a decimal status code returned by RMS for additional information on file errors only. If RMS is not impacted by the SORT error, this status code does not appear.

SRT – C:?SORT COMMAND ERROR

a. Too many input files (more than two, including specification file) or output files (more than one)
b. General snytax error
c. Too many switches
d. Erroneous switches on the specification file
e. An undefined switch

SRT – C:?IMPROPER SWITCH: /FI

a. Less than three or greater than eight scratch files.
b. Invalid terminator

NOTE

Valid terminators are period, comma, slash, equal sign and <CR>. "INVALID TERMINATOR" means that some other character was used as a terminator or that SORT expected to find a terminator where none existed.

SRT – C:?IMPROPER SWITCH: /KE

a. Invalid letter or value
b. Start location or size is 0
c. No period (.) between start location and size
d. Illegal size for data mode
e. Invalid terminator (See NOTE above)

SRT – C:?TOO MANY KEYS

Buffer space overflowed

NOTE

SORT reserves a buffer area for storage of a table based on the input specifications in order to control the processing of each record. This space should be ample for all situations to make this error unlikely.

SRT – C:?NO KEYS SPECIFIED

There are no key switches in the command string and no specification file has been declared.

SRT – C:?KEY AFTER LAST BYTE OF RECORD

The end of an input record key field goes past the stated record size (switch or specification).

SRT – C:?NO /FO SWITCH

You omitted the /FORMAT switch on the input file.

SRT – C:?IMPROPER SWITCH: /FO

You have not specified a valid format type.

SRT – C:?IMPROPER SWITCH: /PR

You specified an invalid sort process.

SRT – C:?INVALID CHARACTER [RMS-Status-Code]

a. Column 6 is not H, I, O, F and record is not ALTSEQ.
b. Process is not SORTR, SORTT, SORTA, SORTI, or blank.
c. Collating sequence is not blank, E, or X.
d. Data type is not B, C, D, F, I, J, K, P, Z.
e. Key type is not D, F, N, O.
f. Logical entry is not A, O, blank, or *.

SRT – C:?ILLEGAL FIELD [RMS-Status-Code]

a. A numeric field in specification contains other than decimal digits or blanks.
b. No key size is given in Header specification.
c. No output size is given in Header Specification if type of SORT is SORTR or SORTT.
d. ALTSEQ is misspelled.
e. ALTSEQ entries do not represent 7-bit octal values.
f. Last location is less than first location in record field identification.
g. Size is invalid for data mode.
h. Sizes of Factors 1 and 2 in Record Specification do not match.
i. Compare relation is undefined.
j. Forced field is other than type C or more than one position.

SRT – C:?ILLEGAL CONSTANT [RMS-Status-Code]

a. Constant given in Factor 2 is greater than 20 characters.
b. Mode of constant does not agree with mode of Factor.
c. Invalid characters appear in constant (e.g., non-digits if the constant is numeric).
d. Sign is omitted from binary or packed constant.

SRT – C:?NO HEADER [RMS-Status-Code]

a. First record of specification file is not an H specification.

SRT – C:?INCORRECT SEQUENCE [RMS-Status-Code]

a. Numeric record sequence is lower than sequence previously encountered.
b. No valid data specification appears when keys are to be stripped from output.
c. Record specification after "include-all" ("include-all" should be last).
d. Key specifications appear after data specifications.

SRT – C:?NO ALTSEQ [RMS-Status-Code]

a. Specification for alternate collation entered in Header column 26 but no ALTSEQ Specifications follow.

SRT – C:?TOO MANY SPECIFICATIONS [RMS-Status-Code]

a. Number of specifications for a particular type of record have overflowed the buffer space.

NOTE

SORT reserves a buffer space for storage of a table based on the input specifications and used to control the processing of each record type. This space should be ample for all situations to make the error unlikely other than in very exceptional circumstances.

APPENDIX C
TRAX I/O ERROR CODES

The I/O error codes return to TRAX tasks are listed below:

Mnemonic	Decimal	Octal	
.BAD	-1	377	Bad parameters
.IFC	-2	376	Invalid function code
.DNR	-3	375	Device not ready
.VER	-4	374	Parity error on device
.ONP	-5	373	Hardware option not present
.SPC	-6	372	Illegal user buffer
.DNA	-7	371	Device not attached
.DAA	-8	370	Device already attached
.DUN	-9	367	Device not attachable
.EOF	-10	366	End-of-file detected
.EOV	-11	365	End-of-volume detected
.WLK	-12	364	Write attempted to looked unit
.DAO	-13	363	Data overrun
.SRE	-14	362	Send/receive failure
.ABO	-15	361	Request terminated
.PRI	-16	360	Privilege violation
.RSU	-17	357	Shareable resource in use
.OVR	-18	356	Illegal overlay request
.BYT	-19	355	Odd byte count (or virtual address)
.BLK	-20	354	Logical block number too large
.MOD	-21	353	Invalid UDC module number
.CON	-22	352	UDC connect error
.NOD	-23	351	System dynamic memory exhausted
.DFU	-24	350	Device full
.IFU	-25	347	Index file full
.NSF	-26	346	No such file
.LCK	-27	345	Locked from read/write access
.HFU	-28	344	File header full
.WAC	-29	343	Accessed for write
.CKS	-30	342	File header checksum failure
.WAT	-31	341	Attribute control list format error
.RER	-32	340	File processor device read error
.WER	-33	337	File processor device write error
.ALN	-34	336	File already accessed on LUN
.SNC	-35	335	File ID, file number check
.SQC	-36	334	File ID, sequence number check
.NLN	-37	333	No file accessed on LUN

Mnemonic	Decimal	Octal	
.CLO	-38	332	File was not properly closed
.NBF	-39	331	No buffer space available for file
.RBG	-40	330	Illegal record size
.NBK	-41	327	File exceeds space allocated, no blocks
.ILL	-42	326	Illegal operation on file descriptor block
.BTP	-43	335	Bad record type
.RAC	-44	324	Illegal record access bits set
.RAT	-45	323	Illegal record attributes bits set
.RCN	-46	322	Illegal record number – too large
	-47		(not used)
.2DV	-48	320	Rename – 2 different devices
.FEX	-49	317	Rename – a new file name already in use
.BDR	-50	316	Bad directory file
.RNM	-51	315	Cannot rename old file system
.BDI	-52	314	Bad directory syntax
.FOP	-53	313	File already open
.BDV	-55	311	Bad device name
.BBE	-56	310	Bad block on device
.DUP	-57	307	Enter – duplicate entry in directory
.STK	-58	306	Not enough stack space (FCS or FCP)
.FHE	-59	305	Fatal hardware error on device
.NFI	-60	304	File ID was not specified
.ISQ	-61	303	Illegal sequential operation
.EOT	-62	302	End-of-tape detected
.BVR	-63	301	Bad version number
.BHD	-64	300	Bad file header
.OFL	-65	277	Device offline
.BCC	-66	276	Block check or CRC error
	-67		(not used)
.NNN	-68	274	No such node
.NFW	-69	273	Path lost to partner
.BLB	-70	272	Bad logical buffer
.TMM	-71	271	Too many outstanding messages
.NDR	-72	270	No dynamic space available
.CNR	-73	267	Connection rejected
.TMO	-74	266	Time out on request
.ECP	-75	265	File expiration date not reached
.BTF	-76	264	Bad tape format
.NNC	-77	263	Not ANSI "D" format byte count
.NNL	-78	262	Not a network LUN
.NLK	-79	261	Task not linked to specified ICS/ICR interrupts
.NST	-80	260	Specified task not installed

Mnemonic	Decimal	Octal	
.FLN	-81	257	Device offline when offline request was issued
.IES	-82	256	Invalid escape sequence
.PES	-83	255	Partial escape sequence
.ALC	-84	254	Allocation failure
.ULK	-85	253	Unlock error

APPENDIX D
RMS COMPLETION STATUS CODES

This appendix describes completion status codes that can be returned by RMS-11 to your program.

All RMS-11 file and record operations return a completion status code into the status field (STS) of the control block (i.e., FAB or RAB) associated with the operation. A symbolic name is defined for each such code. The symbolic names of successful completion status codes take the following form:

SU$xxx

where

xxx is a mnemonic value describing the successful operation.

Symbolic names for error completion status codes take the form:

ER$xxx

where

xxx is a mnemonic value representing the reason the operation failed.

For certain error conditions, RMS-11 uses the status value (STV) field to communicate additional information to your program. The tables in this appendix list all instances in which a particular symbolic value in the STS field indicates the presence of further information in the STV field. A limited number of severe error conditions cause RMS-11 to invoke a fatal error crash routine. Section D.1 of this appendix describes these conditions and the crash routine itself.

The sections that follow describe, respectively, successful completion status codes, error completion status codes, and the RMS-11 fatal error crash routine.

D.1 SUCCESSFUL COMPLETION STATUS CODES

Table D-1 describes successful completion status codes returned by RMS-11 routines.

Table D-1 Successful Completion Status Codes

Symbolic Name	Decimal Value	Description
SU$SUC	1	Operation successful.
SU$DUP	2	A record written into an indexed file as a result of a $PUT or $UPDATE operation contains at least one key value that was already present in another record.
SU$IDX	3	During a $PUT or $UPDATE operation on an indexed file, the record was successfully written. The record can be subsequently retrieved but RMS-11 was not able to optimize the structure of the index at the time the record was inserted. Several indirections will occur, therefore, on retrieval. In some instances, RMS-11 may also return an error code (e.g., ER$RLK) in the STV field of the control block.
SU$RRV	4	During a $PUT or $UPDATE operation on an indexed file, the record was successfully written. However, RMS-11 was unable to update one or more Record Retrieval Vectors (RRVs) and the records associated with the RRVs cannot be retrieved using alternate indexes or RFA addressing mode.

D.2 ERROR COMPLETION STATUS CODES

Table D-2 describes error completion status codes returned by RMS-11 routines.

Table D-2 Error Completion Status Codes

Symbolic Value	Octal Value	Decimal Value	STV	Description
ER$ABO	177760	-16	ER$STK or ER$MAP	Operation aborted: out of stack save area or in core data structures corrupted.
ER$ACC	177740	-32	Kernel error code	Kernel file system could not access the file.
ER$ACT	177720	-48		File activity precludes action (e.g., attempting to close a file with outstanding asynchronous record operation).
ER$AID	177700	-64	XAB address	Bad area identification number (AID) field in allocation XAB (i.e., out of sequence).
ER$ALN	177660	-80	XAB address	Illegal value in alignment boundary type (ALN) field of allocation XAB.
ER$ALQ	177640	-96	(XAB address)	Value in allocation quantity (ALQ) field in FAB (or allocation XAB) exceeds maximum or, during an explicit $EXTEND operation, equals zero.
ER$ANI	177620	-112		Records in a file on ANSI labeled magnetic tape are variable length but not in ANSI D format.
ER$AOP	177600	-128	XAB address	Illegal value in allocation options (AOP) field in allocation XAB.
ER$AST	177560	-144		Invalid operation at AST level: attempting to issue a synchronous operation from an asynchronous record operation completion routine.
ER$ATR	1777540	-160	Kernel error code	Read error on file header attributes.
ER$ATW	177520	-176	Kernel error code	Write error on file header attributes.
ER$BKS	177500	-192		Bucket size (BKS) field in FAB contains value exceeding maximum.
ER$BKZ	177460	-208	XAB address	Bucket size (BKZ) field in allocation XAB contains value exceeding maximum.
ER$BLN	177440	-224		Block length (BLN) field in a FAB or RAB is incorrect.

Table D-2 (Cont.) Error Completion Status Codes

Symbolic Value	Octal Value	Decimal Value	STV	Description
ER$BOF	177430	-232		Beginning of file detected on $SPACE operation to magnetic tape file.
ER$BPA	177420	-240		Private buffer pool address not a double word boundary.
ER$BPS	177400	-256		Private buffer pool size not a multiple of 4.
ER$BUG	177360	-272		Internal error detected in RMS-11 (refer to Section D.4 of this Appendix); no recovery possible; contact a Software Specialist.
ER$CCR	177340	-288		Can't connect RAB (i.e., only one record access stream permitted for sequential files).
ER$CHG	177320	-304		$UPDATE attempting to change a key field that does not have the change attribute.
ER$CHK	177300	-320		Index file bucket check-byte mismatch. The bucket has been corrupted. No recovery possible for the bucket.
ER$COD	1777240	-352	XAB address	Invalid COD field in XAB or XAB type is illegal for the organization or operation.
ER$CRE	177220	-368	Kernel error code	Kernel file system could not create file.
ER$CUR	177200	-384		No current record: operation not immediately preceded by a successful $GET or $FIND.
ER$DAC	177160	-400	Kernel error code	Kernel file system deaccess error during $CLOSE.
ER$DAN	177140	-416	XAB address	Invalid area number in DAN field of key definition XAB.
ER$DEL	177120	-432		Record accessed by RFA access mode has been deleted.
ER$DEV	177100	-448		1. Syntax error in device name. 2. No such device. 3. Inappropriate device for operation (e.g., attempting to create an indexed file on magnetic tape).
ER$DIR	177060	-464		Syntax error in directory name.

Table D-2 (Cont.) Error Completion Status Codes

Symbolic Value	Octal Value	Decimal Value	STV	Description
ER$DME	177040	-480		Dynamic memory exhausted: insufficient space in central space pool or private buffer pool.
ER$DNF	177020	-496		Directory not found.
ER$DNR	177000	-512		Device not ready.
ER$DPE	176770	-520	Kernel error code	Device positioning error.
ER$DUP	176740	-544		Duplicate key detected, duplicates allowed attribute not set for one or more key fields.
ER$ENT	176720	-560	Kernel error code	Kernel file system enter function failed.
ER$ENV	176700	-576		Environment error: operation or file organization not specified in ORG$ macro.
ER$EOF	176660	-592		End of file.
ER$ESS	176640	-608		Expanded string area in NAM block too short.
ER$EXP	176630	-616		File expiration date not reached.
ER$EXT	176620	-624	Kernel error code	File extend failure.
ER$FAB	176600	-640		Not a valid FAB: BID field does not contain FB$BID.
ER$FAC	176560	-656		1. Record operation attempted was not declared in FAC field of FAB at open time. 2. Invalid contents in FAC field. 3. FB$PUT not present in FAC for $CREATE operation.
ER$FEX	176540	-672		File already exists (attempted $CREATE operation).
ER$FID	177530	-680		Invalid file id.
ER$FLG	176520	-688	XAB address	Invalid combination of values in FLG field of key definition XAB (e.g., no duplicates and keys can change).
ER$FLK	176500	-704		File locked by another user – you cannot access the file because your sharing specification cannot be met.

Table D-2 (Cont.) Error Completion Status Codes

Symbolic Value	Octal Value	Decimal Value	STV	Description
ER$FND	176460	-720	Kernel error code	Kernel file system Find function failed.
ER$FNF	176440	-736		File not found.
ER$FNM	176420	-752		Syntax error in file name.
ER$FOP	176400	-768		Invalid file options.
ER$FUL	176360	-784		Device full: can't create or extend file.
ER$IAN	176340	-800	XAB address	Invalid area number in IAN field of key definition XAB.
ER$IDX	176320	-816		Index not initialized (this code can only occur in the STV field when STS contains ER$RNF).
ER$IFI	176300	-832		Invalid IFI field in FAB.
ER$IMX	176260	-848	XAB address	Maximum number (254) of key definition or allocation XABs exceeded or multiple summary, protection, or date XABs present during operation.
ER$INI	176240	-864		$INIT or $INITIF macro call never issued.
ER$IOP	176220	-880		Illegal operation; examples include: 1. Attempting a $TRUNCATE operation to a non-sequential file. 2. Attempting an $ERASE or $EXTEND operation to a magnetic tape file. 3. Issuing a block mode operation (e.g., $READ or $WRITE) to a stream not connected for block operations. 4. Issuing a record operation (e.g., $GET, $PUT) to a stream connected for block mode operations.
ER$IRC	176200	-896		Illegal record encountered in sequential file: invalid count field.
ER$ISI	176160	-912		Invalid internal stream identifier (ISI) field in RAB (field may have been altered by user) or $CONNECT never issued for stream.

Table D-2 (Cont.) Error Completion Status Codes

Symbolic Value	Octal Value	Decimal Value	STV	Description
ER$KBF	176140	-928		Key buffer address (KBF) field equals 0.
ER$KEY	176120	-944		Record identifier (i.e., the 4-byte location addressed by KBF) for random operation to relative file is 0 or negative.
ER$KRF	176100	-960		Invalid key of reference (KRF) in RAB: 1) As input to random $GET or $FIND operation, or 2) As input to $CONNECT or $REWIND (in this case, ER$KRF is returned for the first record operation following the $CONNECT or $REWIND.
ER$KSZ	176060	-976		Key size equals zero or too large (indexed file) or not equal to 4 (relative file).
ER$LAN	176040	-992	XAB address	Invalid area number in LAN field of key definition XAB.
ER$LBL	176020	-1008		Magnetic tape is not ANSI labeled.
ER$LBY	176000	-1040		Invalid value in logical channel number (LCH) field of FAB.
ER$LEX	175750	-1048	XAB address	Attempt to extend an area containing an unused extent.
ER$LOC	175740	-1056	XAB address	Invalid value in LOC field of allocation XAB.
ER$MAP	175720	-1072		In core data structures (e.g., I/O buffers) corrupted. This code can only occur in the STV field when STS contains ER$ABO.
ER$MKD	175700	-1088	Kernel error code	Kernel file system could not mark file for deletion.
ER$MRN	175660	-1104		1. Maximum record number field contains a negative value during $CREATE of relative file. 2. Record identifier (pointed to by KBF) for random operation to relative file exceeds maximum record number specified when file created.

Table D-2 (Cont.) Error Completion Status Codes

Symbolic Value	Octal Value	Decimal Value	STV	Description
ER$MRS	175640	-1120		Maximum record size (MRS) field contains 0 during $CREATE operation and: 1. Record Format is fixed, or 2. File organization is relative.
ER$NAM	175620	-1136		Odd address in Name Block address (NAM) field in FAB on $OPEN, $CREATE, or $ERASE.
ER$NEF	175600	-1152		Not at end-of-file: attempting a $PUT operation to a sequential file when stream is not positioned to EOF.
ER$NID	175560	-1168		Can't allocate internal index descriptor: insufficient room in space pool while attempting to open an indexed file.
ER$NPK	175540	-1184		No primary key definition XAB present during $CREATE of indexed file.
ER$ORD	175500	-1216	XAB address	XABs in chain not in correct order: 1. Allocation or key definition XABs not in ascending (or densely ascending) order. 2. XAB of another type intervenes in key definition or allocation XAB sub-chain.
ER$ORG	175460	-1232		Invalid value in file organization (ORG) field of FAB.
ER$PLG	175440	-1248		Error in file's prologue: file is corrupted and must be reconstructed.
ER$POS	175420	-1264	XAB address	Key position (POS) field in key definition XAB contains a value exceeding maximum record size.
ER$PRM	175400	-1280	XAB address	File header contains bad date and time information (retrieved by RMS-11 because a date and time XAB is present during a $OPEN or $DISPLAY operation); file may be corrupted.
ER$PRV	175360	-1296		Privilege violation: access to the file denied by the operating system.

Table D-2 (Cont.) Error Completion Status Codes

Symbolic Value	Octal Value	Decimal Value	STV	Description
ER$RAB	175340	-1312		Not a valid RAB: BID field does not contain RB$BID.
ER$RAC	175320	-1328		1. Illegal values in record access mode (RAC) field of RAB. 2. Illogical value in RAC field (e.g., RB$KEY with a sequential file).
ER$RAT	175300	-1344		1. Illegal values in record attributes (RAT) field of FAB during $CREATE. 2. Illogical combination of attributes (e.g., FB$CR and FB$FTN) in RAC field during $CREATE.
ER$RBF	175260	-1360		Record address (RBF) field in RAB contains an odd address (block mode access only).
ER$RER	175240	-1376	Kernel error code	File read error.
ER$REX	175220	-1392		Record already exists: during a $PUT operation in random mode to a relative file, an existing record found in the target record position.
ER$RFA	175200	-1408		Invalid RFA in RFA field of RAB during RFA access.
ER$RFM	175160	-1424		1. Invalid record format in RFM field of FAB during $CREATE. 2. Specified record format is illegal for file organization.
ER$RLK	175140	-1440		Target bucket locked by another task or another stream in the same program.
ER$RMV	175120	-1456	Kernel error code	Kernel file system Remove function failed.
ER$RNF	175100	-1472	(ER$IDX)	Record identified by KBF/KSZ fields of RAB for random $GET or $FIND operation does not exist in relative or indexed file (for indexed files only, STV may contain ER$IDX). Record may never have been written or may have been deleted.

Table D-2 (Cont.) Error Completion Status Codes

Symbolic Value	Octal Value	Decimal Value	STV	Description
ER$RNL	175060	-1488		$FREE operation issued but no bucket was locked by stream.
ER$ROP	175040	-1504		Record options (ROP) field contains illegal values or illogical combination of values.
ER$RPL	175020	-1520	Kernel error code	Error while reading prologue.
ER$RRV	175000	-1536		Invalid RRV record encountered in indexed file; file may be corrupted.
ER$RSA	174760	-1552		Record stream active, i.e., in asynchronous environment, attempting to issue a record operation to a stream that has a request outstanding.
ER$RSZ	174740	-1568		Record size specified in RSZ of RAB during SPUT or $UPDATE is invalid: 1. RSZ equals zero. 2. RSZ exceeds maximum record size (MRS) specified when file created. 3. RSZ not equal to size of Current Record for $UPDATE operation to a sequential file on disk. 4. RSZ does not equal MRS (for fixed format records).
ER$RTB	174720	-1584	Actual record size	Record too big for user's buffer: RMS-11 could not move entire record retrieved by $GET operation to user work area (UBF/USZ). Note that this error does not destroy the current context of the stream. Rather, the stream's context is updated as if the operation had been completely successful.
ER$SEQ	174700	-1600		During $PUT operation, key of record to be written is not equal to or greater than key of previous record (and RAC field contains RB$SEQ).
ER$SHR	174660	-1616		Illogical value in SHR field of FAB (e.g., FB$WRI specified for sequential file).

Table D-2 (Cont.) Error Completion Status Codes

Symbolic Value	Octal Value	Decimal Value	STV	Description
ER$SIZ	174640	-1632	XAB address	Invalid SIZ field in key definition XAB during $CREATE (e.g., specified size exceeds maximum record size).
ER$STK	174620	-1648		During asynchronous record operation, RMS-11 has found that the stack is too big to be saved (this code can only occur in the STV field when STS contains ER$ABO).
ER$SYS	174600	-1664	Directive or QIO status code	System directive error.
ER$TRE	174560	-1680		Index tree error: indexed file is corrupted.
ER$TYP	174540	-1696		Syntax error in file type (e.g., more than 3 characters specified).
ER$UBF	174520	-1712		Invalid address in UBF field of RAB: 1. UBF contains 0, or 2. UBF not word aligned (for block mode access only).
ER$USZ	174500	-1728		Invalid USZ field in RAB (i.e., USZ contains 0).
ER$VER	174460	-1744		Syntax error in file version number.
ER$VOL	174440	-1760	XAB address	Invalid VOL field in allocation XAB (i.e., VOL does not contain 0).
ER$WER	174420	-1776	Kernel error code	File write error.
ER$WLK	174410	-1784		Device is write locked.
ER$WPL	174400	-1792	Kernel error code	Error while writing prologue.
ER$XAB	174360	-1808	(XAB address)	XAB field in FAB (or NXT field in XAB) contains an odd address.

D.3 FATAL ERROR CRASH ROUTINE

RMS-11 issues a BPT instruction whenever it encounters inconsistent internal FAB or RAB). This action is taken only when RMS-11 cannot continue processing, since to do so might cause damage to user files or the user's task image. As an example, when the problem is caused by an invalid FAB or RAB, RMS-11 cannot return an error status code in STS since it has no recognizable user control block to work with.

The BPT instruction generated as a result of fatal errors is in the R0RMSA module of RMS-11. The following is the state of the general registers at the time this instruction is issued:

R0 = RMS-11 error code
R1 = Entry SP value
R2 = Entry return PC
R3 = Address of system impure area

General registers R1 and R2 are always valid if the crash routine is invoked by a fatal user call error. When the crash routine is invoked by inconsistent internal conditions, the contents of general registers R1 and R2 may be meaningless if RMS-11 was executing an asynchronous RAB operation.

The following subsections summarize, respectively, the fatal user call errors and the RMS-11 inconsistent internal conditions that can cause invocation of the fatal error crash routine.

D.4 FATAL USER CALL ERRORS

When the fatal error crash routine is invoked because of a user call error, general register R0 contains one of the following error codes:

- ER$FAB
- ER$RAB

These error codes indicate that the user called RMS-11 using a control block that was not a valid FAB (for file operations such as $OPEN, $CREATE, etc.) or RAB (for record operations such as $CONNECT, $GET, $PUT, etc.). This condition can occur for any one of the following reasons:

1. ;The address of the FAB or RAB is 0.
2. The address of the FAB or RAB is odd.
3. ;The control block's BID field does not contain the proper block identifier code (i.e., FB$BID for FABs and RB$BID for RABs).

D.5 RMS-11 INCONSISTENT INTERNAL CONDITIONS ERRORS

When the crash routine is invoked because of RMS-11 inconsistent internal conditions, general register R0 contains one of the following error codes:

- ER$BUG
- ER$MAP

These error codes indicate internal problems with RMS-11 and are considered fatal. They can be caused by improper coding by the user (e.g.,

destroying some internal RMS-11 data base), but are also used to detect RMS-11 bugs. When one of the above error codes is encountered, the user should provide, if possible, the following information to DEC with an SPR:

1. The contents of the general registers.
2. The first ten words, at a minimum, or all words upon the system stack.
3. The operation the program was performing (e.g., $OPEN, $GET, $PUT).
4. The organization of the file being processed.
5. A load map of the task.
6. If running on TRAX, a post-mortem dump.

INDEX

TRAX System
Manager's Guide
AA-D332A-TC

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Did you find errors in this manual? If so, specify the error and the page number.

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify)_____

Name_____ Date_____

Organization_____

Street_____

City_____ State_____ Zip Code_____
or
Country

---------- Fold Here ----------

---------- Do Not Tear - Fold Here and Staple ----------

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Software Documentation
146 Main Street ML 5-5/E39
Maynard, Massachusetts 01754