IAS PDS User's Guide

Order Number: AA–H003C–TC

This manual describes the IAS Program Development System (PDS) interface to the IAS operating system.

Operating System and Version: IAS Version 3.4

May 1990

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

This document was prepared using VAX DOCUMENT, Version 1.2

Contents

PREFACE xv

CHAPTER 1 INTRODUCTION TO IAS 1–1

1.1 INTERACTIVE APPLICATIONS SYSTEM 1–1
1.1.1 Interactive Processing 1–1
1.1.2 Batch Processing 1–1
1.1.3 Real-Time Processing 1–1

1.2 DIFFERENT TYPES OF IAS SYSTEM 1–2
1.2.1 Real-Time System 1–2
1.2.2 Multiuser System 1–3
1.2.3 Timesharing System 1–3

1.3 THE USER INTERFACE 1–3
1.3.1 The Program Development System (PDS) 1–4

1.4 SYSTEM ACCESS SECURITY 1–5

1.5 PROGRAMMING LANGUAGES 1–5
1.5.1 MACRO-11 1–5

CHAPTER 2 KEYBOARD OPERATION 2–1

2.1 THE INTERACTIVE TERMINAL 2–1
2.1.1 Typing Input 2–1
2.1.2 Keyboard Functions 2–1
2.1.3 Control Character Functions 2–4

2.2 CORRECTING INPUT ERRORS 2–5
2.2.1 Cancelling a PDS Command 2–5
2.2.2 Deleting Characters 2–5
2.2.3 Deleting a Line 2–6

2.2.4 Using Upper- and Lowercase ____ 2–6

CHAPTER 3 A SAMPLE INTERACTIVE SESSION 3–1

3.1 INTRODUCTION 3–1

3.2 SAMPLE INTERACTIVE SESSION 3–1

3.3 LOGGING IN 3–3
3.3.1 Control C ____ 3–3
3.3.2 The Username ____ 3–3
3.3.3 The Password ____ 3–3
3.3.4 The PDS Prompt ____ 3–4

3.4 THE CREATE COMMAND 3–4
3.4.1 Correcting Input Errors ____ 3–4
3.4.2 Canceling a Line ____ 3–5
3.4.3 Closing the New File ____ 3–5

3.5 THE TYPE COMMAND 3–5

3.6 THE FORTRAN COMMAND 3–5

3.7 THE LINK COMMAND 3–6

3.8 THE RUN COMMAND 3–7

3.9 THE DIRECTORY COMMAND 3–7

3.10 THE RENAME COMMAND 3–8

3.11 THE DIRECTORY/BRIEF COMMAND 3–8

3.12 THE LOGOUT COMMAND 3–9

CHAPTER 4 ISSUING PDS COMMANDS 4–1

4.1 INTRODUCTION 4–1

4.2 PDS COMMANDS 4–1
4.2.1 Command Strings 4–1
4.2.2 Abbreviated Input 4–2
4.2.3 Underline Convention 4–2

4.3 COMMAND PARAMETERS 4–2
4.3.1 Parameter Lists 4–2
4.3.2 Parameter Prompts 4–3
4.3.3 Optional Parameters 4–3

4.4 COMMAND AND PARAMETER QUALIFIERS 4–3

4.5 INVALID COMMANDS OR SYNTAX 4–4
4.5.1 Active Tasks 4–4
4.5.2 Subsystems 4–4
4.5.3 Errors 4–4
4.5.4 Length of Command Line 4–5

4.6 PDS COMMAND PRIVILEGE 4–6
4.6.1 PDS Command Masks 4–6

4.7 PDS TIMESHARING TASK PRIVILEGE 4–10

4.8 USE OF CTRL/C 4–10
4.8.1 Effect of CTRL/C on Type-ahead 4–11
4.8.2 Effect of CTRL/C on Indirect Command Files 4–11

4.9 PDS DIALUP SUPPORT 4–11

CHAPTER 5 BATCH PROCESSING 5–1

5.1 INTRODUCTION 5–1

5.2 BATCH COMMANDS 5–1

5.3 BEGINNING AND ENDING A BATCH JOB 5–1
5.3.1 The $JOB Command 5–2
5.3.2 The $EOJ Command 5–2

5.4 THE SUBMIT COMMAND 5–2

5.5 BATCH EDITING 5–3

CHAPTER 6 FILE HANDLING 6–1

6.1 INTRODUCTION 6–1

6.2 THE IAS FILE SYSTEM 6–1
6.2.1 Volumes 6–1
6.2.2 File and Volume Protection 6–2
6.2.3 RMS-11 Files Management in IAS 6–3

6.3 FILE SPECIFICATIONS 6–3
6.3.1 File Specification Defaults 6–6
6.3.1.1 Changing Default Values • 6–7
6.3.1.2 Displaying Default Values (Timesharing Systems Only) • 6–7
6.3.2 Wildcards 6–8
6.3.2.1 Input Files • 6–8
6.3.2.2 Output Files • 6–8
6.3.3 Valid File Specifications 6–9

6.4 DEVICE MANAGEMENT 6–9
6.4.1 System Devices 6–9
6.4.2 Accessing a Non-System Device 6–10
6.4.2.1 Logical Device Names • 6–10
6.4.2.2 Logical Units • 6–10
6.4.3 Mounting a Volume on a Device 6–11
6.4.4 Dismounting a Volume 6–12
6.4.5 Allocating a Device (Timesharing Systems Only) 6–12
6.4.6 Deallocating a Device (Timesharing Systems Only) 6–13
6.4.7 Assigning Logical Unit Numbers to a Device (Timesharing Systems Only) 6–13

6.5 FILE MANAGEMENT 6–14
6.5.1 Creating Files 6–14
6.5.1.1 Using the Editor to Create a Sequential File • 6–14
6.5.1.2 User File Directories • 6–14
6.5.2 Manipulating Files 6–15
6.5.2.1 The APPEND Command • 6–15
6.5.2.2 The COPY Command • 6–16
6.5.2.3 The RENAME Command • 6–18
6.5.2.4 The MERGE Command • 6–18
6.5.3 Listing Files 6–18
6.5.3.1 Listing Files on the Line Printer • 6–18
6.5.3.2 Printing Files on Varied Stationery • 6–19
6.5.3.3 Listing Files at an Interactive Terminal • 6–19
6.5.3.4 The DUMP Facility • 6–19
6.5.4 Deleting Files 6–20

CHAPTER 7 IAS EDITORS 7–1

7.1 INTRODUCTION 7–1

7.2 THE TEXT EDITOR (EDI) 7–1

7.3 BATCH EDITING 7–1

7.4 THE DEC EDITOR (EDT) 7–1

7.5 THE KEYPAD EDITOR (KED OR K52) 7–1

CHAPTER 8 INTRODUCTION TO PROGRAM CONTROL 8–1

8.1 INTRODUCTION 8–1

8.2 PROCESSING MODES 8–1

8.3 INDIRECT COMMAND FILES 8–1

8.4 USER LIBRARIES 8–2

8.4.1 Macro Libraries 8–2
8.4.2 Object Module Libraries 8–2

8.5 CREATING SOURCE FILES 8–3

8.6 THE CREATE COMMAND 8–3
8.6.1 The EDIT Command 8–4

8.7 ERROR STATUS RETURNED TO PDS 8–5
8.7.1 Conditional Command Execution 8–5

CHAPTER 9 BASIC-11 9–1

9.1 INTRODUCTION 9–1

9.2 THE BASIC COMMAND 9–1

9.3 CTRL/C 9–1

9.4 TERMINATING A BASIC SESSION 9–2

9.5 EXAMPLE 9–2

CHAPTER 10 COBOL 10–1

10.1 INTRODUCTION 10–1

10.2 CREATING SOURCE FILES 10–1

10.3 THE COBOL COMMAND 10–1
10.3.1 Compiling COBOL Source Files 10–1
10.3.2 COBOL Command Qualifiers 10–2
10.3.3 COBOL Compiler Switches 10–2
10.3.4 Compiler Error Messages 10–3

10.4 LINKING OBJECT FILES 10–3
10.4.1 The LINK Command 10–3
10.4.1.1 Options • 10–3
10.4.1.2 Object Module Libraries • 10–4
10.4.1.3 Output Files • 10–4
10.4.1.4 Example • 10–4

10.5 RUNNING THE TASK 10–5

CHAPTER 11 FORTRAN 11–1

11.1 INTRODUCTION 11–1

11.2 CREATING SOURCE FILES 11–1

11.3 THE FORTRAN COMMAND 11–1
11.3.1 Compiling Source Files 11–2
11.3.2 FORTRAN Command Qualifiers 11–2
11.3.3 FORTRAN Compiler Switches 11–2
11.3.4 Examples 11–3

11.4 LINKING OBJECT FILES 11–3
11.4.1 The LINK Command 11–3
11.4.1.1 Options • 11–3
11.4.1.2 Object Modules • 11–4
11.4.1.3 Output Files • 11–4
11.4.1.4 Example • 11–4

11.5 RUNNING THE TASK 11–5

CHAPTER 12 MACRO-11 12–1

12.1 INTRODUCTION 12–1

12.2 CREATING SOURCE FILES 12–1

12.3 THE MACRO COMMAND 12–1

12.4 ASSEMBLING MACRO-11 SOURCE FILES 12–1
12.4.1 MACRO-11 Command and File Qualifiers 12–2

12.5 LINKING OBJECT FILES 12–2
12.5.1 The LINK Command 12–2
12.5.1.1 Options • 12–3
12.5.1.2 Object Module Libraries • 12–3
12.5.1.3 Output Files • 12–3
12.5.1.4 Example • 12–4

12.6 RUNNING THE TASK 12–4

12.7 DEBUGGING 12–4
12.7.1 The Online Debugging Technique 12–4
12.7.2 User-Written Debugging Aids 12–5

CHAPTER 13 CORAL 66 13–1

13.1 INTRODUCTION 13–1

13.2 CREATING SOURCE FILES 13–1

13.3 THE CORAL COMMAND 13–1
13.3.1 Compiling Source Files 13–1
13.3.2 CORAL Command Qualifiers 13–2
13.3.3 Examples 13–2

13.4 LINKING OBJECT FILES 13–3
13.4.1 The LINK Command 13–3
13.4.1.1 Options • 13–3
13.4.1.2 Object Modules • 13–4
13.4.1.3 Output Files • 13–4
13.4.1.4 Example • 13–4

13.5 RUNNING THE TASK 13–4

CHAPTER 14 PDS COMMAND DESCRIPTIONS 14–1

14.1 INTRODUCTION 14–1

14.2 PDS COMMAND FORMAT 14–1

14.3 PDS COMMAND DESCRIPTIONS 14–3

14.4 PDS COMMAND LIBRARY 14–3

ABORT 14–5
ALLOCATE 14–7
APPEND 14–9
ASSIGN 14–12
BASIC 14–14
CANCEL 14–16
COBOL 14–18
COMPARE 14–24
CONTINUE 14–27
COPY 14–30
CORAL 14–35
CREATE 14–39
DCL 14–45
DEALLOCATE 14–46
DEASSIGN 14–48
DELETE 14–50
DIRECTORY 14–54
DISABLE 14–57
DISMOUNT 14–59
DUMP 14–62
EDIT 14–66
ENABLE 14–69
$EOD 14–70
$EOJ 14–72
FIX 14–73
FORTRAN 14–75
GOTO 14–80
HELP 14–82
IDENTIFY 14–84
INITIALIZE 14–86
INSTALL 14–91
$JOB 14–94
LIBRARIAN 14–96
LINK 14–108
LOGOUT 14–118
MACRO 14–120
MCR 14–125
MERGE 14–127
MESSAGE 14–129

MOUNT	14–131
ON	14–136
PRINT	14–139
QUEUE	14–141
REMOVE	14–147
RENAME	14–149
RUN	14–151
SET	14–159
SHOW	14–173
SORT	14–179
STOP	14–182
SUBMIT	14–184
TRUNCATE	14–186
TYPE	14–188
UNFIX	14–190
UNLOCK	14–192
VERIFY	14–194

INDEX

FIGURES

1–1	Real-Time System	1–2
1–2	Multiuser System	1–3
1–3	Timesharing System	1–4
2–1	LA30/VT05 Layout	2–2
2–2	LA36/VT50 Layout	2–2
2–3	VT52 Layout	2–3
2–4	VT100 Layout	2–3

TABLES

2–1	Keyboard Functions	2–4
2–2	Control Character Functions	2–4
4–1	PDS Command Privilege Classes	4–6
4–2	PDS Command Privileges	4–7
6–1	User Categories	6–2
6–2	IAS Device Mnemonics	6–4
6–3	Standard IAS File Types	6–5
6–4	File Specification Defaults	6–7
6–5	Summary of File Handling Commands	6–20
14–1	COBOL Switches	14–19
14–2	CORAL 66 Switches	14–36
14–3	Duplicate and Update Combinations	14–41
14–4	Use of CREATE Qualifiers with Different File Organizations	14–43

14–5	Response Choices for the /CONFIRM Qualifier	14–51
14–6	FORTRAN-IV Switches	14–76
14–7	FORTRAN-IV Plus Switches	14–78
14–8	DOS and RT11 Initialization Qualifiers	14–86
14–9	RT11 Qualifier	14–87
14–10	RT11 Qualifier	14–87
14–11	Maximum Files Per Device	14–88
14–12	Command Qualifiers and Defaults	14–111
14–13	Task Builder Options	14–115
14–14	Values for MACRO Switches /DS and /EN	14–122
14–15	Values for MACRO Switches /LI and /NL	14–122
14–16	MOUNT Command Qualifiers	14–134
14–17	Required Privileges for the SHOW Commands	14–173

Preface

Manual Objectives and Reader Assumptions

The *IAS PDS User's Guide* is intended for users of the program development system (PDS). The manual is organized as an introduction to PDS and assumes no prior knowledge of the IAS system. Chapter 14 contains a description of each nonprivileged PDS command and assumes a knowledge of the information covered in the preceding chapters.

Document Structure

This book is organized into the following chapters:

1 Introduction to PDS

2 Keyboard Operation

3 A Sample Interactive Session

4 Issuing PDS Commands

5 Batch Processing

6 File Handling

7 IAS Editors

8 Introduction to Program Control

9 BASIC-11

10 COBOL

11 FORTRAN

12 MACRO-11

13 CORAL 66

14 PDS Command Descriptions

Associated Documents

The *IAS Master Index and Documentation Directory* lists and summarizes all the associated IAS documents and intended readership.

1 Introduction to IAS

1.1 Interactive Applications System

The interactive applications system (IAS) is a multifunction operating system designed for the larger PDP-11s. IAS supports the following modes of operation:

1 Interactive

2 Batch

3 Real-time

IAS permits mixed mode processing; that is, it allows the concurrent processing of real-time, batch, and interactive tasks.

1.1.1 Interactive Processing

Interactive applications involve user and system interaction. You communicate with IAS by means of an interactive terminal. At a terminal you run a task, for example, a compiler, or a utility program. Under IAS, interactive tasks can:

1 Be under control of the IAS scheduler - see the *IAS System Management Guide.*

2 Be swapped, that is, brought in and out of memory; see the *IAS System Management Guide.*

3 Communicate with other tasks.

4 Use common utility programs.

1.1.2 Batch Processing

Batch applications are applications that require no user/system interaction after the batch job has been started. A batch job, for example, can involve compiling (assembling), linking, and running a program. Under IAS, batch jobs:

1 Have the same facilities as interactive tasks.

2 Are scheduled with, but at a level below interactive tasks.

3 Have no external communication except with Shareable Global Areas (SGAs), see the *IAS Executive Facilities Reference Manual.*

See Chapter 5 for a description of batch processing.

1.1.3 Real-Time Processing

Real-time applications require a response to actual events. The response depends on the specific real-time application. For example, the application can be used to monitor a temperature-critical chemical process in a refinery. On meeting the critical temperature, the response might be to sound a warning, adjust a valve, or relay information to a control console.

Real-time processing often involves multiprogramming where tasks can interact with each other. For example, a real-time task might activate or provide data for another real-time task. Under IAS, real-time tasks have the following characteristics:

1 Software priority between 1 (low) and 250 (high).

2 Normally checkpointable (that is, can be brought in and out of memory). See the *IAS System Management Guide* for a description of checkpointing.

1.2 Different Types of IAS System

IAS systems are available in three different types:

1 Real-time

2 Multiuser

3 Timesharing

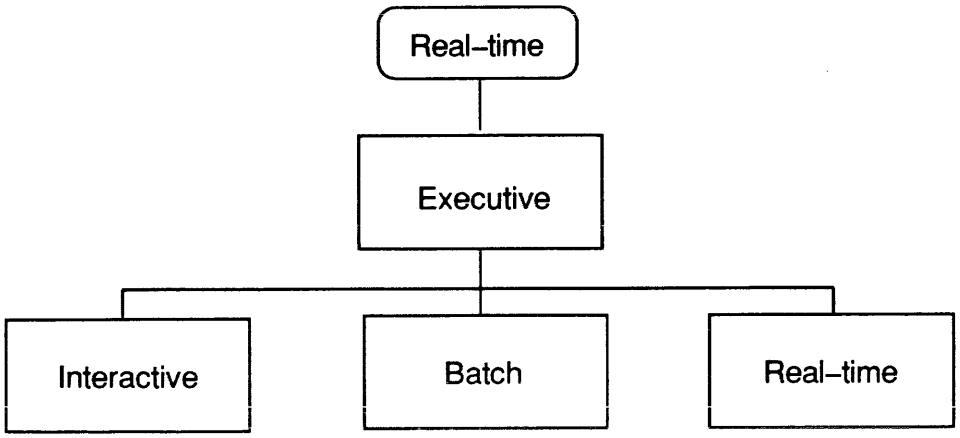
The system manager decides whether IAS is to be a real-time, multiuser, or timesharing system at system generation time. See the *IAS Installation and System Generation Guide* for a description of the system generation procedure.

See Sections 1.2.1 and 1.2.3 for descriptions of the different types of system.

1.2.1 Real-Time System

The real-time system is the simplest form of IAS system. The real-time system contains neither an IAS scheduler nor the timesharing control primitives (TCP). Consequently, this type of system is suitable only for real-time and single-user applications. Figure 1–1 illustrates a real-time system.

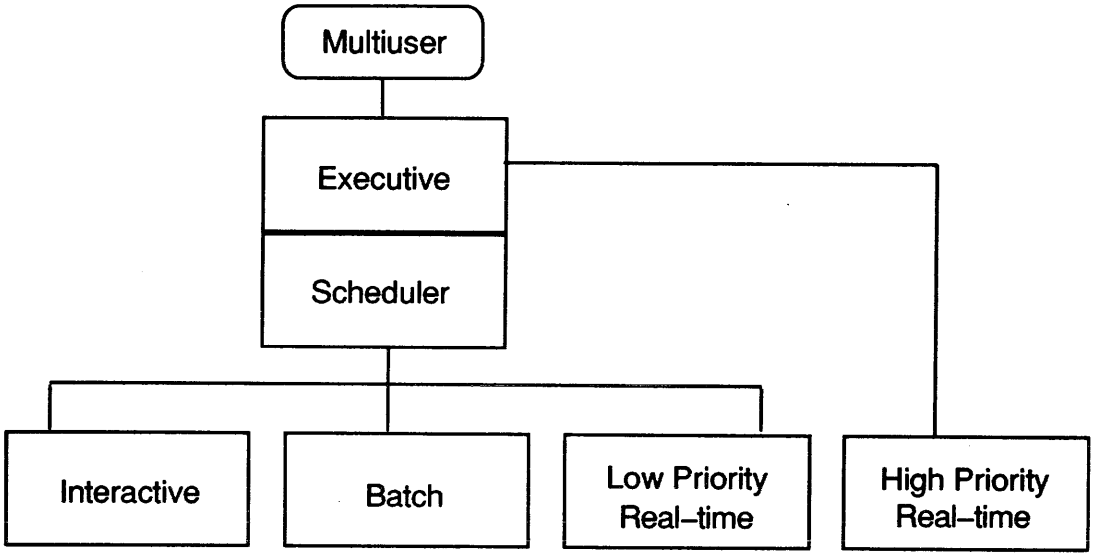
Figure 1–1 Real-Time System



1.2.2 Multiuser System

The multiuser system includes the IAS scheduler but not TCP. This type of system is suitable for multiuser development, but because it does not have the access, regulation, control, and protection facilities of TCP it is only suitable for use in a "safe" environment. Figure 1–2 illustrates a multiuser system.

Figure 1–2 Multiuser System



1.2.3 Timesharing System

The timesharing system has both an IAS scheduler and TCP. It is, therefore, intended for installations running interactive and batch processing, together with a low level of real-time processing. TCP provides the protection and privilege control facilities typically required in a timesharing environment. Figure 1–3 illustrates a timesharing system.

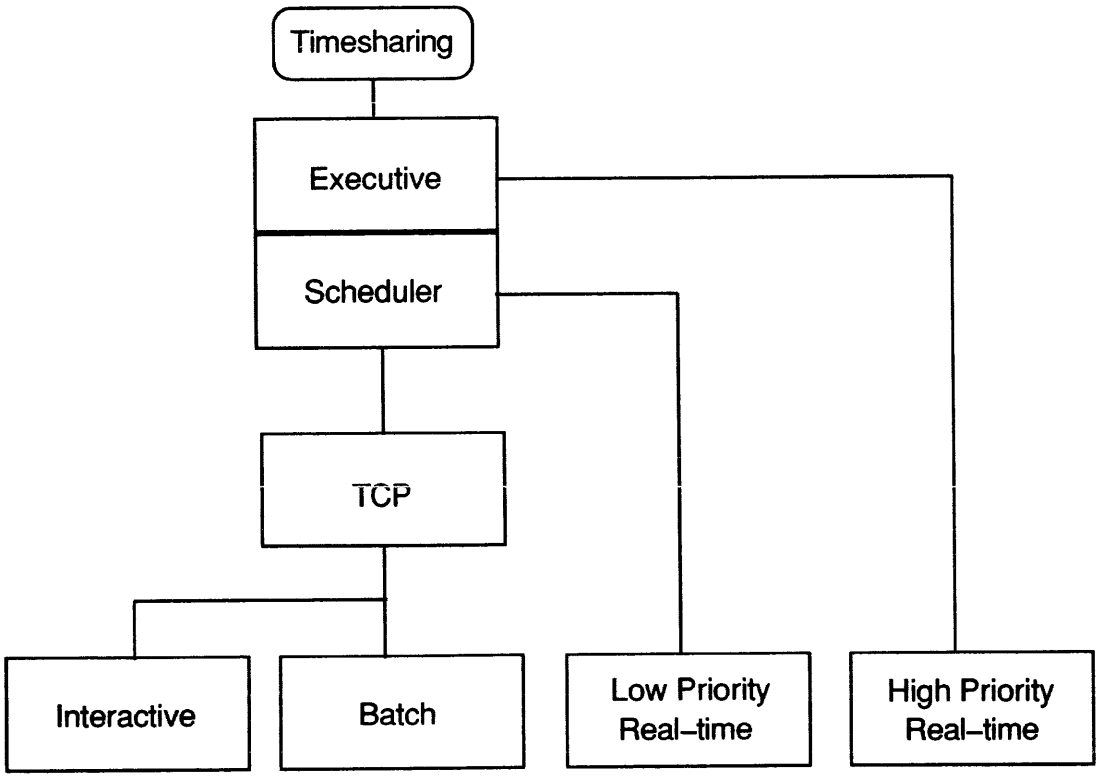
1.3 The User Interface

Under IAS, the user interface is one of the following:

1 Program development system (PDS).

2 Monitor console routine (MCR). See the *IAS MCR User's Guide* for a full description of this interface.

Figure 1–3 Timesharing System



3 A user-written command language interpreter (CLI). See the *IAS Guide to Writing a Command Language Interpreter*.

1.3.1 The Program Development System (PDS)

PDS is the CLI normally used in IAS. The PDS user communicates with IAS by issuing PDS commands from an interactive terminal (see Chapters 2, 4 and 14). PDS commands enable the user to enter and leave IAS, create, compile, link and run programs and obtain information about system status (see Chapter 3 for a sample interactive session). In addition, PDS supports an MCR mode that simulates the MCR interface, while using a terminal with PDS as the CLI. See the MCR command in Chapter 14, and see the *IAS MCR User's Guide* for descriptions of the MCR commands.

PDS includes system manager commands issued at the console terminal. The console terminal prompts SCI> (system control interface) instead of PDS>, but any user can log in at the console provided the system manager has assigned that user PR.SCI privilege. For convenience, the term SCI is used throughout the IAS manual set to describe the PDS system manager commands and interface. See Chapter 4 for a description of command privilege, and see the *IAS System Management Guide* for descriptions of the SCI commands.

1.4 System Access Security

PDS ensures system security by requiring that you enter your user name and password before you can access the system. The system checks your user name to make sure that it is authorized and your associated password is checked for validity. If both your user name and password are acceptable, you can access IAS and issue PDS commands. The system manager gives you your user name and password.

1.5 Programming Languages

IAS supports the programming languages FORTRAN-77 and MACRO-11. The MACRO-11 assembler and FORTRAN compiler are provided with IAS.

Because they produce intermediate code run by an interpreter, BASIC programs can be executed immediately after translation. Source language compilers produce machine-language code and, therefore, require the additional step of linking.

1.5.1 MACRO-11

The programmer who wants to work closely with the PDP-11 hardware and IAS can use the MACRO-11 assembler. In addition to enabling the user to invoke machine-language instructions, MACRO-11 enables you to write instructions in machine-language and to define macros to save repetitive coding sequences.

2 Keyboard Operation

This chapter describes the layout and operation of an interactive terminal. See Chapters 3 and Chapter 4 for information about logging into IAS and using PDS.

2.1 The Interactive Terminal

You type data directly into the system from an interactive terminal (for example, a DECwriter or a visual display unit (VDU)). The keyboard layout of an interactive terminal is very similar to the layout of an ordinary typewriter. The number and letter keys are the same as a typewriter; however, the position of punctuation, special character, and function keys can differ from one type of terminal to another (see Figures 2–1, 2–2, 2–3, and 2–4).

2.1.1 Typing Input

You enter data one line at a time and end each line with a carriage return (RET) or ALTmode (ESC). Carriage return is shown on the keyboards as CR or RETURN. ALTmode is shown on the keyboards as ALT or ESC. Your terminal echoes your text for verification and editing. An additional feature of IAS is typeahead mode, which allows you to enter new data while the system is processing previous input. After the previous input is processed, the system accepts the input that you typed ahead. Input in typeahead mode is not echoed on the screen until the system accepts it. For further information, see the *IAS Device Handlers Reference Manual.*

2.1.2 Keyboard Functions

Use function keys to format and edit a line or to access the upper of two characters that appear on a key. Further keyboard functions become available when you press CTRL and a letter key simultaneously. (See Tables 2.1.2 and 2.1.3 for a detailed description of these functions). Typing a RET (CR or RETURN) causes the system to store the current line or to carry out an action (for example, prompt for input).

Figure 2–1 LA30/VT05 Layout

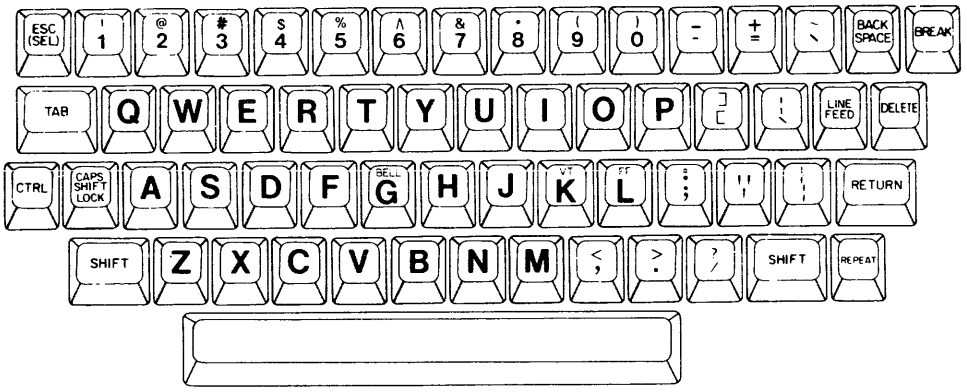


Figure 2–2 LA36/VT50 Layout

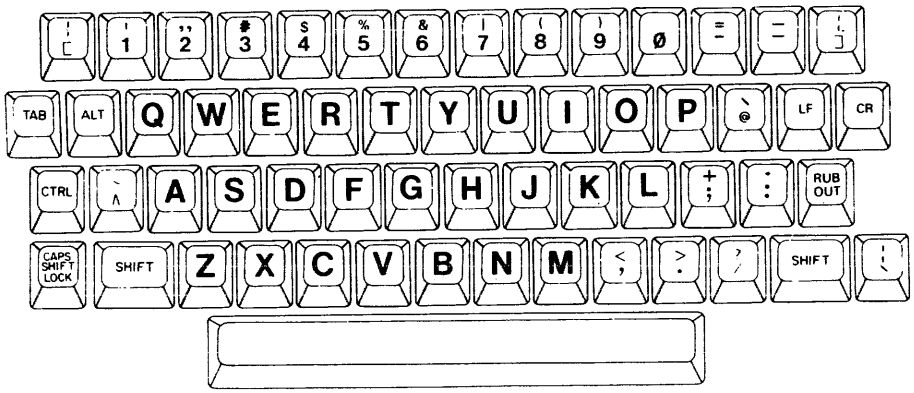


Figure 2–3 VT52 Layout

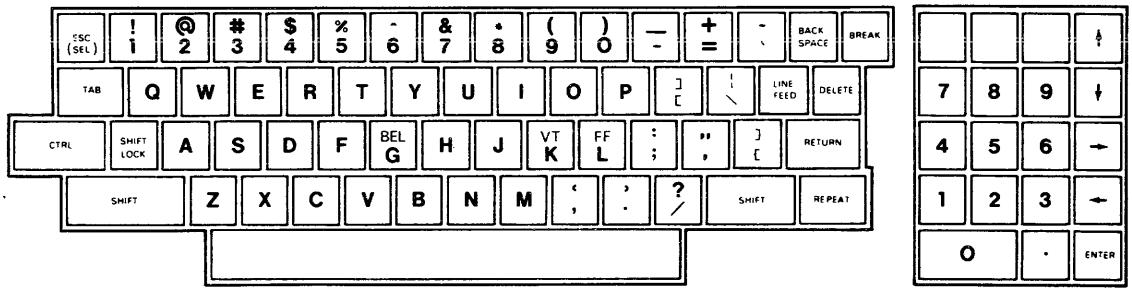


Figure 2–4 VT100 Layout

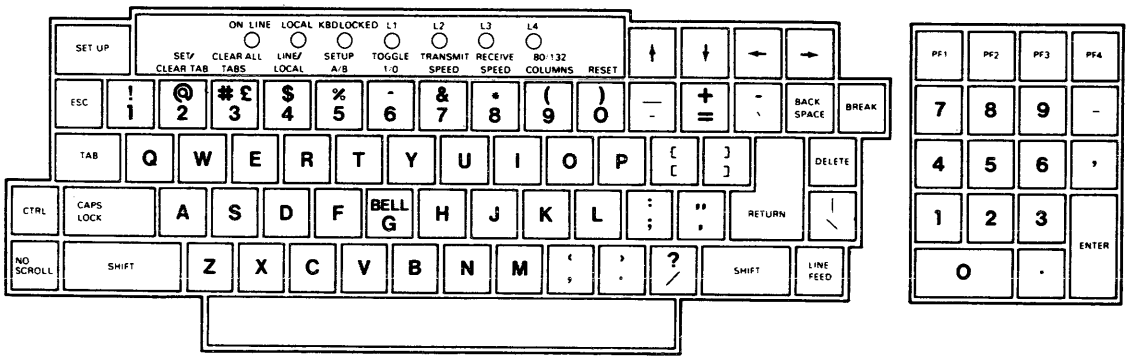


Table 2–1 Keyboard Functions

Key	Description
CR or RETURN	Transmits the current line to the computer and performs a carriage return and line feed. RETURN also causes prompting for mandatory input when you type it after a PDS command string.
CTRL	First part of a number of special two-key combinations called control functions (for example, CTRL/Z). See Section 2.1.3 for a full description.
DELETE or RUBOUT	Erases the last character typed. You can use this key repeatedly to erase a number of characters. On a VDU, the current printing position moves to the left and erases the deleted character. On other terminals, the string of deleted characters is echoed between an initial backslash (\) and a final backslash (\).
ESC or ALT	Similar in operation to RET. When you press RET after a PDS command string, the system prompts for input (either mandatory or optional). In MCR mode, any MCR command you terminate with ESC causes the suppression of the PDS>> prompt until you type CTRL/C. The character can echo as $ on some installations.
LINE FEED or LF	Has no control effect under IAS.
SHIFT	Selects the uppermost of two characters appearing on a key or selects uppercase on an upper- and lowercase terminal. You press the key while you hold SHIFT down. If you use SHIFT with keys that have only one character, it has no effect.
SHIFT LOCK	Alternately locks and unlocks SHIFT.
CAPS LOCK	Alternately locks and unlocks uppercase letters. While CAPS LOCK is in effect all letters are upper case, whether or not you press SHIFT. Other keys are not affected.
SPACE BAR	Advances the current typing position one space at a time.
TAB	Causes the current typing position to move to the next tab stop on the line (normally 8 spaces).

2.1.3 Control Character Functions

Typing a character key while pressing CTRL invokes one of the functions listed in Table 2–2. The combination of CTRL and another character key is called a control character; for example, CTRL/Z.

Table 2–2 lists the control characters supported under IAS and their associated functions.

Table 2–2 Control Character Functions

Control Character	Function
CTRL/C	Invokes PDS if you issue it before logging in. Returns control to PDS if you issue it while a task is running. Cancels a command if you issue it between the PDS prompt and RET.
CTRL/B	On a terminal with a low-speed paper tape reader, CTRL/B signals the computer to start reading the tape.
CTRL/I	Causes the current typing position to move to the next tab stop on the line (performs the same action as TAB).
CTRL/K	Advances the current line to the next vertical tab stop (equivalent to a linefeed).
CTRL/L	Advances continuous stationery to the next top of form (equivalent to a formfeed).

Table 2–2 (Cont.) Control Character Functions

Control Character	Function
CTRL/O	Interrupts and suppresses output to the terminal. Successively pressing CTRL/O causes the suppression and resumption of output. For example, if you want a directory listing on your terminal and the first few lines give you the information you need, CTRL/O suppresses the rest of the directory.
CTRL/Q CTRL/S	You use these keys in conjunction with each other. CTRL/S suspends output to the terminal. CTRL/Q resumes output to the terminal. You can stop and start output without losing any characters.
CTRL/R	Retypes the current line, removing any deleted characters.
CTRL/T	On a terminal with a low-speed paper tape reader, CTRL/T stops reading from the tape. CTRL/T can be present on the tape, or you can switch the reader off and type CTRL/T on the keyboard.
CTRL/U	Deletes the current input line. The prompt, if any, is then repeated.
CTRL/V	Flushes all characters typed ahead of a read. If a read is in progress CTRL/V has no effect. For typeahead modes, see the *IAS Device Handlers Reference Manual.*
CTRL/Z	Terminates a file input from a terminal; that is, signals "end of file".

2.2 Correcting Input Errors

Before you terminate a line, you can correct typing errors or change the line completely by using DELETE or CTRL/U. However, once the line has been terminated and transmitted to the computer, you cannot correct any errors in that line. If the input is held in a file, you can correct it by means of an editing program.

2.2.1 Cancelling a PDS Command

Typing CTRL/C or CTRL/U cancels a PDS command that has not yet been terminated.

2.2.2 Deleting Characters

On a VDU, pressing DELETE deletes the most recent character displayed. DELETE has no effect when the current line is empty. Each deleted character is removed from the screen, and the cursor returns to the position it occupied before you typed the character. For example, to change ACCDE to ABCDE press DELETE four times to erase CCDE, then type the sequence BCDE. The result displays as follows:

```
ABCDE
```

On a hard copy terminal, each deleted character is echoed. The deleted characters are delimited by backslashes (\ ). Using the previous example, the deleted string CCDE echoes as:

```
ACCDE\EDCC
```

After you type the sequence BCDE, the string appears as follows:

```
ACCDE\EDCC\BCDE
```

NOTE: Note that the final backslash is added when a character other than DELETE is typed.

For both the VDU and hard copy terminal, ABCDE is the string accepted and sent to the computer when the line is terminated.

2.2.3 Deleting a Line

Pressing CTRL/U has the following effect:

1 Deletes all the characters on a line.

2 Echoes with ^U.

3 Performs a carriage return/line feed.

You can then enter the correct text. For example, if you type ACCDEFGHI, but meant to type B for the first C, pressing RUBOUT 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 procedure would appear as follows:

```
ACCDEFGHI ^U
ABCDEFGHI
```

After using DELETE to correct a line, but before terminating the line, you can ensure that the final result is correct. To display the line as it is to be sent to the computer, simply press CTRL/R; you can make further corrections at this point if necessary. With CTRL/R and CTRL/U, the prompt, if any, is repeated.

2.2.4 Using Upper- and Lowercase

PDS can accommodate both upper- and lowercase, provided the terminal is upper- and lowercase. The lowercase characters echo as uppercase (depending on the terminal); see the *IAS Device Handlers Reference Manual.*

3 A Sample Interactive Session

3.1 Introduction

This chapter introduces PDS by demonstrating its use in a session at an interactive hardcopy terminal. Section 3.2 records the session, which is then described line by line in the remaining sections of this chapter. The development session covers the following procedures:

1 Logging in

2 Creating a file

3 Correcting typing errors

4 Cancelling a line

5 Closing a file

6 Displaying the contents of a file

7 Translating a source to an object module

8 Linking a program

9 Running a program

10 Obtaining directory information

11 Renaming files

12 Logging out

NOTE: The line numbers at the left margin in Section 3.2 are for reference and are not part of the actual session.

3.2 Sample Interactive Session

This section records a short program development session, where a small program to add three numbers is written in FORTRAN.

```
01  Ctrl/C
02  Username? CAROL
03  Password?
04       IAS PROGRAM DEVELOPMENT SYSTEM  VERSION 3.4
                 17:09:08       15-MAY-90
05  USER CAROL UIC [200,22] TT05: 17:09:21 15-MAY-90
06  PDS> CREATE ADD.FTN
    READY FOR INPUT
07       TYPE 1
08  1    FORMAT(' ENTER TWO NUMBERS')
```

```
09      APPE\EPP^R
10      ACCEPT 2,K,L
11  2   FORMAT
    (22\2\I5)
12      PRINT^U
13      TYPE 3,K+L
14  3   FORMAT(' THE SUM IS ',I5)
15      STOP
16      END
17  ^Z
18  PDS> TYPE ADD.FTN
19      TYPE 1
20  1   FORMAT (' ENTER TWO NUMBERS')
21      ACCEPT 2,K,L
22  2   FORMAT (2I5)
23      TYPE 3,K+L
24  3   FORMAT (' THE SUM IS ',I5)
25      STOP
26      END
27  PDS> FORTRAN ADD
28  17:17:41 SIZE: 10K CPU: 0.10
29  PDS> LINK ADD
30  17:18:38 SIZE: 11K CPU: 12.06 STATUS: SUCCESS
31  PDS> RUN ADD
32  17:30:51
33  ENTER TWO NUMBERS
34  12, 78
35  THE SUM IS 90
36  JOB160 -- STOP
37  17:31:14 SIZE: 7K CPU: 0.02
38  PDS> DIRECTORY
39  DIRECTORY DB0:[200,22]
40  15-MAY-90 17:36
41  ADD.OBJ;1       2.      15-MAY-90 17:17
42  ADD.FTN;1       1.      15-MAY-90 17:17
43  ADD.TSK;1       32.   C  15-MAY-90 17:18
44      TOTAL OF 35./35. BLOCKS IN 3. FILES
45  PDS> RENAME ADD.*;* ADDTWO.*;*
46  PDS> DIRECTORY/BRIEF^U
47  PDS> DIRECTORY/BRIEF
```

```
48  DIRECTORY DB0:[200,22]
49  ADDTWO.OBJ;1
50  ADDTWO.FTN;1
51  ADDTWO.TSK;1
52  PDS> LOGOUT
53  USER CAROL UIC [200,22] TT05: 17:45:01  15-MAY-90
54  CONNECT TIME 14 M  SYSTEM UTILIZATION 12 MCTS
55  BYE
```

3.3 Logging In

The following sections describe the steps shown in the sample development session in Section 3.2.

One method of giving instructions to a computer is to type commands to a "command-line interface". PDS is the command-line interface for IAS. PDS interprets the commands you give to IAS. When PDS is ready, you can issue commands. When you complete the following login procedure, PDS will be ready to receive commands.

3.3.1 Control C

To begin the login procedure, hold down Ctrl and press C. This key sequence is called "CONTROL C", and is written Ctrl/C or ^C.

3.3.2 The Username

In response to Ctrl/C, the Username prompt displays as follows:

```
Username?
```

The username is a unique 1 to 12 character alphanumeric string that uniquely identifies the user. Enter your username at this prompt.

The system manager assigns your username, which is then registered with IAS. If you do not have a username, or have forgotten it, consult the system manager.

3.3.3 The Password

After you type in your username, the password prompt displays as follows:

```
Password?
```

Enter your password to this prompt. The password is an alphanumeric string of from 1 to 6 characters that is associated with your username. The characters you enter do not appear on the screen.

The password is an additional security measure that prevents unauthorized access to the system. You can change your password by using the SET PASSWORD command. See Chapter 14 for more information.

If the password you provide is incorrect, PDS displays the Password? prompt again. You have three chances to type the password correctly before PDS exits and displays "BYE". If, after three attempts, the correct password has not been provided, a warning message indicating an unsuccessful login attempt displays at the system console.

3.3.4 The PDS Prompt

If you supply your password correctly, the system completes the login procedure and determines your user attributes (UIC, privileges, and so on). (See Chapter 6 information on UICs, and Chapter 4 for information on command privileges.) PDS then displays the login banner, followed by your username, UIC, terminal number, and the time and date. For example:

```
IAS PROGRAM DEVELOPMENT SYSTEM  VERSION 3.4
        17:09:08      15-MAY-90
USER CAROL UIC [200,22] TT05: 17:09:21 15-MAY-90
```

The system then displays the PDS prompt (PDS>) to indicate that it is ready to accept PDS commands. Any system notices to be displayed when users log in display before the PDS prompt.

If nothing is typed for several minutes after logging in, and no program is running, PDS becomes inactive or "times out", and your account will is logged off (the exact number of minutes depends on the installation). When timeout occurs, the system displays the following message:

```
TIMEOUT
BYE
```

You must then type CTRL/C to reactivate PDS.

3.4 The CREATE Command

The CREATE command is used to create a file. In the session illustrated in Section 3.2, the CREATE command is used to create a file called ADD.FTN (line 6). ADD is the file name and .FTN is the file type; the file type describes the contents of the file. In this case, it indicates that the file contains a FORTRAN source program. See Table 6–3 for a list of IAS file types.

After terminating the CREATE command by pressing RETURN, you can enter the source program. The first typing position on each line is equivalent to position 1 on a coding sheet or punched card. Use the function keys to format the lines. (See Chapter 2 for more information.) For example, you can use TAB to skip eight spaces to position the text "TYPE 1" in line 7. RETURN terminates each line and moves the typing position 1 of the next line.

3.4.1 Correcting Input Errors

On line 9, an error is made is corrected by using DELETE. You must press the key three times to delete E, P, and P again. The characters deleted are echoed on the terminal as follows:

```
APPE\EPP
```

Each time you press the key, the system deletes the rightmost character. Video display terminals erase each deleted character from the screen and move the printing position to the left. Hardcopy terminals use a backslash character (\ ) to indicate deleted characters.

In this example, the user presses CTRL/R to display the corrected text on a clean line (line 10). For example:

```
APPE\EPP^R
A
```

The user then completes the line correctly and terminates it with RETURN:

```
ACCEPT 2,K,L
```

If, instead of CTRL/R, the user had typed the amended letters CCEPT on the same line, the system would first have closed the string of deleted characters by a second backslash. For example:

```
APPE\EPP\CCEPT 2,K,L
```

On line 11, DELETE is used once more to delete the second 2.

```
2  FORMAT(22\2\I5)
```

3.4.2 Canceling a Line

By mistake the user types PRINT on the next line, then presses CTRL/U to cancel the line and start again on line 13. CTRL/U deletes a line that has not been terminated by RETURN and advances the typing position to the beginning of the next line. The user can then enter the text that was originally intended.

```
PRINT^U
TYPE 3,K+L
```

3.4.3 Closing the New File

The last statement of the source program is END (line 16). After entering the last statement, the user types Ctrl/Z to indicate to the system that the file ADD.FTN is complete. The system displays ^Z, closes the file, then prompts PDS> on the next line.

3.5 The TYPE Command

At the PDS> prompt (line 18) the user issues the TYPE command to display the file ADD.FTN as it appears after corrections. The system responds by printing the contents of the file (lines 19 through 26).

3.6 The FORTRAN Command

After checking that the source program is correct, the user decides to run the program. However, the program must first be translated into instructions that the computer can understand.

In IAS, the FORTRAN command is used to translate a FORTRAN source program. On line 27, the user types the following:

```
PDS> FORTRAN ADD
```

In this case the file is specified as ADD rather than ADD.FTN. The FORTRAN command assumes the file type to be FTN if it is not supplied.

NOTE: The user should ensure that the relevant language compiler is installed before attempting to use that compiler.

After translating the program, the system prints the following text on line 28:

```
17:17:41  SIZE: 10K  CPU: 0.10
```

The figures "17:17:41" refer to the time the system finished translating the program. Line 28 also shows the size of the program just completed, which in this example is the FORTRAN compiler. "0.10" indicates that the translation required one tenth of a second of CPU time.

The system automatically places the translated FORTRAN program, called an "object module", in a file named ADD.OBJ. The file type .OBJ indicates that the file contains an object module.

3.7 The LINK Command

The purpose of the LINK command (line 29) in this session is to couple the object module contained in ADD.OBJ with the FORTRAN subroutines that it needs. FORTRAN programs use a set of subroutines to perform the required functions. For example, the FORTRAN statements TYPE and ACCEPT require the subroutines for input/output functions. The system maintains these subroutines in object module form so that they do not have to be translated each time they are used. The format of a LINK command is:

```
PDS> LINK ADD
```

The file type is assumed to be .OBJ. If there is no file called ADD.OBJ, the system returns an error message. This might occur, for instance, if a user tries to link an untranslated FORTRAN program. (See the note at the end of this section.)

Line 30 displays statistics about the completed execution of the LINK command.

The linked, executable program (the translated program linked with the required subroutines) is then placed in a file called ADD.TSK. The file type TSK stands for "task", which is IAS terminology for an executable program.

NOTE: FORTRAN programs always require a FORTRAN library. Usually the FORTRAN library is installed as part of the system library (SYSLIB) and is, therefore, automatically linked. However, some IAS installations might not include the FORTRAN library in SYSLIB. In this case, your program will not link correctly and an error message displays. If this happens, use the following procedure:

1. **Contact your system operator.**
2. **Ask which FORTRAN compiler (FORTRAN IV or FORTRAN IV-PLUS) is the system default.**
3. **Ask what the name of the corresponding FORTRAN library file in the system UFD [1,1] is.**
4. **Enter the following LINK command, substituting the correct library file name:**

```
PDS> LINK
FILE? ADD,[1,1]library name/LIB
```

3.8 The RUN Command

After the source program is prepared for execution, you issue the RUN command to activate the task as follows:

```
PDS> RUN ADD
```

Again, the file type can be omitted; it is assumed to be .TSK.

Line 32 shows what time the program began to run.

The FORTRAN program ADD is interactive. ADD requests the user to enter two numbers, then adds them together and displays the result (lines 33 to 35):

```
ENTER TWO NUMBERS
12, 78
THE SUM is 90
```

When you write interactive programs, you must remember to prompt for input. If no prompts appear, you do not know what data to enter, or at what point to enter the data. This program uses the statements on lines 19 and 20 to display the prompt:

```
ENTER TWO NUMBERS
```

The numbers 12 and 78 are entered, the input is terminated by RETURN. The program then executes the program statements on lines 23 and 24 by adding the numbers and declaring the sum to be 90. The STOP statement (line 25) then causes the program to stop and the system to display the following line:

```
JOB160 -- STOP
```

The name "JOBnnn" is a name given to the job by the system each time it is run.

The information displayed on the next line is similar to that on line 28 and 30 described in previous sections.

3.9 The DIRECTORY Command

In the session so far, the user has created one file and the system has created two more, as follows:

- ADD.FTN
- ADD.OBJ
- ADD.TSK

NOTE: The system never deletes a file automatically; all three files will still exist. Only the system manager, the owner of the file, or users authorized by the file owner can delete a file.

The DIRECTORY command (line 38) causes the system to display a list of files currently held in the Users File Directory (UFD). Line 39 identifies the UFD as [200,22]:

```
DIRECTORY DB0:[200,22]
```

The 200 identifies the user group and the 22 identifies the user number within the group. The text "DB0:" indicates that the directory resides on a volume mounted on a disk drive named DB0:.

Line 40 states the date and time that the library was requested.

The next three lines list the directory information:

```
ADD.OBJ;1       2.       15-MAY-90 17:17
ADD.FTN;1       1.       15-MAY-90 17:17
ADD.TSK;1       32.  C   15-MAY-90 17:18
```

The number ";1" that appears after the file type is the file's version number and indicates that each file listed is the first version of the file. If the command FORTRAN ADD was issued again, the FORTRAN translator would produce a second object file called ADD.OBJ;2. The old versions can be deleted or retained as security against the loss of later versions.

The value in the second column indicates the number of 512 byte blocks occupied by each file on the disk. The next columns show the data and time each file was created. The "C" that appears on the third line between the number of blocks and the date indicates that the blocks within ADD.TSK;1 are "contiguous" (that is, they can be read in a single disk access).

3.10 The RENAME command

The RENAME command enables you to change the name of a file without changing its contents or location. The RENAME command is issued to rename all three files named ADD at the same time:

```
PDS> RENAME ADD.*;* ADDTWO.*;*
```

The asterisks (*) that appear in the above line enable all three files to be specified at once. An asterisk or "wildcard", is equivalent to "all". ADD.*;* includes all files that have ADD as a file name, disregarding the file type and version number. In this case, ADD.*;* refers to the files ADD.FTN;1, ADD.OBJ;1, and ADD.TSK;1. Because all the files have the same version number but different file, these files could also be referred to as follows:

- ADD.*;1

The command issued on line 45 changes the filename from ADD to ADDTWO. The wildcards in the text "ADDTWO.*;*" indicate that the renamed files retain their original file types and versions. The files are now called:

- ADDTWO.FTN;1
- ADDTWO.OBJ;1
- ADDTWO.TSK;1

3.11 The DIRECTORY/BRIEF Command

When the user reissues the DIRECTORY command (lines 46 and 47) the system lists the files with their new filenames.

NOTE: A CTRL/U was issued to cancel line 46 because of a typing error.

The DIRECTORY command includes the text "/BRIEF", a *qualifier* that modifies the action of the command. The /BRIEF qualifier causes the system to list only the names of the files and to omit information about blocks and time of creation.

Most commands have one or more qualifiers. A slash (/) always precedes the qualifier name. When more than one qualifier is specified, slashes separate one qualifier from the next.

3.12 The LOGOUT Command

To end the interactive session, you must issue the LOGOUT command (line 52). The system then displays user and accounting information on the next two lines and the text "BYE" on the third line. The terminal is now inactive.

4 Issuing PDS Commands

4.1 Introduction

This chapter covers the following topics to describe the use of PDS commands:

1 Commands and parameters

2 Abbreviating input

3 Qualifiers

4 Error conditions

5 Command and task privileges

4.2 PDS Commands

You communicate with the system by issuing commands to PDS. You issue commands at an interactive terminal or by submitting a file of commands to a batch queue. A command comprises a command name that describes the action the system is to take (for example, COPY or LOGIN), and usually one or more parameters. Parameters either describe the items on which the command is to operate or further define the function of the command. For example:

```
PDS> SHOW STATUS
PDS> SHOW DEVICES
```

You can enter commands at an interactive terminal only when the system is prompting PDS>. Some PDS commands (for example, EDIT and BASIC) invoke a program that accepts its own set of commands. These commands are valid only while that program is running. Furthermore, PDS commands are not valid while that program is running; to issue PDS commands, you must first return control to PDS. The descriptions of EDIT and BASIC in Chapter 14 explain how execution is terminated and control is returned to PDS.

4.2.1 Command Strings

Command strings can specify the command name followed by the parameters or can specify only the command name (in which case you supply the parameters in response to prompts). Command strings issued by batch users must specify the command names and parameters on a single or continued line.

In both batch and interactive mode, when two or more parameters are on one line you must separate them by spaces, a comma, and/or tabs.

If a command string is longer than one line, a hyphen (-) as the last character on the line causes the command to continue on the next line.

An exclamation mark (!) after the last character of a command line indicates the start of a comment. The comment text appears after the exclamation mark.

4.2.2 Abbreviated Input

You need only enter enough of a PDS command to distinguish it from all other PDS commands. For example, you can abbreviate the DEALLOCATE command to DEAL. However, DEA is not acceptable because it does not distinguish between DEALLOCATE and DEASSIGN.

4.2.3 Underline Convention

To increase legibility, some qualifiers have an underline character where two or more words have been joined together. For example:

```
PDS> MOUNT/FILE_PROTECTION:(code)
```

When such qualifiers are abbreviated, the underline is treated in the same way as alphabetic characters. The following examples are acceptable, since they identify this qualifier uniquely among the MOUNT qualifiers. The underline convention does not apply to the prefix NO.

```
PDS> MOUNT/FILE_PROT:(code)
PDS> MOUNT/FI:(code)
```

4.3 Command Parameters

Most PDS commands require parameters. For example, the COPY command requires an input file specification and an output file specification, which you can supply as follows.

In Interactive Mode:

```
PDS> COPY RISE.MAC WORK.MAC

PDS> COPY RISE.MAC, WORK.MAC

PDS> COPY
FROM? RISE.MAC WORK.MAC

PDS> COPY RISE.MAC
TO? WORK.MAC

PDS> COPY
FROM? RISE.MAC
TO? WORK.MAC
```

In Batch Mode:

```
$COPY RISE.MAC WORK.MAC

$COPY RISE.MAC,WORK.MAC
```

Batch mode is described in Chapter 5.

4.3.1 Parameter Lists

You can replace some parameters by enclosing a list of parameters in parentheses, separating the parameters by spaces, tabs, and/or a comma. You do not need parentheses when the list replaces the last or only parameter in the command. For example:

```
PDS> APPEND (FILEA.FTN,FILEB.FTN) FILEC.FTN

$DELETE AB.CBL;1 AB.OBJ;1
```

4.3.2 Parameter Prompts

Parameter prompts request specific input to complete a command. The CREATE command demonstrates how PDS prompts for command parameters at an interactive terminal. For example:

```
PDS> CREATE
FILE?
```

Because the file specification was not supplied with the CREATE command, PDS prompts for the file specification. Parameter prompts can reduce the number of input errors made by users who are unsure of the correct command parameters.

The more experienced user may be familiar with the commands and may not need the parameter prompts. Therefore, prompts are suppressed when necessary parameters are supplied. Refering to the previous example, if a file specification had been provided, the FILE? prompt would not have displayed, and the CREATE command would have been immediately executed.

4.3.3 Optional Parameters

Interactive PDS commands prompt for both mandatory and optional parameters. To display the prompt for an optional parameter, however, you must use [ESC] rather than [RETURN] after the last mandatory parameter. For example:

```
PDS> MOUNT [RETURN]
DEVICE? DK: [RETURN]
VOLUME-ID? CHARLY [ESC]
LOGICAL NAME? AB
```

where LOGICAL NAME? is a prompt for an optional parameter of the form AB.

To suppress the prompt LOGICAL NAME?, you must press carriage return after CHARLY. For example:

```
PDS> MOUNT DK2: CHARLY [RETURN]
```

If you have invoked an optional prompt by mistake, you must type [RETURN] immediately after the prompt. For example:

```
PDS MOUNT DK2: CHARLY [RETURN]
LOGICAL NAME? [RETURN]
```

Batch users can either omit the optional parameter from the command string (if it is the last parameter), or replace the optional parameter with two commas if there are further parameters to be specified. For example:

```
$SHOW TASK/ACT,,ALL
```

4.4 Command and Parameter Qualifiers

Command qualifiers modify the function of a command. The PRINT command illustrates the use of commands and qualifiers. The main purpose of the PRINT command is to output one or more specified files on a line printer. To delete the file or files is an option that you indicate by specifying the command qualifier /DELETE. For example:

```
PDS> PRINT/DELETE FILE.LST
```

You can abbreviate each qualifier by supplying enough characters to distinguish it from any other legal qualifier for this command.

File specifications can also have qualifiers; these qualifiers describe properties the file has or is to have. For example, the /PROTECTION qualifier can modify the file specification supplied with the CREATE command, see Chapter 6. The qualifier determines the protection code applied to the newly-created file. For example:

```
$CREATE NEWFILE.DAT/PROTECTION:(SY:RWED,OW:RWED,GR:R,WO:R)
```

4.5 Invalid Commands or Syntax

PDS may not be able to execute a command for one of the following reasons:

1 There are too many tasks already active on the terminal.

2 A subsystem, for example, BASIC, is being used.

3 There has been a user or system error.

4 The command line is too long.

4.5.1 Active Tasks

If you type CTRL/C when a task is active, the task will be interrupted, and PDS will reprompt PDS>. At this point you can issue further commands which initiate another task. You can repeat this process until you reach your task limit. This limit is set by the System Manager, the default is one interruptable task. When this limit is reached and an attempt to run another interruptable task is made the system will report the following errors:

```
COMMAND NOT ALLOWED - TASK SUSPENDED
(Timesharing systems)

  or

COMMAND NOT ALLOWED - MAXIMUM NUMBER OF TASKS ALREADY ACTIVE.
(Multi-user systems)
```

You may issue any non-interruptable command after these messages, in particular ABORT to abandon, or CONTINUE to resume, the latest task activated. See Section 4.8 and Table 4–2 for a description of non-interruptable/interruptable commands.

4.5.2 Subsystems

PDS commands are not valid when you are operating within a subsystem such as BASIC or the Line Text Editor. You must first return control to PDS and then issue a PDS command.

4.5.3 Errors

When a command fails, PDS displays an error or diagnostic message that indicates where the problem lies. The following interactive session includes examples of command failures and the resultant system responses. Asterisks have been added to responses that indicate a failure.

```
CTRL/C
USER NAME? SMITJ
PASSWORD? (The terminal does not display the password.)
*USER NAME NOT AUTHORIZED

CTRL/C
USERNAME? SMITH
PASSWORD? (The terminal does not display the password.)
*PASSWORD?
*PASSWORD?
USER SMITH UIC [100,100] TT07 TIME 16:29:10 15-MAY-84

PDS> COPY
FROM? A$B
*A - ILLEGAL DEVICE
*ILLEGAL FILE - SPECIFICATION
```

The reasons for failure are as follows:

1 USER NAME NOT AUTHORIZED - The user name (SMITJ) you supplied did not grant you access to PDS because you had mistyped the last character.

2 PASSWORD? - By repeating the password prompt, the system indicated that the user SMITH had not typed the correct password.

3 A - ILLEGAL FILE-SPECIFICATION - $ is not a valid character within a file specification.

4 A - ILLEGAL DEVICE ILLEGAL FILE-SPECIFICATION - A is not a valid IAS device name.

Common errors are:

1 Mistyping characters within a command.

2 Not leaving a space where it is needed to distinguish between command components.

3 Not supplying parameters in the correct order.

4 Not specifying the correct devices.

4.5.4 Length of Command Line

PDS expands command lines to include all default devices and UFDs. When you are using certain system tasks (for example, language compilers) this can cause the command line to exceed the limit of 80 characters and the system displays the message:

```
COMMAND TOO LONG
```

To overcome this, do one of the following:

1 Reduce the length of your files names.

2 Split the command into two (or more) separate, shorter commands.

3 Enter MCR mode if you have the appropriate privilege, (see the MCR command in Chapter 14 and enter the command line using MCR. See the *IAS MCR User's Guide* for further details.

4 Install the system task as a $$$task, if you have the appropriate privilege (see Table 4–1, and enter the command line directly.

4.6 PDS Command Privilege

PDS command privilege governs the right of an individual user to issue a specific command or set of commands via PDS. These rights are assigned or withheld by the system manager when the user is authorized.

4.6.1 PDS Command Masks

Each user is allocated two PDS command masks on authorization. One mask is for interactive terminal use and the other for batch use. Each mask consists of 16 bits. A bit is set to 1 to make the corresponding command(s) available. Tables 4–1 and Table 4–2 list the bits and the associated symbolic name and command.

Table 4–1 PDS Command Privilege Classes

Bit	Symbol	Command or Class of Commands
0	PR.FIL	File Manipulation Facilities
1	PR.RUN	Task Manipulation
2	PR.BAS	BASIC
3	PR.COB	COBOL
4	PR.COR	CORAL
5	PR.FOR	FORTRAN
6	PR.LIN	LINK
7	PR.MAC	MACRO
8	PR.SCI	SCI Privilege
9	PR.SUB	SUBMIT (to Batch)
10	PR.MCR	MCR Mode
11	PR.DEV	Device Management
12	PR.DUM	DUMP
13	PR.LIB	LIBRARIAN
14	PR.SYS	System Library Tasks ($$$xxx)
15	PR.RTC	Real-time Commands

Several commands are available to all logged-in users. These are marked ANY in Table 4–2 and are independent of the privilege masks. Table 4–2 shows the privilege required for each command.

Table 4–2 PDS Command Privileges

Command	Privilege	Interruptable/ Noninterruptable[1]
ABORT/TIMESHARING	ANY	NI
ABORT/REALTIME	PR.RTC	NI
ALLOCATE	PR.DEV	NI
APPEND	PR.FIL	I
ASSIGN (timesharing)	PR.RUN	NI
ASSIGN/TASK	PR.RTC	NI
BASIC	PR.BAS	*
CANCEL	PR.RTC	NI
COBOL	PR.COB	I
COMPARE	PR.FIL	I
CONTINUE/TIMESHARING	ANY	NI
CONTINUE/MESSAGE	PR.RTC	NI
CONTINUE/REALTIME	PR.RTC	NI
COPY	PR.FIL	I
CORAL	PR.COR	I
CREATE (file)	PR.FIL	I
CREATE/DIRECTORY	PR.FIL	NI
DCL	ANY	NI
DEALLOCATE	PR.DEV	NI
DEASSIGN	PR.RUN	NI
DEASSIGN/TASK	PR.RTC	NI
DELETE	PR.FIL	*
DIRECTORY	PR.FIL	*
DISABLE	PR.RTC	NI
DISMOUNT	PR.DEV	NI
DUMP	PR.DUM	I
EDIT	PR.FIL	I
ENABLE	PR.RTC	NI
EOD	N/A	N/A
EOJ	N/A	N/A
FIX	PR.RTC	NI

[1]An asterisk (*) indicates special features; see the particular command description under Technical Notes for further information.

Table 4–2 (Cont.) PDS Command Privileges

Command	Privilege	Interruptable/ Noninterruptable[1]
FORTRAN	PR.FOR	I
GOTO	ANY	NI
HELP	ANY	NI
IDENTIFY	ANY	I
INITIALIZE	PR.DEV	NI
INSTALL	PR.RTC	NI
JOB	N/A	N/A
LIBRARIAN	PR.LIB	I
LOGOUT	N/A	N/A
MACRO	PR.MAC	I
MCR	PR.MCR	*
MERGE	PR.FIL	I
MESSAGE	ANY	NI
MOUNT	PR.DEV	I
ON	ANY	NI
PRINT	PR.FIL	I
QUEUE	PR.FIL	*
REMOVE	PR.RTC	NI
RENAME	PR.FIL	I
RUN (timesharing)	PR.RUN	I
RUN (realtime)	PR.RUN, PR.RTC	NI
SET BOOTSTRAP	PR.DEV	NI
SET DEFAULT	ANY	NI
SET END_OF_FILE	PR.FIL	I
SET [NO]QUIET	ANY	NI

[1]An asterisk (*) indicates special features; see the particular command description under Technical Notes for further information.

Table 4–2 (Cont.) PDS Command Privileges

Command	Privilege	Interruptable/ Noninterruptable[1]
SET NOREALTIME_CONTROL	PR.RTC	NI
SET PASSWORD	ANY	NI
SET PRINTING	ANY	NI
SET PRIORITY	PR.RTC	NI
SET PROTECTION	PR.FIL	I
SET TERMINAL attribute	ANY	NI
SET TERMINAL:(TTm,...,TTn)	[1,1]	NI
SET UIC	PR.RTC (multi-user only)	NI
SHOW CLI	ANY	NI
SHOW DAYTIME	ANY	NI
SHOW DEFAULT	ANY	NI
SHOW DEVICES	ANY	NI
SHOW GLOBAL_AREAS	ANY	NI
SHOW I_O_QUEUES	ANY	NI
SHOW LUNS	PR.RTC	NI
SHOW MEMORY	ANY	I
SHOW PARTITIONS	ANY	NI
SHOW SHAREABLE_GLOBAL	ANY	NI
SHOW STATUS	ANY	NI
SHOW SWITCH_REGISTERS	ANY	NI
SHOW TASKS	ANY	NI
SORT	PR.FIL	I
STOP	ANY	NI
SUBMIT	PR.SUB	I
TRUNCATE	PR.FIL	I
TYPE	PR.FIL	I
UNFIX	PR.RTC	NI
UNLOCK	PR.FIL	I
USERS	[1,1]	NI
VERIFY	PR.DEV	I

[1]An asterisk (*) indicates special features; see the particular command description under Technical Notes for further information.

4.7 PDS Timesharing Task Privilege

Privilege controls the execution of timesharing tasks. A task can be executive or directive privileged. See the *IAS Executive Facilities Reference Manual* and the *IAS System Directives Reference Manual*.

PDS users who wish to run tasks in timesharing mode need the requisite bits to be set in their PDS Timesharing Task Privilege Mask (TP1). This mask is set up by the system manager when the user is authorized (see the *IAS System Management Guide*).

If a PDS user tries to initiate a task that is executive privileged, and is not authorized to run such tasks, the command is rejected before execution begins.

If a PDS user runs a task in timesharing mode that issues privileged directives, and is not authorized with directive privilege, each privileged directive is rejected as it is met but the task continues to execute.

Note that all tasks executing in real-time mode are given directive privilege (and can be executive privileged) independently of the user's timesharing task privilege mask. You must have real-time privilege in order to start a real-time task.

4.8 Use of CTRL/C

CTRL/C is used to alert PDS or to cause some immediate action:

1 If PDS is not active at the terminal, CT/C activates it.

2 If a command is being typed, CT/C cancels all that has been typed and returns to the PDS prompt.

3 If a task is being run, it will be interrupted and PDS will prompt.

When CTRL/C is used to interrupt a task, the precise effect depends upon the type of system:

1 On a timesharing system the task is suspended. It will not perform any further processing until the CONTINUE command is used to resume the task.

2 On a multi-user or real-time system, the task continues to run. The task may terminate while commands are being issued to PDS. However, PDS will not recognize that the task has terminated until the CONTINUE or ABORT command is issued.

You can issue any legal PDS command in response to the PDS> prompt after typing CTRL/C. However, the system will only accept non-interruptable commands (see Table 4–2 if you have reached your task limit. This task limit is set by the System Manager; the default is one interruptable task.

An interruptable task is any timesharing task which is affected by CTRL/C. You can invoke these tasks by means of RUN or indirectly by means of an interruptable command, for example, MACRO. The effect of CTRL/C depends on the system as described above.

A non-interruptable task is any task which is invoked as a result of a non-interruptable command, see Section 4.6. These commands and tasks are not affected by CTRL/C. In general non-interruptable commands are those which take a short time to complete, for example, INSTALL, or which produce a comparatively small amount of terminal output, for example, SHOW. You can suppress the output from such commands by typing CTRL/O.

For example, with a task limit of 1 interruptable task:

```
1. Timesharing System

  PDS> RUN MYTASK
  CTRL/C
  TASK SUSPENDED
  PDS> DIR [1,100]
  COMMAND NOT ALLOWED - TASK SUSPENDED
  PDS> INS [11,1]MTAACP
  PDS> SHOW STATUS
  USER LEN UIC[1,1] TT03: 09:55:21 30-JUN-78
  JOB212 SIZE:4K CPU:0.00
  FILES OR DEVICES ASSIGNED
  FILE OR DEVICE  REDIRECTED  LUNS
  TT3:      -   NOLUNS
  DB0: IAS306   -     NOLUNS
  LP0:      -     NOLUNS
  PDS> CONTINUE

2. Multi-User System

  PDS> RUN MYTASK
  CTRL/C
  PDS> DIR[1,100]
  COMMAND NOT ALLOWED - MAXIMUM NUMBER OF TASKS ALREADY ACTIVE
  PDS> INS [11,1]MTAACP
  PDS> SHOW STATUS
  USER MIKE UIC[200,200] TT22: 10:31:22 4-JUL-90
  TT22A
  PDS>CONTINUE
```

4.8.1 Effect of CTRL/C on Type-ahead

This may be set at system generation time or on a per terminal basis by the system manager. The default is:

1 On a timesharing system, CTRL/C will flush type-ahead.

2 On multi-user or real-time systems type-ahead will not be flushed.

4.8.2 Effect of CTRL/C on Indirect Command Files

When you type CTRL/C during the processing of an indirect command file, the command active at that moment is suspended. However, if you then enter CONTINUE the remainder of the indirect command file is not executed because the remaining commands are flushed after typing CTRL/C.

4.9 PDS Dialup Support

If a dialup line is lost during an interactive session while you are logged in, the job is not lost but remains attached to the same line. If a task is running it will be suspended. If any user dials up and is connected to the same line, the following message is printed, followed by a PDS prompt:

```
USER username ALREADY LOGGED IN [WITH SUSPENDED TASK]
```

where:

- username = the name of the user currently logged in.

At this point, only two commands are valid:

- LOGOUT - to log out the user and free the terminal
- CONTINUE - to enter normal interactive mode. If there is a suspended task, the task will be continued and CTRL/C will have to be typed to suspend the task again. Before continuing, PDS prompts "PASSWORD?" and checks the user's password.

If no user is connected to the suspended line within the timeout limit set for PDS by the system manager, the user is logged out and the line disconnected. This also happens if the suspended task exits for any reason while the line is lost (for example, if it completes a mark time directive and exits).

If the running task attempts to perform input or output to the terminal during the short period (about one second) between the loss of the dialup line and the suspension of the task, it receives an error. Some commands (for example, the DIRECTORY command) terminate.

To keep the dialup line connected to a terminal after you have logged out, specify the LOGO/KEEP qualifier. You do not then have to dial in again before next logging in.

5 Batch Processing

5.1 Introduction

This chapter describes batch processing under IAS. Batch jobs are tasks that run from start to finish without user intervention. Section 5.2 describes the use of batch command. Sections Section 5.3 and Section 5.4 describe how to begin, end, and submit a batch job.

5.2 Batch Commands

Most PDS commands apply to both interactive and batch processing. (Refer to the command descriptions in Chapter 14.) Jobs executed using batch processing begin with the $JOB command and end with the $EOJ command. Batch commands must always begin with a dollar sign ($) in the first postion of a line.

As a batch user, you can submit a job in either of the following ways:

1 From an interactive terminal

2 By means of a card reader

When you submit the job from an interactive terminal, you must issue the SUBMIT command. This command submits a file of batch commands to the batch processor. See Section 5.4 for a full description of the SUBMIT command.

When submitting a job by means of a card reader, you include the batch commands in the input stream. For example:

```
$JOB GRAHAM CATJOB 3
$COBOL JOB
$EOJ
```

This example invokes the COBOL compiler to compile the source program held in the file JOB.CBL.

Jobs submitted to batch are placed in the queue and processed in priority order. If the system is configured for multistream batch (up to eight streams are available), the job is run in the first free batch stream. Subsequent jobs are assigned to batch streams, until all streams are exhausted. Other jobs remain in the queue until a batch stream is free.

If a single batch stream is in use, a job remains in the queue until all preceding jobs have been completed.

5.3 Beginning and Ending a Batch Job

The $JOB and $EOJ commands begin and end a single batch job.

5.3.1 The $JOB Command

The $JOB command marks the beginning of a batch job. Parameters to the $JOB command consist of:

1 User name

2 Job name

3 Job time limit

4 Job command mode (optional) in the following form:

```
$JOB/MCR or $JOB/DCL
```

If you do not specify MCR or DCL, the system assumes default mode. The system manager defines the default at system generation.

5 The batch password (optional) of the form:

```
$JOB/PASSWORD:password
```

An example of a $JOB command is:

```
$JOB CATHY TEST 3
```

In this example, CATHY is the user name and TEST is the job name. The number 3 instructs the system to terminate the job after it has used 3 minutes of elapsed time.

Your user name is a 1- to 12-character alphanumeric string that is unique to you; it is identical to your user name parameter to the LOGIN command. See Chapter 3.

The job name is a 1- to 12-character alphanumeric string that uniquely identifies the job. This prints on the operator console when the job starts (and on the batch log). The system does not use the job name.

If your user identity has a batch password associated with it, you must specify this in the $JOB command before the system can accept the job. For example:

```
$JOB/PASSWORD:SECRET SYSTEM ACCOUNTS 30
```

To change the command mode for a batch job within the job, use the $MCR or $DCL commands. However, if the batch job contains mostly MCR mode commands, use the /MCR qualifier.

5.3.2 The $EOJ Command

The $EOJ command terminates a batch job. The command has no parameters.

5.4 The Submit Command

The SUBMIT command submits a file of batch commands to the batch queue from an interactive terminal. For example:

```
PDS>SUBMIT BATCHJOB
```

Submit the file BATCHJOB to the batch queue for processing.

5.5 Batch Editing

IAS provides a batch-oriented editor to create and maintain source language files and data files on disk. This editor, called the source language input program (SLIPER), is described in Chapter 7.

6 File Handling

6.1 Introduction

In IAS, the file is the basic unit of information storage. This chapter describes IAS file handling, its associated commands, and their use.

A file is defined as an ordered collection of information. To store information (for example, a source program), you must create a file and enter the source program into the file. Any subsequent attempt to access or manipulate the source program must be made by identifying the file that contains it (that is, by supplying a file specification). A file specification provides the system with all the information it needs to identify the file, namely:

1 The device where the file is stored

2 The file directory

3 The file name

4 The file type

5 The version number

6.2 The IAS File System

The standard IAS file system for disks, DECtapes, and magnetic tapes is the Files-11 system. Files-11 magnetic tapes conform to American National Standard Magnetic Tape Labels and File Structure for Information Interchange, X3.27-1969. See the *IAS I/O Operations Reference Manual* for a detailed description of the Files-11 file system.

PDS commands default to Files-11. However, by using qualifiers (for example, DOS and RT11), a number of commands can use other file systems. The command descriptions in Chapter 14 explain these commands.

6.2.1 Volumes

The magnetic media where you store files are called volumes and include disks and magnetic tapes. To access a file held on a volume, you must mount the volume; that is, you must physically load the volume on a disk or tape drive and relate it to your task or terminal by using the MOUNT command (see Section 6.4.3). All references to this volume, in PDS commands, are then made by using the device name of the device where it is mounted. For example:

```
PDS> MOUNT DK0: MYDISK
PDS> COPY DK0:[200,200]MYFILE.DAT
TO? SY0:[100,100]THISFILE.DAT
```

You must mount volumes that do not hold files in Files-11 format, using the qualifier /FOREIGN.

6.2.2 File and Volume Protection

IAS protects individual user privacy and system security by providing a facility to restrict access to a volume. Magnetic tapes written in Files-11 format have a volume-level protection code; that is, the protection assigned to the volume applies equally to every file within it. Disks and DECtapes, however, have both an overall protection code for access to the volume and individual protection codes for each file within the volume.

For the purpose of assigning protection codes, IAS defines four types of access:

1 Read (R)—Read access.

2 Write (W)—Write access.

3 Extend (E)—Extend access. (You need extend access to install or run a task.)

4 Delete (D)—Delete access.

IAS defines four categories of user:

1 System

2 Owner

3 Group

4 World

The protection code identifies the type of access associated with each user category. The user categories are listed in Table 6–1.

Table 6–1 User Categories

User Category	Access Privilege
SYSTEM	All tasks that run under a system UIC (or group code of less then 10 octal).
OWNER	All tasks that run under the UIC of the owner of the file or volume.
GROUP	All tasks that run under a UIC that has the same group number as the UIC of the owner of the file or volume.
WORLD	All tasks.

The system uses the UIC to determine file ownership. The system determines your UIC from the user profile file (PDSUPF.DAT), which is defined by the system manager when you are authorized. The code is not necessarily unique to you.

Initially, the system manager applies volume protection when initializing the volume. However, you can respecify volume protection by using the MOUNT command. See Chapter 14 for a description of the MOUNT command.

You apply a file protection code when you create the file. If you do not explicitly specify a protection code for a new file, the system automatically applies the volume default code. This is normally SYSTEM:RWED, OWNER:RWED, GROUP:RWE, WORLD:R. You can subsequently modify the code by using the SET PROTECTION command. For example:

```
PDS> SET PROTECTION
FILE? MYFILE.DAT
PROTECTION? (SYS:RWED,WO:,G:RW)
```

The example above changes the protection code of the file MYFILE.DAT so that the system (SYS:) has all four types of access: world (WO:) is denied all types of access; group (G:) has read and write access; the allowed access of the owner does not change. Any category that is not mentioned keeps the access rights as previously allocated. You can specify the categories in any order. The above example illustrates the following conventions:

1 You must always enclose the protection code in parentheses.

2 You specify the four user categories using codes followed by colons. You can abbreviate the codes to one or more letters.

6.2.3 RMS-11 Files Management in IAS

IAS supports the record management system (RMS-11). RMS-11 is a suite of routines that manages three types of file organization.

1 Sequential

2 Relative

3 Indexed Sequential

The three types differ in the way the records within a file are accessed. The record is the basic unit of information handled by RMS-11.

1 Sequential is the default organization in IAS. To find a particular record, for example, when you use an IAS editor, you must access each record in sequential order until you locate the one you require.

2 Relative is an organization by block number. It enables you to access individual records directly and randomly.

3 Indexed sequential (indexed) enables you to handle, copy, or sort records, depending on the contents of previously specified fields (KEYS) within each record.

See the *RMS-11 User's Guide* for a full description of these types of file organization. The PDS commands APPEND, COPY, CREATE, and MERGE manipulate file organization.

See Chapter 14 for a description of these commands. For a description of the interface at program level, see the *RMS-11 MACRO Reference Manual.*

6.3 File Specifications

The file specification provides the system with the information it needs to perform the following functions:

1 Create a file.

2 Identify an existing file stored on a volume.

3 Read a file from (or write a file to) a device.

The format of a file specification is as follows:

```
dev:[ufd]name.typ;ver
```

where:

dev: Device name that specifies the name of the device where the volume holding the required file is mounted. These are listed in Table 6–2.

A logical name can replace the device field. See Section 6.4.2.1.

[ufd] User file directory (UFD) of the form [m,n], where m and n are octal numbers from 1 to 377 characters long.

name Name of the file; an alphanumeric character string from 1 to 9 characters long.

typ 1 to 3 alphanumeric character filetype that usually identifies some aspect of the file contents. Table 6–3 lists standard file types for IAS files. For example, the file type FTN indicates that the file contains a FORTRAN source program.

ver Version number; an octal number in the range 1 to 77777. The number distinguishes versions of the same file. For example, when you first create a file, the system assigns the file a version number of 1. If you subsequently open that file for editing the editor retains the original file for backup by creating the new file with the same file name and type, but with a version number of 2.

Table 6–2 IAS Device Mnemonics

Mnemonic	Device Type
AD	AD01 A/D converter
AF	AFC11 Analog input
CI	Console Input
CL	Console Log
CO	Console output
CR	Card reader
CT	Cassette
DB	RP04/5/6 disk
DD	TU58 cartridge
DF	RF11 disk
DK	RK05 disk
DL	RL01/2 disk
DM	RK06/7 disk
DP	RP02/3 disk
DR	RM02/RM03/RM05 disk
DS	RS03/4 disk
DT	DECtape
DU	RA60, RA70, RA80, RA81, RA82, RA90, RD31/32, RD51/52/53/54, RX50/33, RC25 disk
DY	RX02 floppy disk
DX	RX01 floppy disk
LB	Device holding system library files
LP	Line printer
LS	LPS A/D converter
MM	TU16, TE16, TU45, TU77 magnetic tapes
MO	Message output
MS	TS11, TS05 magnetic tape

Table 6–2 (Cont.) IAS Device Mnemonics

Mnemonic	Device Type
MT	TU10, TE10, TS03 magnetic tapes
MU	TU81, TU81-PLUS, TK50 magnetic tapes
NL	Null device
PP	Paper tape punch
PR	Paper tape reader
SP	Device holding spooled I/O files
SY	User's system disk
TI	User's data input stream
UD	UDC11 Universal Digital Control
WK	Fast-access device for work files

The mnemonics CI, CL, CO, LB, MO, NL, SP, SY, TI, TO, and WK, are logical device names (pseudo-devices), that can be made to refer to particular physical devices according to the needs of the installation. The null device is particularly useful for program testing. A program written to do I/O to a real device can temporarily be assigned to the null device (NL), while other parts of the program are being tested.

TI and TO are logical device names for your input and output data streams. For example, if you want to read from the terminal, specify TI:

```
PDS> COPY
FROM? TI:
TO? MYFILE.DAT
```

The above command transfers the input text typed at the terminal to the file named MYFILE.DAT. Table 6–3 lists all the standard IAS file types.

Table 6–3 Standard IAS File Types

Type	Description
BAS	A BASIC language source file
BP2	A BASIC-PLUS-2 compiler
BIS	A batch command file
CBL	A COBOL language source file
CMD	A file containing a list of commands
COR	A CORAL language source file
DAT	A data file
DIR	A directory file
EML	An editor macro Library
FLB	A form library file (for use with FMS-11)
FMD	A form description file (for use with FMS-11)
FRM	A form file (for use with FMS-11)
FTN	A FORTRAN or F4P language source file

Table 6–3 (Cont.) Standard IAS File Types

Type	Description
LST	A file in print-image format (listing)
MAC	A MACRO-11 assembly language source file
MAP	A file containing a memory allocation map
MLB	A macro library file
OBJ	An object module (output from MACRO-11, COBOL,FORTRAN, or CORAL)
ODL	An overlay description file
OLB	An object module library file
SAV	A saved system memory image file
SML	A system macro library file
SPR	A spooled output file
SRT	A SORT specification, input, or output file
STB	A symbol table file
SYS	A file for system use
TMP	A temporary file
TSK	A task image file produced by the task builder and suitable for execution

6.3.1 File Specification Defaults

You can omit the device name and UFD fields of a file specification. The system replaces the omitted fields with your default values. The following example outputs the file MYFILE.DAT;1 on the terminal. This file is held in UFD [100,100] on the volume mounted on device DK0 (RK05 unit).

```
PDS> SET DEFAULT DK0:[100,100]
PDS> TYPE MYFILE.DAT;1
```

You can also omit the version number. The system will assume the following:

1 The highest existing version number for an input file specification.

2 The highest existing version number increased by one for an output file specification or 1 if no previous version number exists.

Initially, the device and UFD defaults are determined as follows:

1 The system manager determines the default device for each user.

2 The default UFD is equivalent to the UIC associated with the user name (provided when logging in).

To modify these defaults on a timesharing system, use the SET DEFAULT command. See Chapter 14. You can change the default UIC/UFD on a multiuser system by using the SET UIC command. See Chapter 14.

Table 6–4 File Specification Defaults

Field	Default
device-name	Your system device when logging in. You can subsequently change this on a timesharing system by using the SET command. The new default device must have a volume mounted on it and you must have access to it. A multiuser system user cannot specify a default device.
ufd	The default UFD is equivalent to your UIC when logging in. You can subsequently change this by using the SET DEFAULT or SET UIC command. You must have access to any UFD selected as a default.
name	Cannot be defaulted.
filetype	Can be defaulted in the appropriate context. IAS has standard file types (see Table 6–3) that it uses as defaults in defined contexts.
version	For input specifications, the highest version number. For output specifications, the highest version increased by 1, or 1 if no previous version exists.

6.3.1.1 Changing Default Values

You can change the default device or UFD used in file specifications on a timesharing system at any time by using the SET command; see Chapter 14.

To change the default device on a timesharing system, type the following command:

```
PDS> SET DEFAULT device-name
```

where device-name is the new default device.

To change the default UFD on a timesharing system, type the following command:

```
PDS> SET DEFAULT ufd
```

where ufd is the new default UFD in the format [m,n] and m and n are octal numbers between 1 and 377. See Chapter 14 for a complete description of the SET command.

You cannot change the default device on a multiuser system. To change the default UFD with the default UIC, type the following command:

```
PDS> SET UIC uic
```

where uic is the new default UIC in the format [m,n]. m and n are octal numbers between 1 and 377.

6.3.1.2 Displaying Default Values (Timesharing Systems Only)

At an interactive timesharing terminal, you can display the current default values for the device and UFD field by using the SHOW command. For example:

```
PDS> SHOW DEFAULT
```

The system responds by displaying your default device and UFD.

6.3.2 Wildcards

Wildcards, denoted by an asterisk (*), enable you to specify more than one file in a single file specification.

You can only use wildcards in certain commands, as described in Chapter 14. Generally, these are file manipulation commands (for example, COPY).

6.3.2.1 Input Files

You can use a wildcard in any field of an input file specification except the device field. The following examples show the use of wildcards in input file specifications:

- DEL CATH.DAT;*—Delete all versions of the file named CATH.DAT stored on the default device and UFD.
- DIR DK1:[200,200]*.LST—Display information about all the highest versions of files on DK1: in UFD [200,200] that are of type LST.
- PRINT [30,4]*.MAC;*—Print all versions of the files on the default device in UFD [30,4] that are of type MAC.
- DELETE [*,*]TONY.DAT;*—Delete all versions of the file named TONY.DAT in every directory on the user's default device.
- DELETE [200,200]*,*;*—Delete all files in UFD [200,200] on the user's default device.
- COPY *[90,4]FORT.FOR;*—Illegal specification. The device field cannot be wild.

6.3.2.2 Output Files

When you use a wildcard to replace a field in an output file specification, the wildcard instructs the system to replace the field with the corresponding field in the input file specification. You can place an asterisk in any field of an output file specification except the device field. For example:

```
PDS> COPY CATH.DAT
TO? DK2:*.*
```

This example copies the highest version of the file CATH.DAT from the default device to DK2:. This file will also have the name CATH and the type DAT. If no version of CATH.DAT exists in the output file UFD, the version number of the output file is 1. If the output file UFD already contains one or more versions of CATH.DAT, the newly copied CATH.DAT is given a version number one greater than the previous highest version.

By placing a wildcard in the version of the output file specification, you instruct the system to retain the same version number as the input file. The system returns an error message if the output file UFD contains a file with the same name, type, and version number as the output file.

For example:

```
PDS> COPY
FROM? CATH.DAT
TO? DK2:*.*;*
```

6.3.3 Valid File Specifications

The fields of a file specification that you must supply depend on the type of file you are describing. The two types of files are:

1 Retrievable files written to or stored on disks, DECtapes or magnetic tapes. These files are called named files because they have file names that the system can access.

2 Files that are read from or written to record-oriented devices; for example, a card reader or a line printer, or files held on unlabelled tapes. These files are called unnamed files.

You must always supply the filename field of a named file; that is, you must give an alphanumeric filename or a wildcard (*). Many commands have a default value for the file type field. However, with any command that has no such default, you must always supply the file type field of a named file. The device, UFD, and version fields can be omitted because they have default values. See Section 6.3. You can also replace the device field with a logical name. See Section 6.4.2.1.

The use of wildcards in a file specification depends on the IAS command with which the file specification is issued. Where applicable, the command descriptions in Chapter 14 describe restrictions on the use of wildcards.

The specification of an unnamed file consists only of the device field, which can be a specific device or a logical name. See Section 6.4. If you supply any other field, the system ignores it because UFDs, file names, file types, and versions have significance only for named files. The device field cannot be wild.

6.4 Device Management

Before you can access any information held on a volume (disk or magtape), you must physically load the volume on a device. The device must be known to the system (use the SHOW DEVICES command to list all known devices). You must mount the volume (using the MOUNT command), except for system devices, which are allocated and mounted for all users.

NOTE: No device protection is enforced on multiuser systems. Therefore, if you have a volume mounted on a device, all users have access to the volume.

Volumes are mounted on devices and any data held on those volumes is referred to by PDS by the device name and the device unit number. Record-oriented devices (LP and CR) are referred to by device name. Hence, data is sent to or received from the device itself.

If a device is available, you gain access to it by "allocating" it (that is, by issuing a command that requests the system's permission to use it); see Section 6.4.2. An exception to this procedure occurs when you access a system device.

6.4.1 System Devices

A system device is allocated to all users by the system manager. For example, your system disk and the line printer are normally system devices.

A device such as a line printer cannot be shared by two users simultaneously; however, many users can access it at the same time. The system manager can therefore use a technique called spooling. In the case of a line printer, spooling causes all output written to the printer to be queued. The system creates disk files of all line printer output, maintains a queue containing a list of these files, and prints them one at a time.

You can defer the printing of queued files by using the command SET PRINTING DEFERRED (see Section 6.5.3.1). Deferred printing can be made the default at installations where, for example, the line printer is remote from you. In this case files to be printed are held in a user queue until you issue the SET PRINTING NODEFERRED command and log out. This causes all files in the line printer output queue to be added to the list of files to be printed.

6.4.2 Accessing a Non-System Device

To access a non-system device, you must meet the following requirements:

1 Have a means of obtaining access to the device (the ALLOCATE command, timesharing only, and the MOUNT command).

2 Have a means of keeping commands, especially in batch mode, independent of a particular physical device (logical device names).

3 Have a means of keeping the input/output statements in a program independent of a particular physical device (logical unit numbers).

On a timesharing system, you obtain access to a non-system device by issuing the ALLOCATE and/or the MOUNT command (see Chapter 14. Some devices, such as disk drives, are shareable. Thus, you can mount a disk even though it has already been mounted by another user (assuming the device is not allocated). The volume ceases to be known to the system when the last accessed user dismounts the volume.

On a multiuser system, when a volume is mounted on a device for one user it is automatically available for all users of the system. Any user can then dismount the volume if no files are currently being accessed.

You are granted exclusive access to nonshareable devices (for example, magtape).

6.4.2.1 Logical Device Names

IAS uses logical device names to enable your commands to be independent of a particular physical device. For example, you can specify TA: and use it in place of the corresponding physical device.

Once a logical device name is associated with a physical device name, you can use the logical device name in any command. If a logical device name is the same as a physical device name, IAS assumes that the reference is to the logical device name.

Logical device names can be defined in ALLOCATE (timesharing only) or MOUNT commands. A logical device name has the following syntax:

```
XX[nn]:
```

where XX represents two alphabetic characters and nn is an optional unit number (an octal number ranging from 0 to 77). If you omit nn, 0 is assumed. You can use logical device names only in PDS commands. You cannot use them from within a program or in file specifications input directly to a program.

6.4.2.2 Logical Units

All program I/O is performed on logical units. Logical units are identified by logical unit numbers (LUNs). Before you can use a logical unit for I/O, you must assign a physical device or file. Since different devices or files can be assigned to the logical units on successive runs of a program, the program itself can be device-independent.

You can assign logical units in three ways:

1 By using the ASG option during task build (LINK command).

2 By issuing an ASSIGN command (timesharing only).

3 By establishing the assignment within the program before the file in question is accessed.

You can use the LINK option and the ASSIGN command to assign a physical or logical device to a logical unit. From within a program, however, you can assign a named file to a logical unit. See one of the following manuals for further details:

- The *IAS Executive Facilities Reference Manual*
- The appropriate *IAS FORTRAN User's Guide*
- The *PDP-11 MACRO-11 Reference Manual*

6.4.3 Mounting a Volume on a Device

To access a file held on magnetic media, you must physically load and mount the volume where it is held. System devices that are already mounted when a user logs in are automatically mounted for you. On a timesharing system, for all other volumes, you must issue a MOUNT command to make the device available and gain access to the resident volume. However, on a multiuser system, once a volume has been mounted for one user it is available to all other users. For example:

```
PDS> MOUNT
DEVICE? DK2:
VOLUME-ID? TESTER
```

This example mounts the volume labeled TESTER on DK2:. You can now access any file on the mounted volume, as long as the file protection code permits access.

In this example, the MOUNT command indicates that the volume is in Files-11 format. Volumes in Files-11 format have a volume-identification on the medium itself. This is set when the volume is initialized. The volume-identification is used when the volume is mounted or dismounted.

On a timesharing system, you can omit the unit number in the device specification if you do not know or care on which unit the volume is to be mounted. If you omit the unit number in batch mode, you must then supply a logical name for the device; the logical name replaces the device name in subsequent file specifications. For example:

```
$MOUNT DK: TESTER AA0:
```

You assign the logical name AA0: to the unknown unit. You can now use the logical name instead of the physical device name in subsequent commands. In interactive mode, the system displays a message giving the unit where the volume was actually loaded.

You must specify the unit number on a multiuser system.

Files-11 disks and DECtapes are shareable volumes that more than one user can mount and access. Only one user at a time, however, can mount and access magnetic tape.

Only one user at a time can mount a foreign volume and must inform the system that it is foreign. Volumes that you mount foreign are normally referred to by some external label visible to the operator. For example:

```
PDS> MOUNT/FOREIGN
DEVICE? DT0:
VOLUME-ID? TAPEA
```

The command qualifier /FOREIGN informs the system that TAPEA is not to be accessed as a Files-11 volume and prevents other users from mounting the volume. The operator mounts the volume, with external label TAPEA, on drive DT0:. When a foreign volume is mounted, the system cannot check that the label is correct and that the right physical volume has been mounted.

If the foreign volume is in DOS or RT-11 format, file qualifiers to the COPY, DELETE, and DIRECTORY commands enable you to access files held on the volume. Otherwise, most PDS commands do not apply to foreign files. See the specification of the MOUNT command in Chapter 14 for further information.

6.4.4 Dismounting a Volume

When you have finished accessing a volume, issue the DISMOUNT command to dismount the volume and make the device available for other users.

On a timesharing system, the DISMOUNT command automatically deallocates the device (if it was allocated) unless you specify the qualifier /KEEP. For example:

- Example 1:

```
PDS> DISMOUNT
DEVICE? DK0:
```

- Example 2:

```
$DISMOUNT DT0: TAPEA
```

On a multiuser system, any user can dismount the mounted volume. The volume is dismounted for all users and no protection is enforced by the system unless another user is currently accessing a file.

The parameters for DISMOUNT are the device specification or logical name of the device you want to dismount and the volume identification. You can omit the volume identification.

6.4.5 Allocating a Device (Timesharing Systems Only)

If a device is not a system device and it cannot be mounted, you must use the ALLOCATE command to access the device. This is only applicable to timesharing systems. For example:

```
PDS> ALLOCATE
DEVICE? LP1:
```

The above example allocates a line printer to the user. No one else can use this line printer until the user who allocated it issues a DEALLOCATE command. See Section 6.4.6.

You can also use the ALLOCATE command to obtain exclusive access to a shareable device. For example:

```
$ALLOCATE DK: MC0:
DK3: ALLOCATED
$MOUNT MC0 VOL1
```

In the above example, a batch user has allocated an RK05-type disk drive and assigned it the logical name MC0:. No one else can to access that drive until is has been deallocated. PDS announces which physical device has been allocated for exclusive use to the user, in this case DK3.

Once a device has been allocated, you can mount several volumes one after the other. For example:

```
$ALLOCATE DK: DV1:
$MOUNT DV1: VOL1
    .
    .
    .
$DISMOUNT/KEEP DV1:
$MOUNT DV1: VOL2
    .
    .
    .
$DISMOUNT DV1:
```

In this example, the user obtains exclusive access to a disk drive using the ALLOCATE command. A volume labeled VOL1 is then mounted on the drive. When the user dismounts VOL1, the /KEEP qualifier retains the user's exclusive access to the disk. When VOL2 is dismounted, however, the disk is deallocated as the user has not specified the /KEEP qualifier.

6.4.6 Deallocating a Device (Timesharing Systems Only)

After issuing an ALLOCATE command to obtain exclusive use of a nonmountable device, for example, a line printer, you must issue the DEALLOATE command to free the device. This is applicable only to timesharing systems. For example:

```
$ALLOCATE/DEVICE LP1:
    .
    .
    .
$DEALLOCATE/DEVICE LP1:
```

The DISMOUNT command automatically deallocates an allocated mountable device unless you specify the /KEEP qualifier. For example:

```
$ALLOCATE/DEVICE DK: MC0:
$MOUNT MC0: CATH
    .
    .
    .
$DISMOUNT/KEEP MC0:
$DEALLOCATE MC0:
```

6.4.7 Assigning Logical Unit Numbers to a Device (Timesharing Systems Only)

Use the ASSIGN command to associate a logical or physical device with a logical unit. This is applicable only to timesharing systems. See Section 6.4.2.1 and Section 6.4.2.2 for a definition of logical devices and logical units. For example:

```
PDS> ASSIGN
DEVICE? LP0:
LUN? 6
```

This command assigns LP0: to the logical unit 6. For example, if a program writes to logical unit 6 via the FORTRAN statement WRITE, the results of the write are printed on the line printer.

6.5 File Management

6.5.1 Creating Files

Both batch and interactive users can use the CREATE command to create files.

If you are an interactive user, type CREATE and supply a file specification (no wildcards are allowed). You can optionally modify the file specification by the /PROTECTION qualifier. If you do not specify the /PROTECTION qualifier, the new file is assigned the default file protection associated with the volume. For example:

```
PDS> CREATE
FILE> FORT.FTN/PRO:(OW:RWED SY: GR: WO:)
```

The system uses default values for the device, UFD, and version fields. See Section 6.3.1.

Once you have terminated the command string, you type input to the new file line by line.

When terminated, each line is sent to the file exactly as formatted at the terminal. When the file is complete (that is, you have typed all the required input), you close the file by typing CTRL/Z.

If you are a batch user, you supply the command name optionally modified by /DOLLARS and a file specification (no wild-cards are allowed). You can optionally modify the file specification by using the /PROTECTION qualifier. The qualifier /DOLLARS indicates that the file will be closed by the $EOD command. Otherwise, any batch command terminates the file. Therefore, you must specify the /DOLLARS qualifier whenever a record in the file you are creating contains a $ in position 1. See Chapter 14 for further information on the CREATE command qualifiers. For example:

- Example 1:

```
$CREATE/DOLLARS FORTRAN.FTN/PRO:(OW:RWED SY: GR: WO:)
   .
   .
   .
$EOD
```

- Example 2:

```
$CREATE DK2:[30,4]CALCULATE.MAC
```

6.5.1.1 Using the Editor to Create a Sequential File

You can also create files by means of the EDI command. See Chapter 7 for a description of the editors available with IAS.

6.5.1.2 User File Directories

To create a file on a volume, you must mount the volume (see Section 6.4.3), and you must have write access to a *user file directory* (UFD) on the volume. A UFD is a file that contains details of all the files that have been created on that volume under the UFD identifier.

You can issue the DIRECTORY command to display the contents of a UFD. In batch mode, the listing is output on the batch log; in interactive mode, the listing is output to the terminal.

```
PDS> DIRECTORY

DIRECTORY DB0:[200,22]

15-MAY-90 17:20

ADD.OBJ;1       2.      15-MAY-90 17:17
ADD.FTN;1       1.      15-MAY-90 17:17
ADD.TSK;1       32.  C  15-MAY-90 17:18
```

```
TOTAL OF 35./35. BLOCKS IN 3. FILES
```

If you do not supply a parameter, the system displays information about your current default UFD. However, by supplying one or more file specifications, you can list other directories or specific files. For example:

```
PDS> DIRECTORY ADD.OBJ

DIRECTORY DB0:[200,22]

15-MAY-90 17:20

ADD.OBJ;1      2.      15-MAY-90 17:17

TOTAL OF 2./2. BLOCKS IN 1. FILE
```

To list DOS or RT-11 files, you modify the file specification with the /DOS or /RT11 file qualifier. For example:

```
PDS> DIRECTORY ESC
FILE? RTFILE.MAC/RT11
```

A UFD, like any other file, has a protection code that determines who has access to it. A file can be created under any UFD where you have write access.

6.5.2 Manipulating Files

You can manipulate files to perform the following functions:

1. Append one or more files to an existing file.
2. Copy a file.
3. Rename an existing file.
4. Merge a file with an existing INDEXED or RELATIVE file.

6.5.2.1 The APPEND Command

Use the APPEND command to add one or more files onto the end of an existing file. For example:

- Example 1:

```
PDS> APPEND (A.CBL, B.CBL)
TO? C.CBL
```

  Append files A.CBL and B.CBL to the end of the file C.CBL.

- Example 2:

```
$APPEND MYFILE.MAC YOURFILE.MAC
```

  Append MYFILE.MAC to the end of YOURFILE.MAC.

You must have extend access to a file before the file can be appended.

You specify the input file (or files) first and then the output file. If more than one input file is specified, enclose these in parentheses.

You can retrieve input files from a mounted volume, from a record-oriented device (for example, a card reader) or typed in from an interactive terminal. When you supply more than one input file, the system appends the files in the order specified.

If one of the files is to be input from your terminal (TI), the system transfers everything typed at the terminal after the command string, until you type Ctrl/Z to close the file. For example:

- Example 1:

```
$APPEND (FILE1.MAC, FILE2.MAC), FILE3.MAC
```

  The system appends the input files FILE1.MAC and FILE2.MAC to the file FILE3.MAC.

- Example 2:

```
PDS> APPEND
FILE? JUD.CBL
TO? GRAVES.CBL
```

  The file JUD.CBL is appended to the output file GRAVES.CBL.

- Example 3:

```
PDS> APPEND TI: FRED.MAC
;THESE LINES ARE TO BE
;APPENDED TO THE FILE
;FRED.MAC
Ctrl/Z
PDS>
```

6.5.2.2 The COPY Command

The COPY command creates a duplicate of the contents of an input file in a specified output file. If you omit the version number from the input file, the highest version number is used. If you omit the version number from the output file, the system uses the highest version number plus one. If the file does not already exist in the destination UFD, a version number of 1 is used. Optional command qualifiers allow the output file to be modified in various ways. For example:

- Example 1:

```
PDS> COPY
FROM? MT2:FRED.MAC
TO? DK2:JIM.MAC
```

- Example 2:

```
$COPY MT2:FRED.MAC,DK2:JIM.MAC
```

The examples above copy the highest version of FRED.MAC on MT2: to DK2: and change the file name to JIM on DK2:. In addition to copying from one device to another, you can use the COPY command to copy a file from one UFD to another. For example:

```
PDS> COPY [30,4]FRED.MAC
TO? [100,100]FRED.MAC
```

This example copies the file FRED.MAC in [30,4] to UFD [100,100]. The filename remains unchanged.

Four of the COPY command qualifiers are:

- /ALLOCATION:n
- /CONTIGUOUS
- /OWN
- /REPLACE

These (and the full list of COPY command qualifiers) are described in detail in Chapter 14. Examples of their use are as follows:

- Example 1:

```
$COPY/ALLOCATION:20 DK2:OLDFILE.DAT DK0:OLDFILE.DAT
```

  Copy OLDFILE.DAT from DK2: to DK0: and make the output file 20 blocks long. The /ALLOCATION qualifier is useful for copying a file and changing its size.

- Example 2:

```
PDS> COPY/CONTIGUOUS
FROM? MT2:TU71.MAC DK1:*.*
```

  Copy TU71.MAC from MT2: to DK1: and make the output file contiguous. The wildcards (*) indicate that the fields of the output specification where they occur take the corresponding field values of the input file specification; that is, the output file will also be named TU71.MAC.

- Example 3:

```
PDS> COPY/OWN DB0:[120,71]*.TXT
TO? DB1:[136,120]*.TXT
```

  Copy all files of the file type .TXT from the UFD [120,71] on DB0: to the UFD [136,120] on DB1:, and make UFD [136,120] the owner of these copies.

- Example 4:

```
$COPY/REPLACE MT1:SAME.OBJ;4 DK2:SAME.OBJ;4
```

  The /REPLACE qualifier indicates that the output file overrides a file in the user's default UFD that has the same name, type, and version number. That is, if a file called SAME.OBJ;4 already exists on DK2: in the default UFD, it is deleted and replaced by the new one copied from MT1:.

Two file qualifiers that enable you to copy files to or from an RT-11 or DOS formatted volume are available with the COPY command: /RT11 and /DOS. The qualifier must modify the specification of the file currently in DOS or RT-11 format. When a DOS or RT-11 file is copied, the output filename and file type are always taken from the input file. Therefore, the filename and file type fields of the output file specification must always be wild.

- Example 1:

```
PDS> COPY
FROM? DK2:FRED.DAT/RT11
TO? *.*
```

  Copy the RT-11 file FRED.DAT from the foreign volume on DK2: to the user's default device and UFD.

- Example 2:

```
$COPY TEST.MAC;8 DT0:*.*/DOS
```

  Copy the Files-11 file TEST.MAC;8 to a DOS-formatted foreign volume on DT0:.

6.5.2.3 The RENAME Command

The RENAME command changes the name of a file. The following examples change the file name DEBUG.MAC;1 to RUN.MAC;1.

- Example 1:

```
$RENAME DEBUG.MAC;1 RUN.MAC;1
```

- Example 2:

```
PDS> RENAME
OLD? DEBUG.MAC;1
NEW? RUN.MAC;1
```

6.5.2.4 The MERGE Command

The MERGE command merges a sequentially indexed or relative file (known as the transaction file) with an indexed or relative file (known as the target file). For example:

- Example 1:

```
PDS> MERGE ANN.DAT KATHY.DAT/RELATIVE
```

  Merge the file ANN.DAT with the relative file KATHY.DAT.

- Example 2:

```
PDS> MERGE/LOG
FILE? JEN.DAT/INDEXED
INTO? JOHN.DAT/INDEXED
```

  Merge the indexed file JEN.DAT with the indexed file JOHN.DAT. The /LOG qualifier sends an error log, giving details of records that could not be merged, to the user's terminal.

6.5.3 Listing Files

You can list sequential files on a line printer or at your terminal. You can use either the PRINT, TYPE or DUMP command; the choice depends on the type of listing you want and whether you are operating in interactive or batch mode.

6.5.3.1 Listing Files on the Line Printer

You can use the PRINT command to print files on a line printer. The system might queue all line printer output until all output previously submitted to the queue has been processed. Normally, the output files are printed in the order they were submitted to the queue. However, you can specify that files are only to be printed after a certain time by using the command qualifier /AFTER: time.
The PRINT command is the simplest way to queue a file to the line printer. For example:

- Example 1:

```
PDS> PRINT
FILE? FILE1.DAT,FILE2.DAT,FILE3.DAT
```

- Example 2:

```
$PRINT LIST.MAP
```

- Example 3:

```
PDS> PRINT/AFTER:10:30 FILE.LST
```

You specify the file or files to be printed after the command. The PRINT command always prints the file on the system default line printer, selected by the system manager. You must use the QUEUE command to print on any other device.

The PRINT command provides an option to delete files after they have been printed. You invoke this option by supplying the command qualifier /DELETE. For example:

```
$PRINT/DELETE MYFILE.DAT
```

You can defer printing by issuing the SET PRINTING DEFERRED command. Printing then begins when you log out (by choice or timeout) or you issue the SET PRINTING NODEFERRED command.

6.5.3.2 Printing Files on Varied Stationery

A single printer can have up to seven print queues. Each queue can be associated with a particular type of continuous stationery; for example, fan-fold, graph plotter paper, and pay slips. These queues are referred to by a number, with a value from 0 to 6. 0 is always the default queue. Values 1 to 6 can be used only by means of the PRINT or the QUEUE command or the PRINT$ MACRO directive.

When output spooling is enabled, the system first prints all the 0 queue on the CL device. If printing is queued with other values of n, the system informs the operator by means of the system console whenever a change of stationery is required.

6.5.3.3 Listing Files at an Interactive Terminal

The TYPE command causes one or more specified files to be printed on your interactive terminal. For example:

- Example 1:

```
PDS> TYPE
FILE? FIRST.MAC,SECOND.MAC
```

- Example 2:

```
PDS> TYPE TYPE.CBL
```

If you specify batch mode, the file is listed in the batch log.

6.5.3.4 The DUMP Facility

The DUMP command outputs a specified file on your terminal (TO) or sends the listing to a specified output file. The output shows the internal form of the file, byte by byte; it does not list the file. Command qualifiers modify the form of the listing. For example, you might specify that the file be dumped in ASCII mode. The DUMP facility is useful for debugging programs and for displaying nonprintable characters in ASCII or octal format. See the full specification of the DUMP command in Chapter 14 for all the available options. For example:

- Example 1:

```
PDS> DUMP/ASCII
FILE? DUMP.CBL
```

  List the file DUMP.CBL in ASCII format on the user's terminal.

- Example 2:

```
$DUMP/BYTE/OUTPUT:DK2:DISKFILE.DAT OBJECT.OBJ
```

  Send a listing of the file OBJECT.DAT in byte octal format to a file named DISKFILE.DAT on DK2:

- Example 3:

```
PDS> DUMP/OUT:LP0: FILE.DAT
```

List the file FILE.DAT in word octal format (the default) on the line printer.

6.5.4 Deleting Files

The DELETE command deletes files held on Files-11 disks or DECtapes, or RT-11 or DOS files held on foreign disks or DECtapes.

You must modify specifications of DOS or RT-11 files by a file qualifier, either /DOS or /RT11 as appropriate.

Wildcards (*) (see Section 6.3 are allowed in the file specification. If you omit the version field, you can supply the command qualifier /KEEP:n to preserve the highest n versions of the file or files specified. For example:

- Example 1:

```
PDS> DELETE/KEEP:2
FILE? MATRIX.DAT
```

Delete all versions of the file MATRIX.DAT less than or equal to m-n + 1, where m is the latest version of the file. If the latest version of MATRIX.DAT is 100, this example deletes all versions up to and including version 98. Versions 99 and 100 are kept, but if version 99 does not exist, then only version 100 is kept.

- Example 2:

```
$DELETE ROW.OBJ;4 COLUMN.MAC;4 PEEK.*;*
```

Delete all files named PEEK and the fourth version of the files ROW.OBJ and COLUMN.MAC.

- Example 3:

```
PDS> DELETE DK2:DOSFIL.DAT/DOS
```

Delete the file DK2:DOSFIL.DAT, which is in DOS format.

You can use the PRINT command modified by the /DELETE qualifier to delete files that have been submitted to the line printer. See Section 6.5.3.1 for further details.

Table 6–5 Summary of File Handling Commands

Command	Function
ALLOCATE	Allocate a specified device to the user (timesharing systems only).
APPEND	Add one or more files to the end of a specified sequential file.
ASSIGN	Assign a LUN to a device (timesharing systems only).
COMPARE	Compare two files and produce a summary of differences found.
COPY	Copy an input file to a specified output file.
CREATE	Create a file as specified.
DEALLOCATE	Deallocate a specified device (timesharing systems only).
DEASSIGN	Deassign a LUN from a device (timesharing systems only).
DELETE	Delete specified file.

Table 6–5 (Cont.) Summary of File Handling Commands

Command	Function
DISMOUNT	Dismount a specified volume.
DUMP	List the contents of a file in internal form.
EDIT	Edit an existing file or create a new file.
INITIALIZE	Initialize a volume.
MERGE	Merge a file with an existing indexed or relative file.
MOUNT	Make a volume available to the user (all users on a multiuser system).
QUEUE	Enter a file into a queue.
PRINT	Print one or more files on the line printer.
RENAME	Change the name of an existing file.
SET END_OF_FILE	Specify a file's end_of_file position.
SET PROTECTION	Assign a specified protection code to a file.
SORT	Sort contents of a file into a specified sequence.
TRUNCATE	Truncate one or more files back to their logical end_of_file points.
TYPE	List a file at the user terminal.

7 IAS Editors

7.1 Introduction

This chapter introduces the four IAS editors:

1 The text editor (EDI), primarily for interactive use.

2 The source language input program and editor (SLIPER), a batch-oriented editor.

3 The DEC editor (EDT), an alternative interactive editor.

4 The keypad editor (KED or K52), another interactive editor.

7.2 The Text Editor (EDI)

The EDIT command automatically invokes the text editor (EDI) unless you specify the qualifiers /SLIPER, /EDT, /KED, or /K52. EDI is an interactive, context-editing program that uses editor commands to create and modify source programs and other files containing ASCII data. The *RSX-11M/M-PLUS Utilities Manual* contains a complete description of the text editor (EDI).

7.3 Batch Editing

The source language input program and editor (SLIPER) is a batch-oriented editing program used to create and maintain source language files on disk.

The *IAS Utilities Manual* contains a complete description of SLIPER.

7.4 The DEC Editor (EDT)

The DEC editor (EDT) is a text editor that includes, among other features, the following functions:

- Optional keypad editing.
- Journaling facility.
- Startup command files.
- User-defined key functions.

The DEC Editor (EDT) is fully described in the *EDT Editor Manual*.

7.5 The Keypad Editor (KED OR K52)

The keypad editor (KED or K52) is a text editor. You use KED on a VT100 terminal and K52 on a VT52 terminal. The keypad editor is described in detail in the *PDP-11 Keypad Editor User's Guide*.

8 Introduction to Program Control

8.1 Introduction

This chapter introduces language-independent aspects of running programs under IAS. The next five chapters, one on each language, describe the use of IAS commands for transforming source programs into executing programs or tasks. The languages described are:

1 BASIC-11—Chapter 9

2 COBOL—Chapter 10

3 FORTRAN—Chapter 11

4 MACRO-11—Chapter 12

5 CORAL 66—Chapter 13

MACRO-11 is part of IAS; the other language translators are optional.

8.2 Processing Modes

The two processing modes are interactive and batch. The decision to use batch or interactive mode depends on the nature of the job and the installation requirements. Interactive mode is convenient for complicated editing of source programs or the execution of programs that require small amounts of input data. Batch processing is better suited to processing large amounts of data (for example, a payroll or accounts receivable package).

8.3 Indirect Command Files

An indirect command file is a sequential file containing command input. For example, rather than repeatedly typing commonly used command sequences, you can type the sequence once and store it in a file. To execute the sequence, you type an @ character followed by the file specification. You can invoke the indirect file from any position within the command string, but any characters that follow the indirect file specification are ignored. The system then retrieves the indirect file and executes the commands. For example:

```
PDS> EDIT FILE.CMD
[CREATING NEW FILE]
INPUT
FORTRAN/OBJECT/LIST:CPROG CPROG
LINK CPROG
RUN CPROG
[RETURN]
*EXIT
PDS>
   .
   .
   .
PDS> @FILE
```

The indirect file called FILE.CMD, created by means of the line text editor, contains commands to compile, link, and run the source program CPROG.FTN.

When you invoke the file, you can execute these commands by typing @FILE in response to the PDS prompt. CMD is the default file type for indirect files.

In batch mode, you can create and invoke the same command sequence in the following manner:

```
$CREATE/DOLLARS FILE.CMD
$FORTRAN/OBJECT/LIST:CPROG CPROG
$LINK CPROG
$RUN CPROG
$EOD
   .
   .
   .
$@FILE
```

The $CREATE command string must include the qualifier /DOLLARS, so that the system recognizes the following text as input and not as further batch commands to be processed. The $EOD command terminates the file to be created.

You can subsequently invoke the command file by the command line $@FILE.

Both batch and interactive users can invoke indirect files up to three levels. An indirect file can itself invoke another indirect file; the second file can invoke a third; but the third file cannot invoke a fourth indirect file.

8.4 User Libraries

The LIBRARIAN command enables you to create and maintain your own libraries of commonly-used macros (macro libraries) and routines (object module libraries).

8.4.1 Macro Libraries

MACRO-11 macros can be held in source (text) form in a macro library. Each macro is identified by its macro name. To use one or more macros contained in a macro library file, you must supply the library file specification, modified by the qualifier /LIBRARY, in the list of input files to the MACRO command. See the description of the MACRO command in Chapter 14. You must specify the macro library before the calling module. For example:

```
PDS> MACRO/LI:CAROL/OBJECT MYLIB.MLB/LIB+NEWPROG
```

This example assembles the source program in NEWPROG.MAC and uses macros defined in MYLIB.MLB.

8.4.2 Object Module Libraries

You can store commonly used routines in object module libraries. You can store source code that has been assembled or compiled to form object modules in a library, then incorporate it into a task.

If you use a library object module, you must ensure that the module is linked at task build time. The task builder automatically searches all system libraries; however, it only searches user-written libraries that have been explicitly specified in the LINK command. See the description of the LINK command, Chapter 14. For example:

```
PDS> LINK/MAP:FRANK PROG,MYLIB/LIB:(COMRTN)
```

This example links object modules PROG and COMRTN (stored in MYLIB) to create task PROG.TSK and FRANK.MAP.

The *IAS Task Builder Reference Manual* describes object module libraries in detail. The specification of the LIBRARIAN command in Chapter 14 describes how to create and maintain the libraries.

8.5 Creating Source Files

You can use the CREATE or EDIT commands to create source files. The EDIT command has the advantage that it provides immediate access to all its editing facilities. However, to correct errors made while using the CREATE command, you must rely on keyboard facilities or close the file and then issue the EDIT command.

8.6 The CREATE Command

To create a source file with the CREATE command, you must take the following steps:

Batch Mode

1 Issue the $CREATE command, optionally modified by the qualifier /DOLLARS, followed by a file specification of the file to be created.

2 Insert the source program immediately after the command line.

3 Terminate the source file either by another batch command or, if /DOLLARS has been specified, by the command $EOD.

Interactive Mode

1 Issue the $CREATE command, followed by the file specification of the file to be created.

2 Input the source program at the beginning of the next line.

3 Close the file by typing Ctrl/Z.

The CREATE command is described in detail in Chapter 14. The following example shows the use of the CREATE command in batch and interactive modes:

- Batch mode:

```
$CREATE/DOLLARS COBOL.CBL
   .
   .
   .
00078  IF NF-DELIMITER = CR
00079  PERFORM READ-TRAN-LINE
00080  IF EOFFOUND GO TO G5999
00081  ELSE GO TO GS5
00082  IF CHAR-COUNT ZERO
00083  IF INMARKER < TRAN-LINE-LIMIT GO TO G25.
   .
   .
   .
$EOD
```

- Interactive mode:

```
PDS> CREATE
FILE? TEST.FTN
  SUBROUTINE PROCI
C FIRST DATA PROCESSING ROUTINE
C COMMUNICATION REGION
  COMMON/DTA/A(200),I
    .
    .
    .
RETURN
END
CTRL/Z
```

8.6.1 The EDIT Command

The EDIT command enables interactive users to create and edit a source file by means of the text editor. Batch users should use the CREATE command to create a source file, which can be edited subsequently in either interactive or batch mode; see Chapter 7.

When the EDIT command specifies a nonexistent file, the line text editor creates a file and prompts for input. For example:

```
PDS> EDIT
FILE? NEWSOURCE.CBL
[CREATING NEW FILE]
INPUT
```

You then begin to enter the source file beginning at the first position of the next line after "INPUT".

See Chapter 7 for details on how the text editor is used to edit the new file as it is being created.

To close the new file, you must type carriage return as the first character in the line. This action causes the Editor to display an asterisk (*), which indicates that it expects an editor command rather than further input to the file because the command mode has changed from insert to edit. To close the file and exit to PDS, use the EXIT command. If you want to create further files, reissue the EDIT command. For example:

```
PDS> EDIT
FILE? TONY.FTN
[CREATING NEW FILE]
INPUT
  SUBROUTINE REPORT
C INTERIM REPORT PROGRAM
C COMMUNICATION REGION
  COMMON/DTA/A(200),
  RETURN
  END
RETURN
*EX
[EXIT]
PDS>
```

8.7 Error Status Returned to PDS

When certain tasks exit, it is possible for the system to notify PDS (and hence the user) of the worst error found during execution. The system relies on the task using the Exit with Status Directive (see the *IAS System Directives Reference Manual*). The status of the task is recorded as one of the following:

- SUCCESS
- WARNING
- ERROR
- SEVERE_ERROR

If you do not implement exit-with-status in the task, no status is recorded.

One of the following messages is received:

1 SUCCESS—Indicates that results are as expected.

2 WARNING—Indicates that the task has succeeded but results might not be as expected.

3 ERROR—Stronger than WARNING: results are unlikely to be as expected.

4 SEVERE_ERROR—Indicates one or more fatal errors or that the task was aborted.

If you invoke the task interactively, the termination message to your terminal includes the status. For example:

```
ON ERROR GOTO ERRPR
LINK /MAP:MYPROG MYPROG
ERRPR:PRINT MYPROG.MAP
```

If you invoke the task by means of an indirect command or in batch, you can use the status to control subsequent steps in the command file or batch job, (see Section 8.7.1).

8.7.1 Conditional Command Execution

Indirect and batch command files can include the following control commands:

- ON status action
- STOP
- GOTO label
- CONTINUE

Of these, ON is a conditional command; that is, the ON action takes place only if a particular condition is satisfied. The condition can be satisfied either by the status returned by a task (by means of exit-with-status), or as a result of a PDS command. For example:

```
$ON WARNING GOTO LAB1
$MAC/LI:MYPROG MYPROG
$ON ERROR GOTO LAB2
$LANK/MAP:MYPROG MYPROG
$GOTO LAB3
$LAB1: PRINT MYPROG
$GOTO LAB3
$LAB2: DIR MYPROG.*
$LAB3:
$EOJ
```

In this example, if a warning error is returned from the assembly (MACRO) of MYPROG, MYPROG.LST is printed and the job terminated. If the assembly is succesful, because the next command contains an error (that is, LANK instead LINK) a directory listing of MYPROG.* is printed (LAB2:) and the job terminates.

The ON command should be positioned (in the command stream) before any command(s) to which the condition and action apply. For example:

```
$DIR *.*
$ON ERROR STOP
$MAC MYPROG
$LINK MYPROG
$RUN MYPROG
```

In this example, a directory of the default UFD is always listed. However, if an error occurs during the assembly or task build (MACRO or LINK), the default condition and action is:

```
[$]ON ERROR STOP
```

If no ON command was specified and an error was encountered during execution, no further commands are executed.

The ON command condition and action remains in force until any of the following actions occurs:

- Another ON command is specified.
- PDS terminates ($EOJ and LOGOUT).
- In interactive mode, the top level of command input (that is, command files other than indirect command files) is entered.

If an ON command string contains an error (for example, ON ERRRR CONTINUE) the statement is ignored. If the error is in the action to be taken (for example, ON WARNING CONTINUE) the default condition (ON ERROR STOP) is taken (when this action is required). For example:

- Example 1:

```
ON ERROR TYPE MESFIL.CMD
MAC MYPROG
ON ERRRR CONTINUE
MAC YOURPROG
    .
    .
    .
```

  The second ON statement is ignored and MESFIL.CMD typed if an error is encountered during assembly of either MYPROG or YOURPROG.

- Example 2:

```
ON ERROR CONTINUE
MAC MYPROG
    .
    .
    .
```

  If an error is encountered during assembly of MYPROG, the command file is terminated because, on meeting the condition and attempting to "CONTINUE", PDS reverted to the default ON ERROR STOP.

ON cannot specify the action to be taken during the execution of a task or actions which depend on the outcome of a previous command.

The action set by the ON command can be any legal PDS (DCL) command. However, the following three commands are commonly used with the ON command:

1 STOP.—Prevents all further commands in the file or job being executed.

2 CONTINUE.—Causes the job to continue as though the condition had not occurred (used to override previous conditions set or the default).

  NOTE: When CONTINUE is used in a command file, it does not imply that a task has been previously suspended.

3 GOTO label.—Continues execution from the command immediately following the label specified. A labelled command line is of the form:

  label: command

  The label specified in a GOTO command must appear in the command stream as a label to another command. The labeled command line must occur after any GOTO statement that references it, because the GOTO statement cannot go back through the command stream.

If no action is required when an error is met, specify the following:

```
ON SEVERE_ERROR CONTINUE
```

9 BASIC-11

9.1 Introduction

This chapter introduces BASIC-11. For details about BASIC-11 and BASIC-PLUS-2, see the *IAS/RSX BASIC User's Guide* and the *BASIC-PLUS-2/RSX-11M/IAS/VMS User's Guide*.

BASIC-11 provides immediate translation and storage of a user program while it is being input from an interactive terminal. The PDS user invokes the BASIC interpreter by typing the command BASIC. You cannot use BASIC systems in batch mode under IAS.

The interactive nature of BASIC removes the need for separate steps in the development of a program. Once you have invoked BASIC, you can create, translate, and run a program in a single session.

This chapter describes how to invoke BASIC, create and execute a program, then terminate a session. The following manuals describe the BASIC language:

- *BASIC-11 Language Reference Manual*
- *IAS/RSX BASIC User's Guide*

9.2 The Basic Command

When you issue the BASIC command, BASIC displays the following:

```
PDS> BASIC
IAS BASIC V02-01
READY
```

The text READY indicates that BASIC is ready to receive a command or program line.

The BASIC command has no parameters or command qualifiers.

9.3 Ctrl/C

If you type Ctrl/C while a BASIC program is running, the system stops executing after the current line and displays the number of the last line executed. You can then issue further BASIC commands.

Typing Ctrl/C during the execution of a BASIC LIST, SAVE command, or immediate mode statement stops the execution of those commands or statements. Ctrl/C has no effect on the execution of other BASIC commands.

9.4 Terminating a Basic Session

To terminate a BASIC session and return control to PDS, you must type BYE on a new line. The system then prints information about the session and prompts for further PDS commands. For example:

```
BYE

15:57:32 SIZE:14K CPU:10.24

PDS>
```

9.5 Example

```
PDS> BASIC
READY
OLD MYBASIC
LISTNH
10 REM PROGRAM TO TRANSLATE MONTH NAMES TO NUMBERS
50 T$ = "JANFEBMARAPRMAYJUNJULAUGSEPOCTNOVDEC"
100 PRINT "TYPE THE FIRST 3 LETTERS OF A MONTH";
110 INPUT M$
120 IF LEN (M$) <>3 GO TO 200
130 M=(POS(T$,M$,1) + 2) /3
140 REM CHECK IF MONTH IS SPELLED CORRECTLY
150 IF M <> INT (M) GO TO 200
160 PRINT M$" IS MONTH NUMBER"M
170 GO TO 100
200 PRINT "BAD MONTH" GO TO 100
READY

RUNNH
TYPE THE FIRST 3 LETTERS OF A MONTH? NOV
NOV IS MONTH NUMBER 11
TYPE THE FIRST 3 LETTERS OF A MONTH? DEC
DEC IS MONTH NUMBER 12
TYPE THE FIRST 3 LETTERS OF A MONTH? JAN
JAN IS MONTH NUMBER 1
TYPE THE FIRST 3 LETTERS OF A MONTH? AUD
BAD MONTH
TYPE THE FIRST 3 LETTERS OF A MONTH? Ctrl/C

STOP AT LINE 110
READY
BYE
12:39:27 SIZE:14K CPU:0.76
PDS>
```

In this example the user first invokes BASIC by issuing the BASIC command. BASIC indicates that it is ready to accept BASIC program lines and commands by printing READY. The user then retrieves an existing BASIC program by entering the OLD command. The LIST and RUN commands respectively print and execute this program. Since this program is written as a loop (that is, after executing line 200 it loops back to line 100) it executes indefinitely. By entering Ctrl/C, the user terminates the program execution. BASIC then prints the line number where execution stopped. The BYE command terminates the BASIC session.

10 COBOL

10.1 Introduction

COBOL is the acronym for COmmon Business Oriented Language. You perform four steps to produce an executable task from a COBOL source program:

1 Create one or more source files.

2 Compile the source files.

3 Link the compiled (object) modules to form an executable task.

4 Run the task.

This chapter describes how to use IAS commands to perform these steps. You can create the file to compile, link, and run the task in a single batch job. See the following manuals for information about programming in COBOL on the PDP-11:

- *PDP-11 COBOL Language Reference Manual*
- *PDP-11 COBOL User's Guide*

10.2 Creating Source Files

Use the CREATE or EDIT commands to create source files. The EDIT command has the advantage that it enables an interactive user immediate access to all its editing facilities. However, to correct errors made while using the CREATE command, you must rely on keyboard facilities or close the file, then issue the EDIT command.

10.3 The COBOL Command

By default, the COBOL command compiles a source program and produces an object file. For example:

```
PDS> COBOL
FILE? SOURCE.CBL
```

This command string compiles the program SOURCE.CBL and produces an object file named SOURCE.OBJ. If you omit the file type field in the specification of the source file, the COBOL compiler assumes it to be CBL.

10.3.1 Compiling COBOL Source Files

You can only specify one source file with each COBOL command. For example:

- Example 1:

```
PDS> COBOL
FILE? COBSRC
```

- Example 2:

```
$COBOL COBSRC
```

- Example 3:

```
PDS> COBOL COBSRC
```

Each of the command strings above instructs the system to compile the source file specified and to produce compiler output as the defaults dictate.

By default, the compiler performs the following functions:

1 Produces an object file with the name of the source file and OBJ as the file type.

2 Compiles the source file according to the compiler's default switches. See the description of the COBOL command in Chapter 14 for further details.

3 Produces a skeleton .ODL file used during program linking if file processing features that use RMS-11 facilities are used. The *PDP-11 COBOL User's Guide* describes this facility in detail.

10.3.2 COBOL Command Qualifiers

The qualifiers to the COBOL command are:

- /OBJECT[:filespec]
- /NOOBJECT
- /LIST[:filespec]
- /NOLIST
- /SWITCHES:(switches)

The compiler produces an object file unless you specify /NOOBJECT. You can specify a name for the object file after /OBJECT or leave the object file to be named by default. See the description of the COBOL command in Chapter 14 for further details.

You can specify /LIST to obtain a listing. The /LIST :filespec qualifier enables you to store the listing in a file; otherwise, the listing file is printed at the line printer, then deleted.

10.3.3 COBOL Compiler Switches

The PDP-11 COBOL compiler provides switches to enable you to tailor the compilation to particular needs. You specify the switches by means of the /SWITCHES qualifier to the COBOL command. For example:

```
$COB/SWITCHES:(/MAP) SOURCE.CBL
```

The specified switches must be enclosed in parentheses. For example:

```
PDS> COBOL/SWITCHES(/ERR:2/MAP/CVF)/LIST:ACCOUNT.LST
FILE? ACCTS.CBL
```

If you do not specify switches, the default switches are as follows:

```
(/ERR:0/ACC:1/NOMAP)
```

The COBOL command specification in Chapter 14 lists and describes all the compiler switches.

10.3.4 Compiler Error Messages

The compiler generates error messages (diagnostic, warning, and fatal) whenever an error is detected in the source program. Any error detected by the compiler results in the associated message being embedded within the source program listing. The compiler prints the error message either before or after the erroneous source program statement. If the statement in error cannot be identified, the errors are flagged at the end of the listing. See the *PDP-11 COBOL User's Guide* for a detailed description of error messages.

10.4 Linking Object Files

Issue the LINK command to link COBOL object files to create an executable task.

10.4.1 The LINK Command

The LINK command invokes the IAS task builder to build an executable task from object files. You can generate object files directly by using the COBOL command, or you can extract them from the user-written system library files; see Chapter 8. In particular, you must specify the system module libraries COBLIB.OLB and RMSLIB.OLB.

The *IAS Task Builder Reference Manual* contains a complete description of the task builder. The LINK command is described in Chapter 14.

To link one or more COBOL programs using the system default task builder switches and options, issue the LINK command followed by the list of object files to be linked together into an executable task. For example:

```
LINK/OPTIONS PRODUCTS STOCKS [1,1]COBLIB/LI [1,1]RMSLIB/LI
OPTIONS?
```

The above example links together the COBOL object files PRODUCTS.OBJ and STOCKS.OBJ, and routines in COBLIB and RMSLIB.

10.4.1.1 Options

The qualifier /OPTIONS enables you to specify task builder options. In interactive mode, the presence of the qualifier /OPTIONS in the command qualifier list causes the task builder to prompt OPTIONS? after the input files have been specified. For example:

```
PDS> LINK/OPTIONS
FILE? PROG.OBJ,REPORT.OBJ,[1,1]COBLIB/LI,[1,1]RMSLIB/LI
OPTIONS?
```

You then enter the options one line at a time. A slash (/) as the first character in a line then terminates the option input. For example:

```
PDS> LINK/OPTIONS
FILE? MYCOB.OBJ,PROG.OBJ,[1,1]COBLIB/LI,[1,1]RMSLIB/LIB
OPTIONS? UNITS=9
OPTIONS? ASG=DT1:7:8:9
OPTIONS? /
```

In batch mode, the presence of the /OPTIONS qualifier in the command qualifier list causes the task builder to expect one or more options to be specified on one or more lines immediately following the command string. A line containing a slash (/) in the first character position terminates the list of options. For example:

```
$LINK/OPTIONS PROG REPORT [1,1]COBLIB/L [1,1]RMSLIB/L
UNITS=9
ASG=DT1:7:8:9
/
```

The use of the /OPTIONS qualifier also inhibits the task builder from building the task to the system resident library (SYSRES). This library is inappropriate for COBOL programs since the library contains commonly used file control services (FCS) routines, rather than RMS-11 routines.

The task builder options are summarized in a table in the LINK command (Chapter 14).

10.4.1.2 Object Module Libraries

The file qualifier /LIBRARY specifies a library file that contains the user-written object modules to be incorporated in the task.

In addition, you must specify the supplied object module libraries COBLIB.OLB COBOL programs in that order.

```
PDS> LINK/OPTIONS CBLPROG [1,1]COBLIB/LI[1,1]RMSLIB/LI
OPTIONS?
```

The task builder automatically searches system object module libraries for referenced modules.

If the .ODL file generated by the COBOL compiler or a user supplied .ODL file is used for a complex structured program, the library specifications for COBLIB and RMSLIB must be included in the .ODL file. See the *PDP-11 COBOL User's Guide* for further details.

10.4.1.3 Output Files

The task builder does not generate any output files, other than an executable task image, unless you specifically request them by supplying the relevant qualifiers. The possible output files and their associated qualifiers are:

Output File	Qualifier
Task Image file	/TASK[:filespec]
Memory allocation map file	/MAP[:filespec]
Symbol definition file	/SYMBOLS[:filespec]

10.4.1.4 Example

The following example links three object files.

```
PDS> LINK/TASK:WAGES/MAP:WAGES/OPTIONS
FILES? PAY,PEOPLE,MONTH[1,1]COBLIB/LI,[1,1]RMSLIB/LI
OPTIONS? UNITS = 5
OPTIONS? ASG=DT2:1:2,TI:3,MT:4:5
OPTIONS? /)
PDS>
```

The LINK command links the three object files to create a task image file named WAGES.TSK and a map file named WAGES.MAP.

10.5 Running the Task

A COBOL programmer compiles and links a task in separate operations. The programmer then uses the RUN command to execute the task image created by the LINK command. To run a linked COBOL task, issue the RUN command and specify the task image file generated by the LINK command. For example:

- Example 1:

```
PDS> RUN
FILE? WAGES
```

- Example 2:

```
$RUN WAGES
```

Both examples instruct the system to run the task named WAGES.TSK.

11 FORTRAN

11.1 Introduction

FORTRAN is the acronym for the FORmula TRANslation language. You perform four steps to produce an executable task from a FORTRAN source program:

1 Create one or more source files.

2 Compile the source files.

3 Link the compiled (object) modules to form an executable task.

4 Run the task.

This chapter describes how to use IAS commands to perform these steps. See the following manuals for information about programming in FORTRAN IV or FORTRAN IV-PLUS:

- *IAS/RSX, VAX/VMS FORTRAN IV User's Guide*
- *FORTRAN IV-PLUS User's Guide*
- *PDP-11 FORTRAN Language Reference Manual*

11.2 Creating Source Files

You can use the CREATE or EDIT commands to create source files. See Section 8.5. The EDIT command has the advantage that it provides you with immediate access to all its editing facilities. However, to correct errors made while using the CREATE command, you must rely on keyboard facilities or close the file and then issue the EDIT command.

11.3 The FORTRAN Command

The basic function of the FORTRAN command is to compile one or more FORTRAN source programs. Command qualifiers, including compiler switches and options, determine the form of the output to be generated by the compiler. The system manager determines which compiler is to be the default compiler. To use the nondefault compiler, qualify the FORTRAN command with the task name. For example:

- FORTRAN/FOR MYPROG (for the FORTRAN IV compiler)
- FORTRAN/F4P MYPROG (for the FORTRAN IV PLUS compiler)
- FORTRAN MYPROG (for the default FORTRAN compiler)

11.3.1 Compiling Source Files

You can specify only one source file with each FORTRAN command. For example:

- Example 1:

```
PDS> FORTRAN
FILE? INVERT
```

- Example 2:

```
$FORTRAN INVERT
```

- Example 3:

```
PDS> FORTRAN INVERT
```

Each of the command strings above instructs the system to compile the source file specified and to produce compiler output as the defaults dictate.

By default, the compiler does the following:

1 Produces an object file that has the name of the source file and the OBJ as the file type.

2 Compiles the source file according to the compiler's default switches. See the description of the FORTRAN command in Chapter 14.

11.3.2 FORTRAN Command Qualifiers

Command qualifiers, each preceded by a slash (/), immediately follow the command name. For example:

```
PDS> FORTRAN/LIST/OBJECT/SWITCHES:(/CK) SOURCE.FTN
```

You specify command qualifiers to modify the function of the FORTRAN command according to the needs of the program. You can also specify qualifiers to affirm default compiler actions. For instance, the example above specifies /OBJECT even though the FORTRAN command produces an object file by default.

11.3.3 FORTRAN Compiler Switches

FORTRAN compiler switches are listed after the /SWITCHES: qualifier. The list of switches must be enclosed in parentheses. The preceding slash separates each switch from the next within the list. For example:

```
$FORTRAN/SWITCHES:(/CK/CO:7/TR:LINES) PROG1.FTN
```

The switches differ, depending on whether the programmer is using FORTRAN IV or FORTRAN IV-PLUS. Both sets of switches are listed in the specification of the FORTRAN command in Chapter 14.

11.3.4 Examples

The following commands compile a FORTRAN source file:

- Example 1:

```
$FORTRAN/OBJECT/LIST:PRINT RDIN
```

  Compile the source program RDIN.FTN, create an object file name RDIN.OBJ and create a listing file called PRINT.LST.

- Example 2:

```
$FORTRAN RPRT.FTN
```

  Compile the source program RPRT.FTN to create an object file named RPRT.OBJ.

The file specification to the /LIST qualifier need not include a file type. In this case, the system assumes a file type of LST.

11.4 Linking Object Files

The LINK command links FORTRAN object files to create an executable task.

11.4.1 The LINK Command

The LINK command invokes the IAS task builder to build an executable task from object files. You can generate these objects directly by using the FORTRAN command or you can extract them from the user-written system library files. See Section 8.4.

The *IAS Task Builder Reference Manual* contains a complete description of the task builder. The LINK command is described in Chapter 14.

To link one or more FORTRAN programs using the system default task builder switches and options, issue the LINK command followed by the list of object files to be linked together into an executable task. For example:

```
LINK PERFECT NUMBER
```

This links together the FORTRAN object files PERFECT.OBJ and NUMBER.OBJ.

11.4.1.1 Options

The qualifier /OPTIONS enables you to specify task builder options. In interactive mode, the presence of the qualifier /OPTIONS in the command qualifier list causes the task builder to prompt OPTIONS? after the input files have been specified. For example:

```
PDS> LINK/OPTIONS
FILE? PROG.OBJ,REPORT.OBJ
OPTIONS?
```

You then enter the options one line at a time. A slash (/) as the first character in a line terminates the option input. For example:

```
PDS> LINK/OPTIONS
FILE? FORT.OBJ,PROG.OBJ
OPTIONS? ACTFIL=8
OPTIONS? MAXBUF=160
OPTIONS? UNITS=9
OPTIONS? ASG=DT1:7:8:9
OPTIONS? /
```

In batch mode, the presence of the /OPTIONS qualifier in the command qualifier list causes the task builder to expect one or more options to be specified on one or more lines immediately following the command string. A line containing a slash (/) in the first character position terminates the list of options. For example:

```
$LINK/OPTIONS PROG.OBJ,REPORT.OBJ
ACTFIL=8
MAXBUF=160
UNITS=9
ASG=DT1:7:8:9
/
```

See the task builder Options table in the LINK command (Chapter 14) for a summary of the task builder options. The table indicates with an F the options that are relevant to FORTRAN programs.

11.4.1.2 Object Modules

The file qualifier /LIBRARY specifies a library file that contains the user-written object modules to be incorporated in the task. The task builder automatically searches system object module libraries for referenced modules. For example:

```
$LINK (FORT.OLB/LIBRARY:(MOD1,MOD2),FORTRAN.OBJ)
```

11.4.1.3 Output Files

The task builder does not generate any output files other than an executable task image, unless you specifically request them by supplying the relevant qualifiers. The possible output files and the associated qualifiers are as follows:

Output File	Qualifier
Task image file	/TASK[:filespec]
Memory allocation map file	/MAP[:filespec]
Symbol definition file	/SYMBOLS[:filespec]

11.4.1.4 Example

The following example links three object files:

```
PDS> LINK/TASK:CALC/MAP:CALC/OPTIONS
FILES? RDIN.OBJ,PROC1.OBJ,RPRT.OBJ
OPTIONS? UNITS=5
OPTIONS? ASG=DT2:1:2,TI:3,MT:4:5
OPTIONS? /
PDS>
```

The LINK command links the three object files to create a task image file named CALC.TSK and a map file named CALC.MAP.

11.5 Running the Task

A FORTRAN programmer compiles and links a task in separate operations.

To run a linked FORTRAN task, issue the RUN command and specify the task image file generated by the LINK command.

- Example 1:

```
PDS> RUN
FILE? CALC
```

- Example 2:

```
$RUN CALC
```

Both examples instruct the system to run the task named CALC.TSK.

12 MACRO-11

12.1 Introduction

MACRO-11 is an assembly language for use on the PDP-11. Perform the following four steps to produce an executable task from a MACRO-11 source program:

1 Create one or more source files.

2 Assemble the source files.

3 Link the assembled (object) files, to form an executable task.

4 Run the task.

This chapter describes how to use IAS commands to perform these steps. It also introduces the online debugging technique (ODT), a system program that aids in debugging assembled and linked object programs.

See the *PDP-11 MACRO-11 Reference Manual* for information about programming in MACRO-11.

12.2 Creating Source Files

Use the CREATE or EDIT commands to create source files. See Section 8.3. The EDIT command has the advantage that it provides you with immediate access to all its editing facilities. However, to correct errors made while using the CREATE command, you must rely on keyboard facilities or close the file and then issue the EDIT command.

12.3 The MACRO Command

The MACRO command assembles one or more source files containing MACRO-11 statements into a single, relocatable binary object file. Command qualifiers, including assembler switches, determine the output to be generated by the assembler.

12.4 Assembling MACRO-11 Source Files

The following command string assembles the source files LOCATE.MAC and FIND.MAC:

- Example 1:

```
PDS> MAC
FILE? LOCATE+FIND
```

- Example 2:

```
$MACRO LOCATE+FIND
```

Each of the command strings above instructs the system to assemble the source files specified and to produce assembler output as the defaults dictate. Note that the MACRO command requires the source files to be concatenated with a plus sign (+). By default, the assembler produces an object file that has the name of the last source file specified but with OBJ as the filetype.

12.4.1 MACRO-11 Command and File Qualifiers

The qualifiers to the MACRO command are:

- /[NO]LIST[:filespec]
- /[NO]OBJECT[:filespec]
- /SWITCHES:(swlist)
- /[NO]CROSSREFERENCE

The file qualifiers are:

- /LIBRARY
- /PASS:n
- /SWITCHES:(swlist)

Specify file qualifiers immediately after the relevant file specification. For example:

```
$MAC MACLIB.MLB/LIB+TEST
```

The LIBRARY qualifier instructs the assembler to treat MACLIB.MLB as a macro library file.

Specify command and file qualifiers to modify the function of the MACRO command according to the needs of your program. You can also specify qualifiers merely to affirm default assembler actions. See the MACRO command (Chapter 14) for a list of MACRO command qualifiers and defaults.

The specification of the MACRO command in Chapter 14 lists all the possible command and file qualifiers. Consult the *PDP-11 MACRO-11 Reference Manual* for a full description. For example:

```
PDS> MACRO/OBJECT:FINAL
FILE? ROUT.MAC+MAIN.MAC
```

This code assembles the source programs ROUT.MAC and MAIN.MAC to produce an object file named FINAL.OBJ.

12.5 Linking Object Files

The LINK command links MACRO-11 object files to create an executable task. See Section 12.7 for information about debugging linked object programs.

12.5.1 The LINK Command

The LINK command invokes the IAS task builder to build an executable task from object files generated by the FORTRAN or MACRO command and/or from object modules held in user-written and system library files.

The *IAS Task Builder Reference Manual* contains a complete description of the task builder. This section gives information to help you use the LINK command.

To modify the action of the task builder, specify various options. To link one or more MACRO-11 programs with the system default task builder switches and options, issue the LINK command, followed by the list of object files to be linked together into an executable task. For example:

```
$LINK REALTIME ADCONVERT
```

This code links together the object programs REALTIME.OBJ and ADCONVERT.OBJ.

12.5.1.1 Options

The /OPTIONS qualifier enables you to specify task builder options. In interactive mode, the presence of the qualifier /OPTIONS in the command qualifier list causes the task builder to prompt OPTIONS? after the input files have been specified. For example:

```
PDS> LINK/OPTIONS
FILE? PROG.OBJ, REPORT.OBJ
OPTIONS?
```

You then enter the options one line at a time. A slash (/) as the first character in a line terminates the list of options and the tASK bUILDER Resumes executing. For example:

```
PDS> LINK/OPTIONS
FILE? MAIN.OBJ, PROG.OBJ
OPTIONS? TASK=SYSMAN
OPTIONS? UIC=[1,1]
OPTIONS? SGA=SYSRES:RO
OPTIONS? /)
```

In batch mode, if the /OPTIONS qualifier is in the command qualifier list, the task builder expects one or more options to be specified on one or more lines immediately following the command string. You must specify a single option on each line. A card or line containing a slash (/) in the first character position terminates the list of options. For example:

```
$LINK/OPTIONS PROG.OBJ, REPORT.OBJ
TASK=SYSMAN
UIC=[1,1]
SGA=SYSRES:RO
/)
```

Chapter 14 contains a summary of the task builder options in the specification of the LINK command. The summary marks the options relevant to MACRO programs with the letter M.

12.5.1.2 Object Module Libraries

The file qualifier /LIBRARY specifies the library files that contain the user-written object modules to be incorporated in the task. The task builder automatically searches system object module libraries for referenced modules. For example:

```
$LINK MACRO.OLB/LIBRARY:(MAC1,MAC2) MACRO.OBJ
```

12.5.1.3 Output Files

The task builder does not generate any output files, other than an executable task image, unless you specifically request them by supplying the relevant qualifiers. The possible output files and the associated qualifiers are:

Output File	Qualifier
Task image file	/TASK[:filespec]
Memory allocation map file	/MAP[:filespec]
Symbol definition file	/SYMBOLS[:filespec]

12.5.1.4 Example

The following example links three object files to form a task named CALC.TSK.

```
PDS> LINK/TASK:CALC/MAP:CALC/DEBUG/OPTIONS
FILE? (SEG1.OBJ,SEG2.OBJ,MACRO.OBJ)
OPTIONS? UNITS=5
OPTIONS? ASG=DT2:1:2,TI:3,MT:4:5
OPTIONS? /)
PDS>
```

The command string above links the three object files to create a task image file named CALC.TSK and a map file named CALC.MAP. The /DEBUG qualifier instructs the task builder to include a debugging aid (that is, the ODT program), and task builder options assign logical unit numbers.

12.6 Running the Task

A MACRO-11 programmer assembles and links a task in separate operations. The programmer can then use the RUN command to begin execution of the task image created by the LINK command.

When you use it to execute a MACRO-11 task, the RUN command has no qualifiers and only one parameter (the file specification of the task to be run). The file containing the executable task is the task image file generated by LINK. For example:

- Example 1:

```
PDS> RUN)
FILE? CALC.TSK
```

- Example 2:

```
$RUN CALC.TSK
```

Both examples instruct the system to run the task named CALC.TSK.

12.7 Debugging

12.7.1 The Online Debugging Technique

IAS provides the online debugging technique (ODT) to help you debug assembled and linked object programs. To incorporate ODT in the linked program, you specify the /DEBUG qualifier to the LINK command, see Section 12.5.1.1. For example:

```
$LINK/DEBUG MACRO.OBJ
```

The task builder then automatically includes ODT in the task image.

The *IAS ODT Reference Manual* contains a complete description of ODT. In brief, however, you interact with ODT and the object program to perform the following functions:

1 Print the contents of any location for examination or alteration.

2 Run all or any portion of the task using the breakpoint feature.

3 Search the task for specific bit patterns.

4 Search the task for words which reference a specific word.

5 Calculate a block of words or bytes with a designated value.

6 Fill a block of words or bytes with a designated value.

The breakpoint is one of ODTs most useful features. When debugging a program, it is often desirable to allow the program to run normally up to a predetermined point, at which time the contents of various registers or locations can be examined and possibly modified. To accomplish this, ODT acts as a monitor to the user program.

During a debugging session, the current assembly listing and memory allocation map of the program to be dubugged must be available at the terminal. You can make minor corrections to the program online during the debugging session. You can then run the program under control of ODT to verify any changes made. You must note any major corrections, however, (such as a missing subroutine) on the assembly listing and incorporate them in a subsequent updated program assembly.

12.7.2 User-Written Debugging Aids

You can also incorporate a user-written debugging aid in a linked object program. The file containing the debugging aid is specified with the /DEBUG qualifier. For example:

```
PDS> LINK/DEBUG:[1,1]DDT/READ_WRITE/SYMBOLS
FILES? MACRO.OBJ
```

13 CORAL 66

13.1 Introduction

CORAL is the acronym for the Computer On-line Real-time Applications Language. Perform the following four steps to produce an executable task from a CORAL 66 source program:

1 Create one or more source files.

2 Compile the source files.

3 Link the object module(s) to form an executable task.

4 Run the task.

This chapter describes how to use IAS commands to perform these steps. Consult the following manuals for information about programming in CORAL 66:

- *IAS/RSX/VMS CORAL 66 User's Guide*
- *CORAL 66 Language Reference Manual*

13.2 Creating Source Files

You can use the CREATE or EDIT commands to create source files. See Chapter 8, Section 8.5. The EDIT command has the advantage that it provides you with immediate access to all its editing facilities. However, to correct errors made while using the CREATE command, you must rely on keyboard facilities or close the file, then issue the EDIT command.

13.3 The Coral Command

The basic function of the CORAL command is to compile one or more CORAL source programs. Command qualifiers, including compiler switches and options, determine the form of the output that the compiler is to generate.

13.3.1 Compiling Source Files

The following command strings compile the source files INVERT.COR and INVERT1.COR.

- Example 1:

```
PDS> CORAL
FILE? INVERT+INVERT1
```

- Example 2:

```
$CORAL INVERT+INVERT1
```

- Example 3:

```
PDS> CORAL INVERT+INVERT1
```

Each of the command strings above instructs the system to compile the source files specified and to produce compiler output as the defaults dictate.

By default, the compiler performs the following functions:

1 Produces an object file that is given the name of the first source file and OBJ as the filetype.

2 Compiles the source file according to the compiler's default switches. See the CORAL command description in Chapter 14.

13.3.2 CORAL Command Qualifiers

Each command qualifier is preceded by a slash (/) and immediately follows the command name. For example:

```
PDS> CORAL/LIST/OBJECT/SWITCHES:(/BC) SOURCE.COR
```

You specify command qualifiers to modify the function of the CORAL command according to the needs of the program. You can also specify qualifiers merely to affirm default compiler actions. For instance, the example above specifies /OBJECT, even though the CORAL command produces an object file by default.

You enter compiler switches after the /SWITCHES: qualifier. Enclose the list of switches in parentheses. The slash preceding each switch separates each one within the list. For example, in batch mode, switches are specified as follows:

```
$CORAL/SWITCHES:(/BC/OP:2/LI:SRC) PROG1.COR
```

The switches are listed in the description of the CORAL command; see Chapter 14.

13.3.3 Examples

The following examples demonstrate commands that compile a CORAL source file:

- Example 1:

  ```
  $CORAL/OBJECT/LIST:PRINT RDIN
  ```

  Compile the source program RDIN.COR, create an object file name RDIN.OBJ, and create a listing file called PRINT.LST.

- Example 2:

  ```
  $CORAL/OBJECT/LIST:LPROC1 PROC1
  ```

  Compile the source program PROC1.COR, create an object file named PROC1.OBJ, and create a listing file called LPROC1.LST.

- Example 3:

  ```
  $CORAL/OBJECT/LIST:READ RPRT.COR
  ```

  Compile the source program RPRT.COR, create an object file named RPRT.OBJ and create a listing file called READ.LST.

Note that the file specifications to the /LIST qualifier need not include a file type. In this case, the system assumes the file type to be LST.

13.4 Linking Object Files

Issue the LINK command to link CORAL object files to create an executable task.

13.4.1 The LINK Command

The LINK command invokes the IAS task builder to build an executable task from object files. You can generate object files directly by using the CORAL command, or you can extract them from user-written and system library files. See Section 8.5.

The *IAS Task Builder Reference Manual* contains a complete description of the task builder. See Chapter 14 for a description of the LINK command.

To link one or more CORAL programs using the system default task builder switches and options, you issue the LINK command followed by the list of object files to be linked together into an executable task. The following example demonstrates how to link together the CORAL object files PERFECT.OBJ and NUMBER.OBJ.

```
LINK PERFECT NUMBER [11,50]COROTS/LIB
```

13.4.1.1 Options

The qualifier /OPTIONS enables you to specify task builder options. In interactive mode, the presence of the qualifier /OPTIONS in the command qualifier list causes the task builder to prompt OPTIONS? after the input files have been specified. For example:

```
PDS> LINK/OPTIONS
FILE? PROG.OBJ,REPORT.OBJ,[11,50]COROTS/LIB
OPTIONS?
```

You then enter the options one line at a time. If you place a slash (/) as the first character in a line, the option input terminates and the task builder resumes execution. For example:

```
PDS> LINK/OPTIONS
FILE? CORAL.OBJ,PROG.OBJ,[11,50]COROTS/LIB
OPTIONS? ACTFIL=8
OPTIONS? MAXBUF=280
OPTIONS? UNITS=9
OPTIONS? ASG=DT1:7:8:9
OPTIONS? /
```

In batch mode, the presence of the /OPTIONS qualifier in the command qualifier list causes the task builder to expect one or more options to be specified on one or more lines immediately following the command string. A line containing a slash (/) in the first character position terminates the list of options. For example:

```
$LINK/OPTIONS PROG.OBJ,REPORT.OBJ,[11,50]COROTS/LIB
ACTFIL=8
MAXBUF=280
UNITS=9
ASG=DT1:7:8:9
/
```

There is a summary of the task builder options in the LINK command in Chapter 14_. Note that the MAXBUF and FMTBUF options have a special meaning when linking CORAL programs.

13.4.1.2 Object Modules

The file qualifier /LIBRARY specifies a library file that contains the user-written object modules to be incorporated in the task. The task builder automatically searches system object module libraries for referenced modules. For example:

```
$LINK COROTS/LIB,COROTS.OLB/LIBRARY:(MOD1,MOD2),CORAL.OBJ
```

13.4.1.3 Output Files

The task builder does not generate any output files, other than an executable task image, unless you specifically request them by supplying the relevant qualifiers. The possible output files and the associated qualifiers are:

Output File	Qualifier
Task image file	/TASK[:filespec]
Memory allocation map file	/MAP:[filespec]
	or
	/MAP:[filespec/qualifier]
Symbol definition file	/SYMBOLS[:filespec]

The MAP filespec qualifier can be /FILES, /FULL, /NARROW, /SHORT, /WIDE.

13.4.1.4 Example

The following example links three object files.

```
PDS> LINK/TASK:CALC/MAP:CALC/OPTIONS
FILES? RDIN.OBJ,PROC1.OBJ,RPRT.OBJ,[11,50]COROTS/LIB
OPTIONS? UNITS=5
OPTIONS? ASG=DT2:1:2,TI:3,MT:4:5
OPTIONS? /
PDS>
```

The LINK command links the three object files to create a task image file named CALC.TSK and a map file named CALC.MAP.

13.5 Running the Task

A CORAL programmer compiles and links a task in separate operations. The programmer then uses the RUN command to execute the task image created by the LINK command.

To run a linked CORAL task, issue the RUN command; then the task image file generated by the LINK command is run. For example:

- Exaple 1:

```
PDS> RUN
FILE? CALC
```

- Example 2:

```
$RUN CALC
```

Both examples instruct the system to run the task named CALC.TSK.

14 PDS Command Descriptions

14.1 Introduction

This chapter lists and describes all interactive and/or batch commands available to the PDS user.

14.2 PDS Command Format

The general format of a PDS command is:

```
[$]command-name[/quals] [parameter-1][,...,parameter-n]
```

In the description of commands in this manual, the following conventions apply:

1 Brackets

- Square brackets [ ] surround optional values; for example:

```
COPY[/quals]
```

- Round brackets ( ) are part of the command; for example:

```
COBOL/SW:(/MAP)
```

2 Dollar sign [$]

The dollar-sign [$] must appear in position 1 of a command to be executed in batch mode. It can optionally appear in a command executed in interactive mode.

3 Command names

You can abbreviate the command name to the number of characters that uniquely identifies that command. For example, you can abbreviate LOGOUT to LOG.

NOTE: You can abbreviate most commands to three letters.

4 Parameters

A parameter describes either a value that a command is to use when executing, or further defines the action a command is to take. If you are an interactive user, you can supply parameters in response to prompts. (See Section 4.3). Otherwise, at least one space must separate the first parameter from the command name. In this case, parameters are then separated from each other by one or more spaces and/or a single comma (,).

5 Parentheses and ellipses ("...")

Some commands enable you to replace a single parameter with a list of values. When you do this, you can surround the list by parentheses. You do not need parentheses when the parameter you are replacing is the last or only parameter in the command string. The following examples illustrate this.

- Example 1:

```
DELETE (A.DAT;2,B.DAT;1,C.DAT;4
```

In this example, the parentheses are optional.

- Example 2:

```
APPEND (A.DAT B.DAT) C.DAT
```

This command specifies that files A.DAT, and B.DAT are to be added to the end of file C.DAT. You need parentheses because the parameter you are replacing is not the last parameter.

In the description of a command format, ellipses indicate that a list of values of the same type can replace a single value.

6 Qualifiers

You use a qualifier to modify the default action of a command. A qualifier always begins with a slash (/). Both command names and parameters can have associated qualifiers. For example:

```
PRINT/DELETE MYFILE.DAT
CREATE DAT36.DAT/PROTECT:(WO:RWED)
```

Many qualifiers have associated qualifier values. You separate the qualifier from the qualifier value by a colon (:), for example, /KEEP:1. Whenever a qualifier requires a list of values, you must enclose the list in parentheses. For example:

```
/BLOCKS:(m-n)
```

A qualifier must not contain any spaces.

7 Continuation Character (-)

You use a hyphen (-), optionally followed by spaces and/or a comment, to indicate the continuation of a command on the next line. For example:

```
PDS> COPY A.DAT -
>B.DAT
```

NOTE: Following a continuation character, the system reprompts with a ">" on the following line.

8 Comment Character (!)

An exclamation point (!) delimits the start of a comment. Comments can occur only after the last character of a command or after a hyphen. Comments are for your information only and do not affect the processing of the command. For example:

```
PDS> COPY A.DAT B.DAT ! FILE A TO FILE B

MOUNT/DENSITY:800 MT:- ! MOUNT MY

VOLID3 TU10:  ! TAPE ON ANY TU10
```

9 Concatenation Character (+)

A plus sign (+) indicates concatenation; that is, the records in the file specification on the left of the plus sign are processed, followed by the records in the file specification on the right of the plus sign. For example:

```
PDS> MACRO A+B
```

The MACRO-11 statements in file A.MAC followed by the MACRO-11 statements in file B.MAC are read by the MACRO assembler.

14.3 PDS Command Descriptions

The layout of each command description is as follows:

1 Function

This section describes the function of the command.

2 Required Privilege

This section states the privilege requirement(s) you need to issue the command. The system manager assigns privileges.

3 Format

This section supplies the correct command format, as well as a description of the command parameters and qualifiers. If the command format is preceded by a [$], the command is also valid in batch mode.

4 Command Variations

This section details any variation in the use of a command between multiuser and timesharing systems. See Chapter 1 for a definition of these terms. References to multiuser systems also include real-time systems.

5 Technical Notes

This section lists any additional information you need to issue the command (for example, restrictions, default action).

6 Examples

This section supplies working examples and an explanation (if necessary).

14.4 PDS Command Library

The remainder of this chapter contains a description of the following PDS commands.

[$]ABORT
[$]ALLOCATE
[$]APPEND
[$]ASSIGN
[$]BASIC
[$]CANCEL
[$]COBOL
[$]COMPARE
[$]CONTINUE
[$]COPY
[$]CORAL
[$]CREATE
[$]DCL
[$]DEALLOCATE
[$]DEASSIGN
[$]DELETE
[$]DIRECTORY
[$]DISABLE
[$]DISMOUNT
[$]DUMP

[$]EDIT
[$]ENABLE
[$]EOD
[$]EOJ
[$]FIX
[$]FORTRAN
[$]GOTO
[$]HELP
[$]IDENTIFY
[$]INITIALIZE
[$]INSTALL
[$]JOB
[$]LIBRARIAN
[$]LINK
[$]LOGOUT
[$]MACRO
[$]MCR
[$]MERGE
[$]MESSAGE
[$]MOUNT
[$]ON
[$]PRINT
[$]QUEUE
[$]REMOVE
[$]RENAME
[$]RUN
[$]SET
[$]SHOW
[$]SORT
[$]STOP
[$]SUBMIT
[$]TRUNCATE
[$]TYPE
[$]UNFIX
[$]UNLOCK
[$]VERIFY

ABORT

FUNCTION

The ABORT command enables you to abort the execution of a current timesharing or real-time task.

REQUIRED PRIVILEGE

ABORT/TIMESHARING—ANY

ABORT/REALTIME—PR.RTC

FORMAT

PDS> [$]ABORT *[/quals] [taskname] [terminal]*

parameter definitions

/quals

Can be any of the following qualifiers:

Qualifier	Explanation
/TIMESHARING	Aborts the currently active timesharing task, which has been suspended by Ctrl/C. The taskname and terminal parameters cannot be specified.
/[NO]REGISTER_DUMP	Used only with the /TIMESHARING qualifier. It causes the contents of the registers to be displayed when the task is aborted. If a task aborts because of a fault (for example, an odd address) the register contents are always displayed. **Note:** This is the default.
/REALTIME	Aborts a real-time task. The taskname must be specified, but the terminal is optional.

taskname

The installed name of the task to be aborted.

terminal

The terminal where the task to be aborted was activated. Default is the current terminal.

COMMAND VARIATIONS

The qualifier /[NO]REGISTER_DUMP is not available on multiuser systems.

EXAMPLES

- **Example 1:**

```
PDS> RUN MYPROG
Ctrl/C
TASK SUSPENDED
PDS> ABORT
09:25:26 SIZE:4K CPU:0.03
```

- **Example 2:**

```
PDS> SET QUIET

PDS> RUN MYPROG
Ctrl/C
TASK SUSPENDED
PDS> ABORT/TIMESHARING

PDS>
```

- **Example 3:**

```
PDS> RUN [270,273]QAMAR1
Ctrl/C
TASK SUSPENDED

PDS> ABO/REG
TASK ABORTED
PS = 174000
PC = 001366
RO = 001000
R1 = 000000
R2 = 000000
R3 = 140551
R4 = 155300
R5 = 000000
SP = 001000
09:25:26 Size: 4K CPU: 0.04

PDS>
```

- **Example 4:**

```
PDS> ABORT/REALTIME RTTSK
```

- **Example 5:**

```
PDS> ABORT/REALTIME MYTSK TT6
```

ALLOCATE

FUNCTION

The ALLOCATE command allocates a specified device to you and optionally associates a logical name with the device.

REQUIRED PRIVILEGE

PR.DEV

FORMAT

PDS> [$]ALLOCATE *[/DEVICE]*

DEVICE? *devicename* ESC

[LOGICAL NAME? *logicalname]*

parameter definitions

/DEVICE
Allocates a device. This is the default.

devicename
Specification of the device to be allocated to the user.

logicalname
Locigal name to be associated with the device. The logical name is in the form xymn, where x and y are alphabetic characters and m and n are octal digits.

COMMAND VARIATIONS

On a multiuser system, the ALLOCATE command is illegal.

TECHNICAL NOTES

You have exclusive access to the device until either you deallocate the device or the system deallocates the device. The system automatically deallocates a device when you dismount the device or deassign the last LUN to which the device is assigned, unless you modify the DISMOUNT or DEASSIGN command with the qualifier /KEEP.

You cannot explicitly allocate a system device (that is, a device allocated to all users by the system manager). If *devicename* does not include a unit number, the system allocates any available device of the specified type and, in interactive mode, prints at the user's terminal the physical unit allocated. In batch mode if no explicit unit number is specified, you must define a logical name in order to refer to that device in subsequent commands. The SHOW DEVICES command can be issued for a list of available devices.

EXAMPLES

- Example 1:

```
PDS> ALLOCATE MM [ESC]
LOGICAL NAME? NT0:
MM0: ALLOCATED
PDS> MOUNT NT0: VOL75
PDS> DISMOUNT/KEEP NT0
PDS> MOUNT NT0: VOL75
PDS> DISMOUNT NT0:
```

  On completion of this command, the volume is dismounted and the device deallocated.

- Example 2:

```
PDS> ALLOCATE/DEVICE
DEVICE? DK0:

PDS>
```

- Example 3:

```
$ALLOC MT: LM0:
```

APPEND

FUNCTION

The APPEND command adds records from one or more input files, to the end of an existing file.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> [$]APPEND

FROM? *[()]infile[/quals] [,....infilen][/quals][)]*

TO? *outfile*

parameter definitions

infile1,infilen

Input file specifications. If you specify more than one file, you must enclose the list in parentheses.

/quals

Any of the following:

Qualifier	Explanation
/SEQUENTIAL	Input file is sequential. This is the default.
/INDEXED [/KEY:NUMBER:n]	Input file is an indexed sequential (ISAM) file. You can specify the order records are appended by using the /KEY:NUMBER qualifier. If you specify /KEY:NUMBER, you can omit /INDEXED. The default is /KEY:NUMBER:1 (the primary key).
/RELATIVE	Input file structure is relative.

outfile

Output file where the input files are to be appended.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

If one or more files in a list of input files is in error, the system ignores the files in error and appends the rest to the output file. All file specifications must include a file name and a file type. If you do not supply a version number, the system assumes the highest version number for the input file and the highest version plus 1 for the output file.

If one of the files you have specified is to be input from your terminal (TI:), the system transfers everything you type in after the completed command string. The transfer continues until you type Ctrl/Z to terminate the input file.

EXAMPLES

- Example 1:

```
PDS> APPEND (A.OBJ B.OBJ) C.OBJ
```

- Example 2:

```
PDS> APPEND
FILE? (ABC.FTN DEF.FTN)
TO? XYZ.FTN
```

- Example 3:

```
PDS> APPEND TWO.MAC,ONE.MAC
```

- Example 4:

```
$APPEND (ABC.DAT,DEF.DAT),XYZ.DAT
```

- Example 5:

```
PDS> APPEND ADDIT.DAT/KEY:NUM:3 OLDONE.DAT
```

  Appends all records from the ISAM file ADDIT.DAT to OLDONE.DAT in an order determined by key number 3 (the second alternate key field).

- Example 6:

```
PDS> APPEND (FILE1.TXT,[200,40]*.TXT) UPDATED.TXT
```

  Appends text file FILE1.TXT and all .TXT files in [200,40] to UPDATED.TXT in the current UFD.

- **Example 7:**

```
PDS> APPEND TI: BATCHJOB.BIS
$DIR [200,200]
$EOJ
[Ctrl/Z]
PDS>
```

Appends the commands $DIR and $EOJ to the end of the BATCHJOB.BIS file.

ASSIGN

FUNCTION

The ASSIGN command assigns a logical unit number (LUN) to a device.

REQUIRED PRIVILEGE

ASSIGN device—PR.RUN ASSIGN task—PR.RTC

FORMAT

PDS> [$]ASSIGN *[/qual] devicename lun*

parameter definitions

/qual = /TASK:taskname

Installed name of the task for which the installed LUN assignment is to be changed.

devicename

Specification of the device to be assigned to the logical unit. The device must be one mounted for you by the MOUNT command, or one to which all users have access.

lun

Logical unit number.

COMMAND VARIATIONS

ASSIGN devicename lun is not available on a multiuser system.

TECHNICAL NOTES

The ASSIGN devicename lun command assigns a LUN only to the device specified for timesharing tasks run from your terminal using the RUN command. The ASSIGN/TASK command reassigns LUNs for the installed task named in the command. The reassignment remains in effect until you remove the task or issue another ASSIGN/TASK command.

You can assign LUNs as follows:

1 By means of the ASSIGN command before a task is run.

2 By means of a task builder option when a task is linked. See the *IAS task builder Reference Manual.*

3 From within a program by means of the system directive ALUN$ or OPEN$ or the FORTRAN subroutines ASSIGN and ASNLUN. See the *IAS System Directives Reference Manual.*

If the ASSIGN command associates a device name with a LUN, that assignment overrides any made for that LUN by the task builder. If any executing program assigns a logical unit, that assignment overrides the action of any ASSIGN command for that LUN. The system automatically deassigns any LUNs when the device is dismounted or deallocated. You can also issue the DEASSIGN command to deassign a device from a LUN.

EXAMPLES

- Example 1:

```
$ASSIGN DP0: 7
```

- Example 2:

```
PDS> ASSIGN
DEVICE? LP0:
LUN? 6
```

- Example 3:

```
PDS> ASSIGN DK2:
LUN? 5
```

- Example 4:

```
PDS> SHO LUN CAROL
**** CAROL
SY0 1,2,3,4
TI0 5
CL0 6

PDS> ASSI/TASK:CAROL
DEVICE? TI:
LUN? 2

PDS> SHO LUN CAROL
**** CAROL
SY0 1,3,4
TI0 2,5
CL0 6
```

BASIC

FUNCTION

The BASIC command invokes a BASIC language processor.

REQUIRED PRIVILEGE

PR.BAS

FORMAT

PDS> [$]BASIC *[/qual]*

parameter definitions

/qual

One of the following:

Qualifier	Explanation
/B11	Invoke the BASIC-11 interpreter. Applicable to systems that have both BASIC-11 and BASIC-PLUS-2.
/BP2	Invoke the BASIC-PLUS-2 compiler. Applicable to systems that have both BASIC-11 and BASIC-PLUS-2.

If you do not specify a qualifier, the installation's default BASIC processor is invoked.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

The following description relates to the BASIC-11 interpreter only. For details of BASIC-11, see the *IAS/RSX BASIC User's Guide* and the *BASIC-11 Language Reference Manual.*

For details of BASIC-PLUS-2, see the *BASIC-PLUS-2/RSX-11M/ VMS User's Guide* and the *PDP-11 BASIC-PLUS-2 Language Reference Manual.*

When the BASIC command is issued, BASIC indicates that the interpreter is ready to receive a command or program line by displaying the following prompt:

```
READY
```

To terminate a BASIC session and return control to PDS, type BYE on a new line. The system then prints information about the session and prompts for further PDS commands. For example:

```
BYE
15:57:32 SIZE: 10K CPU: 3:09
PDS>
```

When the BASIC interpreter is executing a program, typing Ctrl/C causes the system to stop executing after the current line. The terminal displays the number of the last line executed, you can then issue further BASIC commands.

Typing Ctrl/C during the execution of a BASIC LIST or SAVE command or an immediate mode statement stops the execution of those commands or statements. It has no effect on the execution of other BASIC commands.

EXAMPLES

```
PDS> BASIC

READY
BYE
15:15:21 SIZE: 10K CPU: 3:09
PDS>
```

CANCEL

FUNCTION

The CANCEL command cancels the periodic scheduling of requests for a real-time task.

REQUIRED PRIVILEGE

PR.RTC

FORMAT

PDS> [$]CANCEL

TASK? *taskname*

[TERMINAL? *terminal]*

parameter definitions

taskname
Installed name of the task whose scheduled requests are being canceled.

terminal
Terminal where the task to be canceled was activated. Default is the user's terminal.

COMMAND VARIATIONS

Not applicable.

EXAMPLES

- **Example 1:**

```
PDS> CANCEL XKE2
```

- **Example 2:**

```
PDS> CAN MYTS TT4
```

COBOL

FUNCTION

The COBOL command compiles a COBOL source program.

REQUIRED PRIVILEGE

PR.COB

FORMAT

PDS> [$]COBOL *[/quals]*

FILE? *filespec*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/OBLJECT[:filespec]	Produces an object file, named according to filespec if it is supplied (no wildcards allowed). The default file type is .OBJ. /OBJECT is the default qualifier.
/NOOBJECT	Produces no object file.
/LIST[:filespec]	Produces a listing file named according to filespec if it is supplied (no wildcards allowed). The default filetype is .LST.
/NOLIST	Produces no listing file (the default condition).
/SWITCHES:(switches)	Applies the specified COBOL switches. Refer to Table 14–1.

filespec

Specification of the file containing the COBOL source program. The specification must contain a file name. If you omit this file type, the system assumes it is .CBL.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

Defaults

Object File—If you specify the qualifier /OBJECT without a file specification, the object file is given the name of the source file, and the file type .OBJ. The default is /OBJECT.

Listing File—If you supply /LIST without a file specification, the listing file is sent to the line printer. The system default is /NOLIST.

COBOL V4.1

The following new switches are available on COBOL V4.1: /DE, /FLG:kk, /-SKL, /SUB, and /TMP:Dev. /-SKL replaces the old switch /ODL, which is no longer available on COBOL V4.1.

COBOL Switches

The COBOL command can include compiler switches that permit you to tailor the compilation to meet particular needs. Table 14–1 lists the switches and their meaning:

Table 14–1 COBOL Switches

Switch	Default	Meaning
/HELP		Displays information on your terminal on how to use the compiler switches.
/ERR:n	/ERR:0	Suppresses the printing of diagnostics with a severity number less than n; n must be in the range 0 to 2. where: 0 = Informational diagnostics 1 = Warning diagnostics 2 = Fatal diagnostics The switch cannot suppress severity 2 (fatal) diagnostics. An entry of 2 suppresses the printing of all severity numbers less than 2.
/ACC:n	/ACC:1	Produces an object program only if the source program contains diagnostics with severities equal to or less than n; n must be in the range 0 to 2.

Table 14–1 (Cont.) COBOL Switches

Switch	Default	Meaning
/MAP		Produces special map listings of: Data Division Procedure Map External Subprograms Referenced Data and Control PSECTs OTS Routines Referenced Segmentation Map
NL		Instructs the compiler not to list the source statements copied from a library file. The resultant source listing contains only the COPY statement.
/CVF		The source program is in conventional format; that is, 80-character source lines with Area A beginning in character position 8. The default is that area A begins at position 1.
/CREF		Includes a cross-reference listing as a part of the listing file output. When you specify /CREF, data names, procedure names, and source line, numbers are sorted into ascending order and appended to the end of the compilation listing. Use the symbol # to indicate lines that contain the lines with the definition of the reference name.
/CSEG:nnnn		Specifies the maximum size of procedural code PSECT that the compiler is to produce, where nnnn is the maximum size in decimal bytes. The minimum value of nnnn is 100.
/KER:kk		Instructs the compiler to generate PSECT names using the two-character kernel specified by kk to make them unique to this compilation. kk is a two-character string that can contain the numbers 0 to 9 and the letters A to Z.
/OBJ		Prints the object location where the code for each verb of the program is located. The information is listed on the line preceding the source statement it describes.
/ODL		Generates an ODL file (default condition). To override the default condition, enter /-ODL. Note that /ODL is no longer valid on COBOL V4.1 but has been replaced by /-SKL.
/OV		Overlays all procedural PSECTs (segments). Consequently, the root or main program contains no procedural statements.
/PFM:nn	/PFM:10	Defines the maximum number of nested perform statements in the program being compiled. If you specify this, the compiler generates a nested PERFORM stack equal in depth to the decimal number specified by nn. The default nested perform size is 10. It is advantageous to use this switch to adjust the nested PERFORM stack size to the exact number required. This assures maximum use of memory (in that only the exact amount of PERFORM stack space is generated).
/PLT		Automatically pools literals to minimize the memory required to store them (default condiitons). However, pooling literals slows down compiler execution speed. To bypass literal pooling, for increased compiler speed, enter /-PLT.
/RO		Generates read-only PSECTs for the Procedure Division object modules.
/SYM:n		Obtains more symbol table space for the compilation. n (an integer in the range of 1 to 4) specifies the space required for the maximum number of data names and procedure names allowed in the compilation. See below for the correspondence between the integer specified by n, and the number of data names and procedure names assigned. /SYM:n Switch Values

Table 14–1 (Cont.) COBOL Switches

Switch	Default	Meaning

n	Max. Data Names	Max. Procedure Names
1	761	761 (default)
2	1021	1021
3	1531	1531
4	2039	2039

/-BOU **/BOU**

Suppresses bounds-checking for subscripts and indexes. If you specify this switch, the compiler does not generate code to check subscript and index values for acceptable ranges as defined by the OCCURS clause.

Suppression of bounds-checking can increase execution speed for a program that executes a large number of subscripted or indexed data references.

NOTE: The results are unpredictable if a program uses out-of-range subscripts or indexes with the /-BOU switch.

/CM6 **/-CM6**

Causes the compiler to interpret COMPUTATIONAL usage as it did in releases before Version 4.0. The effect is the same as changing all COMPUTATIONAL references to COMPUTATIONAL-6.

NOTE: COMPUTATIONAL-6 is a data format intended only for use in program conversion from PDP-11 COBOL releases prior to Version 4.0. Therefore, it should be considered for temporary use only.

/DE

Creates offset entries (used by CID, the COBOL Interactive Debugger) in the Data Map for all program data-names. Otherwise, only data-names referenced in the Procedure Division are listed with offset entries. /DE also creates symbol table files (DBG) that are used by Symbolic CID. /DE allows CID to DEPOSIT into index names and index data-names.

Table 14–1 (Cont.) COBOL Switches

Switch	Default	Meaning
/FLG:kk	/-FLG:	Causes the compiler to issue informational diagnostics for all COBOL syntax in this program that is not in your selected (kk) level of Federal Standard COBOL. Federal Information Processing Standards Publication 21-1 (FIPS PUB 21-1) announced the adoption of American National Standard COBOL, X3.23-1974, as the Federal COBOL Standard. This publication identifies the following four levels of Federal Standard COBOL: Low, Low-Intermediate, High-Intermediate, and High. Use the kk values shown below to monitor your programming and system development effort. This switch helps protect your investment by informing you of the language level you are using, thereby promoting a high degree of program inter-changeability for use on a variety of computer systems.

kk		Informational Diagnostics
LO	1.	Identifies all source lines containing Digital COBOL language extentions with diagnostic message: 1142 DEC EXTENSION
	2.	Identifies all source lines containing COBOL syntax supported by this compiler in each of the three higher language levels: a. Diagnostic message for Low-Intermediate Language Level syntax requiring a Low-Intermediate COBOL compiler: 1143 LOW INTERMEDIATE REQUIRED FOR THIS CONSTRUCT b. Diagnostic message for High-Intermediate Language Level syntax requiring a High-Intermediate COBOL compiler: 1144 HIGH INTERMEDIATE REQUIRED FOR THIS CONSTRUCT c. Diagnostic message for High Language Level syntax requiring a High Level COBOL compiler: 1145 HIGH LEVEL REQUIRED FOR THIS CONSTRUCT
LI	1.	Identifies all source lines containing Digital COBOL language extensions with diagnostic message: 1142 DEC EXTENSION
	2.	Identifies all source lines containing COBOL syntax supported by this compiler in each of the two higher language levels: a. Diagnostic message for High-Intermediate Language Level syntax requiring a High-Intermediate COBOL compiler: 1144 HIGH INTERMEDIATE REQUIRED FOR THIS CONSTRUCT b. Diagnostic message for High Language Level syntax requiring a High Level COBOL compiler: 1145 HIGH LEVEL REQUIRED FOR THIS CONSTRUCT

For more information, refer to American National Standard Programming Language COBOL, X3.23-1974 and Federal Information Processing Standards (FIPS) Publication 21-1.

Table 14–1 (Cont.) COBOL Switches

Switch	Default	Meaning
/-SKL	/SKL	SUPPRESSES the generation of an SKL file. The default (/SKL) causes the compiler to generate an SKL each time it creates an object file.
/SUB		Identifies a subprogram. The compiler expects subprograms to contain: PROCEDURE DIVISION USING [data-name-1,...data-name-n] If your subprogram does not have input arguments, you can use the /SUB switch instead of the USING phrase.
/TMP:DV		Specifies a 1- to 15-character device name (DV) to contain all temporary files created by the compiler.

For further information on programming in COBOL, see the *PDP-11 COBOL Language Reference Manual*.

EXAMPLES

- Example 1:

```
PDS> COBOL COBPROG.CBL
```

  This example compiles COBPROG and produces an object module COBPROG.OBJ.

- Example 2:

```
PDS> COBOL/SWITCHES:(/MAP)/LIST
FILE? COBPROG
```

  This example compiles COBPROG, produces an object module COBPROG.OBJ, and prints the listing file on the line printer, as well as producing a special map listing.

COMPARE

FUNCTION

The COMPARE command enables you to compare two files line by line with one another and to produce any of the following output:

1 A listing of the differences found.

2 A listing of one file with the differences flagged.

3 A SLIPER file that converts one file to the other.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> [$]COMPARE *[/quals]*

OLD FILE? *oldfile*

NEW FILE? *newfile*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/[NO]OUTPUT[:filespec]	Outputs all the differences found to the line printer. If you give a file specification, the output is directed to the specified file. If you specify /NOOUTPUT, the system prints only the number of differences found. Default is /OUTPUT.
/CHANGE_BARS[:n]	n is the decimal character code to be used. The system prints *newfile* with those lines that differ from *oldfile*, marked by the specified character. For example, 124 for vertical bar (octal 174). Default is decimal code 33 for exclamation point (!) (octal 41).

Qualifier	Explanation
/[NO]COMMENT	Includes all comments in the file comparison. Comments are defined as all characters on any source line preceded by a semicolon (;). If you specify /NOCOMMENT, all comments are ignored. Default is /COMMENT.
/[NO]FORM_FEEDS	Includes all form feeds in the file comparison. If you specify /NOFORM_FEEDS, records that contain only a form feed are ignored. Default is /NOFORM_FEEDS.
/LINES:n	Specifies the number of lines that determine a match. This match means that n successive lines in each input file have been found to be identical. When a match is found, all differences that occur before the match are output. In addition, the first line of the current match is output after the differences to aid in locating the place within each file at which the differences occurred. Default is 3 lines.
/[NO]NUMBERS	Specifies that output file lines be preceded by their line number. Line numbers are incremented by one for each record read, including blank lines. Default is /NUMBERS.
/[NO]MULTIPLE_BLANKS	Includes all multiple blanks (that is, spaces and tabs) in the file comparison. If you specify /NOMULTIPLE_BLANKS, all multiple blanks are treated as a single space. Default is /MULTIPLE_BLANKS.
/[NO]TRAILING_BLANKS	Includes all trailing blanks in the file comparison. If you specify /NOTRAILING_BLANKS, all trailing blanks are ignored. Default is /TRAILING_BLANKS.
/[NO]SLIPER	Outputs a SLIPER file (specified by /OUTPUT) that converts *oldfile* to *newfile*. Default is /NOSLIPER.
/[NO]BLANK_LINES	Includes all blank lines in the file comparison. If you specify /NOBLANK_LINES, all blank lines are ignored. Default is /NOBLANK_LINES.

oldfile
The file to be used in the file comparison.

newfile
The new file to be compared with the old file.

The default command is as follows:

```
COMPARE/OUTPUT/COMMENT/NOROFM_FEEDS/LINES:3/MULTIPLE_BLANKS
    /TRAILING_BLANKS/NOBLANKS/LINE_NUMBERS
```

COMMAND VARIATIONS

Not applicable.

EXAMPLES

- Example 1:

```
PDS> COMPARE/NOOUTPUT/NOFORM_FEEDS/NOMULT
OLD FILE? MKX03.MAC;1
NEW FILE? MKX03.MAC
```

  This example compares MKX03.MAC;1 and MKX03.MAC (latest version).) Form feeds and multiple blanks are ignored. Only the number of differences found displays at your terminal.

- Example 2:

```
PDS> COMPAR/SLIP/OUTPUT:CONVRT.CMD BCPLIO.MAC;1
NEW FILE? BCPLIO.MAC
```

  This example produces a SLIPER command file, CONVRT.CMD. When you use that command file as input to SLIPER, it makes BCPLIO.MAC;1 identical to the latest version.

CONTINUE

FUNCTION

The CONTINUE command restarts execution of a previously interrupted task.

REQUIRED PRIVILEGE

CONTINUE/TIMESHARING—ANY
CONTINUE/DEBUG—ANY
CONTINUE/MESSAGE—PR.RTC
CONTINUE/REALTIME—PR.RTC

FORMAT

Timesharing Format:

PDS> [$]CONTINUE[/TIMESHARING]

Debug Format:

PDS> CONTINUE/DEBUG

Message Output Format:

PDS> CONTINUE/MESSAGE

TASK? *taskname*

[TERMINAL? *terminal]*

Real-Time Format:

PDS> CONTINUE/REALTIME

TASK? *taskname*

[TERMINAL? *terminal]*

parameter definitions

taskname

For message output format, installed name of the task to be continued after being suspended by the suspend form of message output (that is, MO....). For real-time format, installed name of the task being resumed after previously being suspended by the SUSPEND or STOP directive.

terminal

Terminal where the task to be resumed was activated. Default is the user's terminal.

COMMAND VARIATIONS

CONTINUE/DEBUG is illegal on a multiuser system.

TECHNICAL NOTES

Timesharing

On a timesharing system, enter the CONTINUE[/TIMESHARING] command after the task has been suspended by typing Ctrl/C. Typing CONTINUE reactivates the currently suspended task.

On a multiuser system, although a task is not actually suspended, type CONTINUE so that PDS can complete task termination processing for the task last activated.

Indirect or Batch Command File

In an indirect or batch command file, the CONTINUE[/TIMESHARING] command has no effect other than proceeding to the next command. In this context, you can use the command in conditional statements (for example, ON ERROR CONTINUE).

Debug

CONTINUE/DEBUG enables you to continue a task after typing Ctrl/C and to determine the task's current memory location. This is useful when you debug a program.

To use CONTINUE/DEBUG, you must first link the task to the on-line debugging technique (ODT). Otherwise, the task aborts. ODT is described in the *IAS ODT Reference Manual*.

The current memory location is displayed in the following form:

```
TE:nnnnnn
```

where nnnnnn is an octal number.

Message Output

The CONTINUE/MESSAGE command continues the execution of a task when the task has specified the suspend option when it uses the message handler after a message is output. See the *IAS Device Handlers Reference Manual*.

Real Time

Use the CONTINUE/REALTIME command to continue the execution of a task that has been suspended by means of the SUSPEND (SPND$) or STOP (STOP$) directive.

EXAMPLES

- Timesharing example:

```
TASK SUSPENDED
PDS> CONTINUE/TIMESHARING
```

- Debug example:

```
Ctrl/C
TASK SUSPENDED
PDS> CONTINUE/DEBUG
TE:001542
```

- Real-time example:

```
PDS>CONT/REALTIME XKEE3 TT2
```

COPY

FUNCTION

The COPY command enables you to copy the contents of a file to another file. See the Technical Notes for a description of the COPY operations.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> [$]COPY *[/quals]*

FROM? *infile[/filequals]*

TO? *outfile[/filequals]*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/ALLOCATION:n	Allocates n blocks to the output file. The maximum value of n is 32767 blocks.
/ASCII[:n]	Formatted ASCII (for foreign files)
	The transfer is performed as formatted ASCII. Formatted ASCII is defined as ASCII data records terminated by a carriage return or a form feed.
	If you specify n, fixed length record s of size n are generated. Output record is padded with nulls, if necessary.
	If you do not specify n, then variable length records are generated. The output record size equals the input record size.
	ASCII data is transferred as 7-bit quantities. The eighth bit of each byte is masked off before transfer. Ctrl/Z (ASCII 032(8)) is treated as logical end of input file for formatted ASCII transfers from DOS-11 cassette to Files-11. [NULLs (ASCII 000(8)), RUBOUTs (ASCII 177(8)) and Vertical Tabs (ASCII 013(8)) are ignored.

Qualifier	Explanation
/BINARY[:n]	Formatted Binary (for Foreign files) The output file is to be formatted binary. If you specify n, it indicates the fixed length size of each record in bytes (512 bytes is the maximum). The command pads records with nulls to create the specified length. If you do not specify n, standard DOS and RT-11 formatted binary records are produced.
/BLOCK_SIZE:n	Specifies the block size for tape output. n = the block size in bytes. If you do not specify /BLOCK_SIZE, a block size of 128 is assumed. To specify the block size for other media the qualifier is applied to the appropriate file specification (see below).
/CONCATENATE	Concatenates a new file from two or more existing files (these must be separated by a plus sign (+)). This qualifier does not apply to foreign files. The default depends on the input file specification (see Technical Notes). You cannot use this qualifier with /DATE[:KEEP].
/[NO]CONTIGUOUS	Makes the output file contiguous. Note that this qualifier has no effect when copying from magnetic tape.
/CREATE_DIRECTORY	Creates a User File Directory (UFD) entry on the output device for each input file UFD, unless the UFD already exists on the output device.
/DATE[:KEEP] /NODATE	Date of the input file, rather than the date of transfer. :KEEP is the only valid value and you can omit it. You cannot use this qualifier with /CONCATENATE. The default is /NODATE.
/IMAGE[:n]	Image mode (for foreign files) The output file is to be in image mode. Image mode forces fixed length records. If you specify n, it indicates the desired record length (512 bytes is the maximum). The default value of n is 512.
/OWN	Makes the destination UFD the owner of the copy or copies. Not applicable to foreign files. See the file qualifier description, below. See also Technical Notes.
/REPLACE	Replaces the existing output file (if any).
/REWIND[:ERASE] /NOREWIND	Rewinds the magnetic tape and optionally erases it starting the file transfer. You cannot specify this qualifier for DECtapes. For further information see the Technical Notes.
/[NO]SPAN_BLOCKS	Enables you to control whether records copied from disk to magtape (or vice versa) cross block boundaries. If you specify /NOSPAN_BLOCKS, records do not cross block boundaries. If you omit it, files are copied with records possibly crossing block boundaries.
/UPDATE	Updates an existing file, destroys existing data in the output file, and replaces it with the input file data. This qualifier does not delete the existing file before rewriting the data, and the file identification number (FILE-ID) remains the same. This qualifier has no default. For the difference between /UPDATE and /CONCATENATE see the Technical Notes.
/VERIFY	Valid only with a CT output file specification. This qualifier causes each record written to the to be read and verified.

infile

Input file specification. Concatenated files are linked by a plus sign (+).

/filequals

One of the following:

Qualfier	Explanation
/DENSITY:n	Magnetic tape density. :n can have the value 800 or 1600. If you do not specify /DENSITY:n the magnetic tape density defaults to 800. If you specify /DENSITY:n with a nonmagnetic tape device, the density is ignored. This file qualifier is only available for foreign volumes. (See Technical Notes for further details).
/DOS	File is in DOS format.
/RT11	File is in RT11 format.
/SEQUENTIAL	File is a sequential file.
/[NO]SHARED	(Input file only) Enables shared reading of a file that is already open for writing by another user or task if /SHARED is specified. The default for this qualifier is /NOSHARED. (See the Technical Notes for further details.)
/INDEXED[/KEY:NUMBER:n]	(Input file only) The single input file is an indexed sequential (ISAM) file. The records from the indexed input file are to be copied in the order determined by key number n (n>0) to create a new sequential file. If you specify /INDEXED but not /KEY, the records are copied in the order determined by key number 1 (the primary key).
/RELATIVE	(Input file only) Single input file has RELATIVE organization.
/BLOCK_SIZE:n	Defines the blocksize for magnetic tapes. Using this qualifier enables larger blocks to be written onto magnetic tape, thereby saving space taken up by inter-record gaps.

outfile

Output file specification.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

The COPY command enables you to perform the following tasks:

1 Copy a sequential file to another sequential file.

2 Copy a group of sequential files to another group of sequential files (using wildcards).

3 Concatenate of a number of sequential files to a single sequential file.

4 Copy records from a single indexed or relative file to a single sequential file. To copy an indexed or relative file to a file of the same kind, see the MERGE command.

If either infile or outfile has a file name, it must also have a file type.

If you omit the version number from the input file, the highest version number is used. If you omit the version number from the output file, the highest version number plus one is used. If the file does not already exist in the destination UFD, the system assumes version number 1.

The default for the /REWIND[:ERASE] qualifier is /NOREWIND if both file specifications are Files-11, and /REWIND if one of the file specifications is foreign. You can specify this qualifier only if the infiles and outfiles are DOS or Files-11 (magtape). If you specify /REWIND[:ERASE] with a nonmagnetic tape file, it is ignored. You can specify the :ERASE option only for Files-11 (ANSI) magtape.

You can use wildcards whenever an input file specification does not describe concatenated files. If any of the file name, file type or version fields of the output file contain a wildcard, all fields must be wild; you can, however, omit the version field.

If you specify a wild version number in an output file specification, the version numbers for that file are preserved. If you do not specify a file name, the system assumes wildcards (that is, *.*;*).

If the input or output file specification is qualified by /DOS or /RT11, and the output file specification has a wild UFD (*,*), each of the output files is put into a UFD equivalent to that of the input file.

If /DOS or /RT11 modifies either file specification then you can not concatenate the input files, and the output file name and file type must be wild (that is, you can not rename foreign files.)

When you use the /SHARED qualifier, there is no guarantee that any information you obtain is correct. This is because the end-of-file pointer might be incorrect when you open the file.

If you enter infile from the terminal (TI:), the system transfers to the output file all that you type in after the completed command string. The transfer continues until you type Ctrl/Z to terminate the input file.

If either infile or outfile is not in Files-11 format, you must modify its specification by using either /DOS or /RT11. The system does not accept any other foreign formats.

If you specify /DENSITY:n for either the input or output file specification, without qualifying them by /DOS or /RT11, the input file is assumed to be Files-11 and the output file is assumed to be DOS format.

Because of the unused space at the end of blocks, if a file is copied from disk to magnetic tape it occupies more blocks on the tape than it did on the disk. Furthermore, when the file is copied from magnetic tape back to disk, the resulting disk file is also longer than the original disk file.

The /UPDATE qualifier opens an existing file and writes in new data from the beginning. The /CONCATENATE qualifier creates a new file from two or more existing files.

The COPY command is not the best method of making a replica of an indexed sequential file. A better method is to use the CREATE command. This creates a new, empty file with the same structure as the original file. You then use the MERGE command to merge the original file into the new file.

When you use the COPY/CREATE_DIRECTORY, the output filespec must have a UFD of [*,*] or the new UFD is not created.

EXAMPLES

- **Example 1:**

```
PDS> COPY A.CBL B.CBL
```

- **Example 2:**

```
$COPY E.TXT,F.TXT
```

- **Example 3:**

```
PDS> COPY
FROM? E.TXT
TO? F.TXT
```

- **Example 4:**

```
PDS> COPY/OWN DK0:[*,*]*.*
TO? DK1:[*,*]*.*
```

- **Example 5:**

```
PDS> COPY DATA.DAT DT0:*.*
```

- **Example 6:**

```
PDS> COPY/IMAGE:100/VERIFY DT2:*.*/DOS CT:
```

- **Example 7:**

```
PDS> COPY INDEXED.DAT/IND/KEY:NUM:2 SEQUEN.DAT
```

- **Example 8:**

```
PDS> COPY/CREATE_DIRECTORY DK0:[100,1]*.MAC DK3:[*,*]*.*
```

CORAL

FUNCTION

The CORAL command invokes the CORAL 66 compiler to compile CORAL 66 source file(s). Command qualifiers control output file options and subsequent processing.

REQUIRED PRIVILEGE

PR.COR

FORMAT

PDS> [$]CORAL *[/quals]*

FILE? *filespec+filespec.....*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/LIST[:filespec]	Produce a listing file; named as specified. If the file type is omitted from filespec, the system assumes it to be .LST.
/NOLIST	Do not produce a listing file.
/OBJECT[:filespec]	Produce an object file; name as specified.
/NOOBJECT	Do not produce an object file.
/SWITCHES:(/sw1.../swn)	Use specified CORAL 66 switch options.

filespec

Specification of a source program file to be compiled. If the file type is omitted, the system assumes it to be .COR. No wildcards are allowed.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

Defaults

1 By default, the compiler produces an object file with the name of the source file and with .OBJ as the file type.

2 A listing file is sent to the line printer if you specify /LIST with no file name. /NOLIST is the default qualifier.

CORAL 66 Switches

Table 14–2 CORAL 66 Switches

Switch	Default	Description
/LI:arglist	/LI:SRC:SYM:STA	Specifies the listing options, arglist can be:
		ONL to override the language word "NOLIST"
		SRC for source list
		SYM for symbol table list
		MEX for source list and macro expansions
		STA for statistics
		/NL:arglist (arglist is one of the above arguments) means suppress the printing of the specified list; for example, /NL:STA means suppress the printing of statistics.
/BC	/-BC	Checks array, table element, and switch bounds
/IS:isv	/IS:dis	Specifies instruction set.
		isv=EAE—For extended arithmetic element
		isv=P45—For extended instruction set
		isv=FIS—For extended instruction set and floating instruction set
		isv=FPP—For floating point processor
		dis—Selected at kit installation time.

Table 14–2 (Cont.) CORAL 66 Switches

Switch	Default	Description
/OP:n	/OP:1	Specifies optimization (/OP.) The /OP switch enables code generated by the compiler to be replaced where possible by more efficient code; n can be: 0 No optimization of type OP 1 Two passes of OP optimization 2 Iterative passes of OP optimization until no further reduction.
/OS:n	/OS:0	Specifies optimization (/OS). The /OS switch enables the selection of optimization code constructs, which would not normally be optimized, n can be: 1 Optimization around an anonymous reference 2 Optimization when an overlay is used 4 Optimization around locations to formal location parameters With /OS, any combination of the values 1,2,4 can be summed, for example: n=3—retain logical registers on meeting anonymous reference or data overlay Default value: n=0—no optimization of type OS
/SP	/-SP	Listing file is queued to the spooler and deleted after it is printed. If you do not request spooling, the default /-SP is assumed in which case the listing file is preserved on the device indicated in the listing file specification.
/TE:n	/TE:0	Conditional Compilation. Compile declarations and statements prefixed by TEST m, provided n is greater than or equal to m. n is a decimal integer constant and is positive or zero. If /TE is omitted, TEST declarations and statements are ignored.
/CR	/-CR	Card reader. This switch is used when columns 73-80 of the input file contain sequence numbers and are to be ignored.
/IE	/-IE	Non-IECCA keywords. This switch highlights non-IECCA keywords in the listing file as warning messages.
/WI:n	/WI:132	Sets width. This switch enables the listing file to be set to a specified width. n is a decimal number in the range 8 through 132.
/RO:n	/RO:1	The read-only qualifier renders instruction PSECTS read-only. n can be one of the following: 0 All PSECTS are RW 1 PSECT $CORRO RO, all others RW (default) 2 PSECT $CORRO and all instruction PSECTS RO, others RW.
/TR	/-TR	Setting this switch causes file and line number information to be printed if an error occurs at run-time.

EXAMPLES

- **Example 1:**

```
PDS> CORAL NEWFILE
```

- **Example 2:**

```
PDS> CORAL/SW:(/BC/OP:2) FILES.COR
```

- **Example 3:**

```
$CORAL/OBJ:YRFILE.OBJ MYFILE
```

CREATE

FUNCTION

The CREATE command enables you to create a file or directory.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

General Format

PDS> [$]CREATE *[/quals]*

FILE? *newfile[/filequals]*

Directory Format

PDS> [$]CREATE */DIRECTORY[/ALLOCATION:n]*

DEVICE AND/OR UFD? [dev:]ufd *[/quals]*

Name Format

PDS> [$]CREATE/NAME

FROM? *filename*

TO? *aliasname*

parameter definitions

General Format Paramters:

/quals

One of the following:

Qualifier	Explanation
/DOLLARS	Use only when creating a sequential file in BATCH mode. All batch input up to the next $EOD is used to fill the created file.
/OWN	Causes the destination UFD to be also the owner of the file.
/ALLOCATION:n	Forces the file's initial allocation to be n blocks.
/REPLACE	Replaces an existing file.

newfile

File specification of the file to be created.

/filequals

One of the following:

Qualifier	Explanation
/ALLOCATION:n	Forces the file's initial allocation to be n blocks.
/CONTIGUOUS	Forces the file created to be contiguous.
/PROTECTION:(code)	Create the file with the specified protection access code, see Section 6.2.2.
/FORMAT:type	Specifies the record type of the file. You can use this qualifier only with indexed or relative files. The following types are available: FIXED:n — Fixed length records; n must be specified and is the length of each record in bytes. VARIABLE[:n] — Variable length records. You can use n optionally to specify the maximum length otherwise the default length of 0 is assumed that is, unlimited length. If you specify RELATIVE you must also specify n. CONTROLLED[:n] — Variable length records with a fixed control field. You can specify n optionally to define the length of a record, if not then the default length of 0 is assumed.
/BUCKET_SIZE:m	You can use this qualifier only with indexed or relative files. It specifies the unit of allocation of this kind of file. m specifies the number of blocks to be allocated to each bucket.
/RELATIVE	Specifies relative file organization.
/SEQUENTIAL	Specifies sequential file organization. If you do not specify the organization, SEQUENTIAL is assumed.
[/INDEXED]/KEY: (parameterlist)	Create an ISAM file. If you specify indexed, then NUMBER:1 must appear in one of the key definitions to specify the primary key field.

Qualifier	Explanation	
	Use /KEY to specify a key field within the records. /KEY has three mandatory and two optional parameters. Parameters are separated either by spaces, tabs, or a comma. The parameter list is placed inside round brackets.	
	Parameter list consists of the following:	
	NUMBER:i	(Mandatory parameter.) Key field number. i is 1 for the primary key, 2 for first alternate, 3 for second alternate, and so on.
	POSITION:j	(Mandatory parameter.) Starting byte of the key within the record. j=0 corresponds to the starting byte (byte 0) of the record.
	SIZE:k	(Mandatory parameter.) Length of the key field.
	[NO]UPDATE	Specifies the key field can change during an update process.
	[NO]DUPLICATE	Specifies duplicate keyfields can exist in a record. If UPDATE is specified, DUPLICATE is implicit.

Table 14–3 shows the legitimate combinations of DUPLICATE and UPDATE.

Table 14–3 Duplicate and Update Combinations

Keytype	Combination			
	UPDATE DUPLICATE	UPDATE NODUPLICATE	NOUPDATE DUPLICATE	NOUPDATE NODUPLICATE
Primary	Error	Error	Allowed	Default
Alternate	Default	Error	Allowed	Allowed

Directory Format Paramters:

/DIRECTORY
Specifies that a UFD is being created.

/ALLOCATION:n
Number of files for which room is initially allowed in the directory. The file system extends the directory file as needed, if this value is subsequently exceeded.

dev
Device where the directory is to be created. If it is omitted, the default device is assumed.

ufd
UFD to be created.

/quals
One of the following:

Qualifier	Explanation
/PROTECTION:(code)	Create the directory with the specified protection code. See Section 6.2.2.
/ALLOCATION:n	See the command qualifier above.

CREATE/DIRECTORY

Cannot be interrupted (see Chapter 4).

Name Format Parameters

/NAME

Enables a synomym for a file to be entered in a directory, thus enabling the file to be accessed by more than one name.

filename

Input file specification.

aliasname

New directory entry file specification.

TECHNICAL NOTES

General Format

The CREATE command enables you to create a file. If the file is a sequential file, you can copy input from a terminal or batch stream into it. You can create an empty file suitable for RMS-11 use, then fill it using the MERGE command.

In batch mode, put the text for the new file after the command. Any $ command terminates the file unless the CREATE command contains the /DOLLARS qualifier, which specifies that only the command $EOD can terminate the file.

In interactive mode, the CREATE command reads the input to the new file from your terminal. Typing Ctrl/Z terminates the file.

For SEQUENTIAL files, the CREATE command has the same function as a COPY command that specifies TI: as the device in the input file specification.

If you do not specify any organization, SEQUENTIAL is assumed.

You can specify the BUCKET_SIZE file qualifier only for relative and indexed files. A bucket can contain from 1 to 32 virtual blocks. The default value is one virtual block per bucket. If you increase the number, file processing usually improves.

The relationship between bucket size and record size is important. Because RMS-11 does not allow records to cross bucket boundaries, the number of virtual blocks per bucket must conform to one of the formulas defined in the *RMS-11 MACRO Reference Manual* under the section BKS-Bucket Size.

You can use some CREATE command qualifiers and file qualifiers (including BUCKET_SIZE) only with certain file organizations. These restrictions are shown in Table 14–4.

Table 14–4 Use of CREATE Qualifiers with Different File Organizations

Command Qualifiers	Sequential File Organization	Indexed File Organization	Relative File Organization
	/ALLOCATION /DOLLARS /OWN /REPLACE	/ALLOCATION	/ALLOCATION
File Qualifiers	/ALLOCATION	/ALLOCATION	/ALLOCATION
		/BUCKETSIZE	/BUCKETSIZE
	/CONTIGUOUS	/CONTIGUOUS	/CONTIGUOUS
		/FORMAT	/FORMAT
		/INDEXED	
		/KEY	
	/PROTECTION	/PROTECTION	/PROTECTION
			/RELATIVE
	/SEQUENTIAL		

COMMAND VARIATIONS

Not applicable.

EXAMPLES

- Example 1:

```
PDS> CREATE
FILE? MYDATA.DAT;5
READY FOR INPUT
ABCD
EFGH
Ctrl/Z
PDS> CREATE ANOTHER.DAT/PROTECTION:(OW:RW)
READY FOR INPUT
    .
    .
    .
Ctrl/Z
PDS>
```

- Example 2:

```
$CREATE/DOLLARS DEBUG.MAC
$EOD
```

- Example 3:

```
PDS> CREATE JOHN.DOE/KEY:(NUMBER:1,SIZE:10,POSITION:0)
```

Creates JOHN.DOE as an indexed sequential (ISAM) file with variable length records and one key of reference. The key is 10 (decimal) bytes long and appears in the first byte (byte 0) of each record.

- Example 4:

```
PDS> CREATE/DIRECTORY DK1:[11,17]
```

This example creates the UFD [11,17] on DK1:.

- Example 5:

```
PDS> CREATE/DIRECTORY/ALLOCATION:6 LB0:[14,7]
```

This example creates the UFD [14,7] on LB0: with space initially allowed for 6 files.

- Example 6:

```
PDS> CREATE/DIR DP0:[200,200]/PRO:(SYSTEM:RWED,OWNER:RWED,WORLD:)
```

This example creates a UFD of [200,200] on disk DP0. The access is RWED for System, RWED for Owner ([200,200]), the volume default for Group, and no access for World.

- Example 7:

```
PDS> CREATE/DIR [123,22]
```

This example creates a UFD of [123,22] on the user's current default device.

- Example 8:

```
PDS> CREATE/NAME NAME1.DAT NAME2.DAT
```

Creates an alternative name (an aliasname, NAME2.DAT) for referring to the file NAME1.DAT.

DCL

FUNCTION

The DCL command enables you to return to DCL mode from MCR mode. See the *IAS MCR User's Guide* for a description of MCR mode.

REQUIRED PRIVILEGE

PR.ANY

FORMAT

PDS>> [$]DCL

COMMAND VARIATIONS

Not applicable.

EXAMPLES

```
PDS> MCR
PDS>> PIP DB0:/FR
DB0: HAS 28931. BLOCKS FREE , 142867 USED OUT OF 171798
PDS>> DCL
PDS>
```

DEALLOCATE

FUNCTION

The DEALLOCATE command deallocates a specified device.

REQUIRED PRIVILEGE

PR.DEV

FORMAT

PDS> [$]DEALLOCATE *[/DEVICE]*

DEVICE? *devicename*

parameter definitions

/DEVICE
Optional command qualifier.

devicename
Physical or logical device specification of the device to be deallocated.

COMMAND VARIATIONS

On a multiuser system, the DEALLOCATE command is illegal.

TECHNICAL NOTES

/DEVICE is the only qualifier available to the nonprivileged user; consequently, it is the default and need not be specified.

Normally, the system automatically deallocates a device when you dismount the volume on it or deassign it from a logical unit number. However, when you have issued the ALLOCATE command to obtain access to a nonmountable device that has not been assigned to a logical unit, you must use the DEALLOCATE command to release it. You can also use the DEALLOCATE command after a DEASSIGN/KEEP or DISMOUNT/KEEP command.

EXAMPLES

- Example 1:

```
PDS> DEALLOCATE DK1:
```

- Example 2:

```
$DEALLOCATE/DEVICE DD0:
```

DEASSIGN

FUNCTION

The DEASSIGN command deletes the association between a device and a logical unit number (LUN).

REQUIRED PRIVILEGE

DEASSIGN—PR.RUN
DEASSIGN/TASK—PR.RTC

FORMAT

PDS> [$]DEASSIGN *[/qual]*

LUN? *lun*

parameter definitions

/qual

Qualifier	Explanation
/KEEP	Inhibits any deallocation or dismounting of the associated device.
/TASK:taskname	Deassigns the LUN associated with the specified task.
lun	Logical unit number to be deassigned.

COMMAND VARIATIONS

The DEASSIGN and DEASSIGN/KEEP commands are illegal on a multiuser system.

TECHNICAL NOTES

If the specified LUN is the last to which a device is assigned, the device is dismounted and deallocated unless you specify the command qualifier /KEEP.

EXAMPLES

- Example 1:

```
PDS> DEASSIGN
LUN? 7
```

- Example 2:

```
$DEASSIGN/KEEP 3
```

- Example 3:

```
PDS> DEASS/TASK:MYTASK 4
```

DELETE

FUNCTION

The DELETE command deletes one or more specified files.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> [$]DELETE *[/quals]*

FILE? *filespec1[/filequal][,...filespecn]*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/KEEP[:n]	Deletes files whose version numbers are less than or equal to m-n, where m is the latest version of a file. DELETE/KEEP keeps version numbers m-n+1 to m, if they exist. If versions have already been deleted between m-n and m, fewer than n versions are kept. You can only use DELETE/KEEP when the version field in a file specification is omitted or wild. If you omit n, it is assumed to be 1. **NOTE:** If a DELETE/KEEP is attempted on files that are protected, because of the directory structure, the system attempts a number of times to delete the file. You should press Ctrl/O and wait, or abort the operation.
/[NO]CONFIRM	Deletes files selectively if /CONFIRM is specified by prompting you for a response before deleting (see Technical Notes for response choices). You cannot specify /CONFIRM with any other qualifier. The default is /NOCONFIRM.
/[NO]LOG	Lists the names of the deleted files if /LOG is specified. You cannot specify /LOG with foreign files. The default is /NOLOG.

Qualifier	Explanation
/FILE_ID:m:n	Accesses the file by its file identification number. m:n are the file and sequence numbers (the DIRECTORY/FULL command supplies these). If you specify this qualifier, only a device can be specified as the filespecification; if you give none, the default device is assumed.
/[NO]ERROR_MESSAGE	Suppresses the error message NO SUCH FILES when attempting to delete non-existent files if /NOERROR_MESSAGE is specified. You cannot specify this qualifier with foreign files. The default is /ERROR_MESSAGE.
/NAME	Remove the file aliasname from a directory (see the CREATE/NAME command). Only the directory entry is deleted; the file itself remains and can be accessed using any other directory entries. Use DELETE/NAME only for names you created using CREATE/NAME.

filespec

File specification of a file to be deleted. Wildcards are allowed. You must specify the version field unless you use /KEEP or the file is foreign, (see also /FILE-ID above).

/filequal

/DOS—Modifies the specification of a foreign file in DOS format.

/filequal

/RT-11—Modifies the specification of a foreign file in RT-11 format.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

Typing Ctrl/C aborts the DELETE command.

Table 14–5 Response Choices for the /CONFIRM Qualifier

Letter	Terminator	Operation
Y	RETURN	Delete this file and continue.
Y	Ctrl/Z	Delete this file and return to PDS.
N	RETURN	Save this file and continue.
N	Ctrl/Z	Save this file and return to PDS.
	RETURN	Save this file and continue.
	Ctrl/Z	Save this file and return to PDS.
Q	RETURN	Save this file and return to command mode.
Q	Ctrl/Z	Save this file and return to PDS.

Table 14–5 (Cont.) Response Choices for the /CONFIRM Qualifier

Letter	Terminator	Operation
G	RETURN	Delete this and all remaining candidates, list deleted files, and return to command mode.
G	Ctrl/Z	Delete this and all remaining candidates, list deleted files, and return to PDS.

EXAMPLES

- **Example 1:**

```
PDS> DELETE (A.TYP;1, B.TYP;1, DK0:C.*;*)
```

- **Example 2:**

```
PDS> DELETE/KEEP:1
FILE? DK1:[200,200]*.XYZ
```

- **Example 3:**

```
$DELETE/KEEP DK0:[200,200]*.LIS
```

- **Example 4:**

```
PDS> DELETE DT0:TEST.MAC/DOS
```

- **Example 5:**

```
PDS> DELETE/FILE_ID:36242:27
```

- **Example 6:**

```
PDS> DELETE/NAME NEWNAME.DAT;*
```

- **Example 7:**

```
PDS> DELETE/FILE_ID:36242:27ESC
DEVICE? DK0:
```

- **Example 8:**

```
PDS> DELETE/NOERROR_MESSAGE TEST.DAT;1
```

By specifying /NOERROR_MESSAGE, you do not get an error message if the file TEST.DAT;1 does not exist.

- **Example 9:**

```
PDS> DELETE/CONFIRM [200,200] MYFILE.DAT;*

DELETE FILE DB1:[200,200] MYFILE.DAT;1 [Y/N/G/O]? Y(RETURN)
DELETE FILE DB1:[200,200] MYFILE.DAT;2 [Y/N/G/O]? G(RETURN)

THE FOLLOWING FILES HAVE BEEN DELETED:
DB1:[200,200] MYFILE.DAT;2
DB1:[200,200] MYFILE.DAT;3
PDS>
```

Deletes MYFILE.DAT;1 and goes to the next candidate MYFILE.DAT;2. Deletes this file and all remaining versions of MYFILE.DAT. Lists the deleted files and returns you to PDS.

- Example 10:

```
PDS> DELETE/LOG *.OBJ;*
```

Deletes all files with a filetype of .OBJ and lists all the deleted files.

DIRECTORY

FUNCTION

The DIRECTORY command lists information about a file or group of files within a specified UFD.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> [$]DIRECTORY *[/quals] [filespec1[/filequal],..filespecn]*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/OUTPUT:outfile	Lists information in the specified output file.
/SUMMARY	Specifies that you only need a summary line of the following format: TOTAL OF nnnn./mmmm. BLOCKS IN xxxx. FILES where: nnnn = blocks used (decimal) mmmm = blocks allocated (decimal) xxxx = number of files (decimal)
/BRIEF	Lists only the name, type and version of the file(s). Not available for DOS or RT11 files.
/[NO]ERROR_MESSAGE	Suppresses the error message NO SUCH FILES if /NOERRROR_MESSAGE is specified when you are manipulating files. You cannot specify this qualifier for files on foreign volumes. The default is /ERROR_MESSAGE.
/FREE	Shows free space available on the specified device. If you specify this qualifier, you can specify only the device name in the file specification.

Qualifier	Explanation
/FULL	Lists the following file details (not available for DOS or RT11 files): 1 Name, file type and version. 2 File identification number in the format (file number, file sequence number). 3 Number of blocks used or allocated. 4 File code. null = non-contiguous C = contiguous L = locked 5 Creation date and time. 6 Owner UIC and file protection in the format [group, owner] [system, owner, group, world] 7 Date and time of the last update and the number of revisions.
/WIDTH:n	Specifies width of the full directory output, (n = the number of characters per line). You can specify this qualifier only when listing a full directory, and you must specify /FULL when you specify /WIDTH:n. The default for n is the buffer size of the output device, (n is decimal). You cannot use this qualifier for DOS or RT-11 files.
/PRINT	Output the directory listing to the line printer.
/FILE_ID:m:n	Access a directory by its file identification numbers. m:n are the file and sequence numbers. If you specify this qualifier then you can only specify the device name in the filespec.

filespec

File specification that indicates the directory entries to be listed. Wild cards are allowed. If you omit the filespec, the system lists information about all the files in your default directory.

/filequal

/DOS specifies the file is in DOS format.

/filequal

/RT11 specifies the file is in RT-11 format.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

By default, the DIRECTORY command lists at your terminal (interactive mode) or in your output stream (batch mode) the name, file type, version, size, file code, and date and time of creation of all the files in your current default directory.

You cannot examine the files in a directory for which you do not have read access.

To interrogate the directories of DOS or RT-11 volumes, you must modify the file specification with the /DOS or /RT11 file qualifier.

The directory listing of a DOS or RT-11 file corresponds to the directory format of the foreign volume. The qualifiers /BRIEF and /FULL are not valid for requesting foreign directory information.

When a directory listing of a Files-11 (ANSI) magnetic tape is produced, the creation time for all files appears as 00:00, because there is no place for the creation time of a file in the ANSI file header label.

Only the device name of the file specificaton is valid with /FILE_ID and /FREE.

Typing Ctrl/C aborts the DIRECTORY command.

EXAMPLES

- Example 1:

```
PDS> DIRECTORY ESC
FILE? MATRIX.DAT/DOS
```

- Example 2:

```
PDS> DIR/FULL/OUTPUT:DK0:DIR.DAT ESC
FILE? DK1:[200,200]*.LST
```

- Example 3:

```
$DIR/BR FRED.*
```

- Example 4:

```
PDS> DIR
```

  This prints on the terminal the directory information for the user's default directory.

- Example 5:

```
$DIRECTORY DK1:*.*/RT11
```

- Example 6:

```
PDS> DIRECTORY/FREE ESC
DEVICE? DK0
```

DISABLE

FUNCTION

The DISABLE command enables you temporarily to inhibit the execution of a named task.

REQUIRED PRIVILEGE

PR.RTC

FORMAT

PDS> DISABLE

TASK? *taskname*

parameter definitions

taskname
Installed name of the task being disabled.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

Use the DISABLE command to inhibit task execution. Tasks disabled in this manner cannot be run until they are enabled using the ENABLE command. See the *IAS System Directives Reference Manual*.

EXAMPLES

```
PDS> DISABLE XKE20
```

DISMOUNT

FUNCTION

The DISMOUNT command enables you to dismount a volume on a specified device.

REQUIRED PRIVILEGE

PR.DEV

FORMAT

PDS> [$]DISMOUNT *[/qual]*

DEVICE? *devicename*

[VOLUME-ID? *volumeident]*

parameter definitions

/qual

One of the following:

Qualifier	Explanation
/KEEP	Instructs the system not to deallocate the device.
/GLOBAL	Instructs the system to dismount the globally mounted device (provided no other users currently have it mounted).
/REALTIME	Instructs the system to dismount the device that was mounted for exclusive access by real time tasks.
/LOCK	Volume is marked for dismount even if the files on it are currently being accessed. Further access is locked and the volume is dismounted when the current file access is completed.

devicename

Physical or logical name of the device to be dismounted.

volumeident

Optional parameter that specifies the identification of the volume to be dismounted (see the MOUNT command).

COMMAND VARIATIONS

On multiuser systems the /GLOBAL, /REALTIME, and /KEEP qualifiers are illegal.

/LOCK is illegal on a timesharing system.

TECHNICAL NOTES

Timesharing Systems

If you do not specify /KEEP, the system dismounts the volume on the device, deallocates the device, and deassigns it from any logical unit number.

If the qualifiers /GLOBAL or /REALTIME are omitted, the default action of the DISMOUNT command is to dismount the volume for the timesharing user issuing the command. In this case the system does the following:

1 Dismounts the volume from the device for the user.

2 Deallocates the device if it was previously allocated (unless /KEEP is used).

3 Deassigns the device from any logical unit number(s) that the user has assigned.

Only if the device was allocated to you (by means of the ALLOCATE command) does the command qualifier /KEEP prevent the system from deallocating the device.

A volume mounted for several users on a shareable device is not unloaded until the last user accessing the volume issues a DISMOUNT command. This final dismount causes an unload request to be printed on the system console. However, the unload request for volumes mounted /GLOBAL or /REALTIME is not issued until an explicit DISM/GLOBAL or DISM/REAL is specified.

Multiuser Systems

Volumes can be dismounted by anyone; that is, a multiuser system does not provide any form of device management or protection.

Multiuser and Timesharing Systems

Dismounting any member of a multivolume magtape set causes all tapes within that set to be dismounted. See the MOUNT command for further details.

EXAMPLES

- **Example 1:**

```
PDS> DISMOUNT
DEVICE? MY0:
```

- **Example 2:**

```
$DISMOUNT/KEEP TU1: ACCTS
```

DUMP

FUNCTION

The DUMP command produces a listing of the contents of a file.

REQUIRED PRIVILEGE

PR.DUM

FORMAT

PDS> [$]DUMP *[/quals]*

FILE? *filespec*

or:

PDS> [$]DUMP *[/quals][/qual2]*

DEVICE? *[device_specification]*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/ASCII	Specifies that you want a listing of data in ASCII mode. The control characters (0 - 37) are printed as Ctrl/ followed by the alphabetic character corresponding to the character code +100(octal). For example, bell (code 7) is printed as Ctrl/ (code 107). Lower case characters (140–177) are printed as % followed by the corresponding upper case character (character code minus 40).

Qualifier	Explanation
/BLOCKS:(m-n)	Specifies that you want a listing of the first (m) to (n) logical or virtual blocks where m and n are octal numbers. If either m or n is than 16 bits (that is, greater than 177777) then you must specify it as two numbers, as follows: (a,b) where a is the first 16 bits and b is the second 16 bits. If you specify the /BLOCKS:(m-n) switch as /BLOCK:0 in file mode, no physical blocks are listed. This is useful when you want to list only the header portion of the file. (See the /HEADER switch below). This qualifier is mandatory in device mode as it specifies the range of physical blocks to be dumped (see the Technical Notes).
/BYTE	Specifies that you want a listing of data in octal byte format.
/DECIMAL	Specifies that you want the data dumped in decimal word format.
/FORMAT:HEADER	Formats data blocks with Files-11 header structure; outputs other blocks as an unformatted octal dump.
/HEADER	Results in a listing of the file header as well as the specified portion of the file. If you only want the header portion of the file, you can specify /HEADER/BLOCKS:0.
/HEADER/FORMAT:FILES-11	Formats the header as a Files-11 header (default).
/HEADER/NOFORMAT	Prints the header as an unformatted octal dump.
/HEXADECIMAL	Specifies that you want the data dumped in hexadecimal byte format. Note that a hexadecimal dump reads from right to left.
/LONGWORD	Specifies that you want the data dumped in hexadecimal double-word format.
/NUMBER[:n]	Enables control of line numbers. Line numbers are normally reset to zero whenever a block boundary is crossed. The /NUMBER[:n] qualifier enables you to number lines sequentially for the full extent of the file, that is, the line numbers are not reset when block boundaries are crossed. The optional value (:n) enables you to specify the value of the first line number. The default is 0.
/OUTPUT:filespec	Outputs the listing to the specified file or device.
/PRINT	Outputs the listing to the default printer.
/RADIX_50	Specifies that you want data to be dumped in RADIX-50 format.
/RECORD	Specifies that you want data to be dumped a record at a time.
/REWIND	Specifies that you want DUMP to issue the rewind command before referencing a specified tape. You can specify this qualifier at any time to reposition a tape at the load point (BOT).
/START	Provides the starting block number of the file and an indication of whether or not it is contiguous. For example: PDS> DUMP/START DK0:RICKSFILE.DAT;3 STARTING BLOCK NUMBER = 0,135163 C File RICKSFILE.DAT, version 3, is a contiguous file starting at block 0,135163. (See /BLOCKS:(m:n) for a description of block and number format.)
/WORD	Specifies that you want the data dumped in hexadecimal word format.

filespec

Specification of the file or device whose blocks are to be dumped.

qual2

One of the following:

Qualifier	Explanation
/DENSITY:n	Specifies the density of a TU16 input magnetic tape. The value for n can be 800 or 1600. If you do not specify /DENSITY, the tape is read at the density currently set in the tape controller. (See the MOUNT command in this manual, and the *IAS MCR User's Guide* for a description of the MOUNT command and /DENSITY or /DENS qualifiers). You must specify the qualifier /BLOCKS and the device specification with this qualifier.
/FILE-ID:m:n	Where m and n are the file number and file sequence number, respectively, of the input file.

device_specification

Optional device type and unit number.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

The DUMP command ignores any print formatting characters that can appear in the records. The listing is printed at your terminal by default. However, you can specify a different output device by using the /OUTPUT or /PRINT qualifiers.

The DUMP command can operate in one of three modes:

1 File Mode

In file mode, you specify a file. All, or a specified range (/BLOCKS:m-n) of blocks of the named file are listed. The blocks are numbered from 1 to n, where the first block is 1 and the last block in the file is numbered n. You must mount the input volume and it must contain named files. You can not use a wild card in the filename.

2 Device Mode

In device mode, you specify a device; and a specified range /BLOCKS:(m-n) of physical blocks to be listed.

a. You need the /BLOCKS:(m-n) qualifier.

b. Physical blocks refer to the actual 512-byte blocks on disk and DECtape, and physical records on magtape and cassette. The DUMP command handles physical records up to 2048 bytes in length.

c. Physical blocks are numbered from 0 to n, where n is the last physical block on the device.

d. You must mount the volume to be listed FOREIGN.

3 Record mode

In record mode, you can dump a whole file or the first part of it. Each record of the file is printed separately in the specified format. The first block you dump must always be block 1 of the file, but you can use any block number for the last block.

EXAMPLES

- **Example 1:**

```
PDS> DUMP MYFILE.DAT
```

- **Example 2:**

```
PDS> DUMP/ASCII
FILE? MYDATA.DAT
```

- **Example 3:**

```
$DUMP A.MAC;4
```

- **Example 4:**

```
PDS> MOUNT/FOR/NOOP DK0: MYDISK
PDS> DUMP/BLOCK:(5-14) DK0:
```

EDIT

FUNCTION

The EDIT command invokes one of the following IAS text editors:

1 The line text editor (EDI), (interactive use).

2 The source language input program and editor (SLIPER), (batch-oriented).

3 The DEC standard editor (EDT), (interactive use).

4 The keypad editor (KED or K52) for users with the FMS-11 optional software (interactive use on VT100 or VT52 terminals).

The *RSX-11M/11-PLUS Utilities Manual* describes EDI and SLIPER in full. The *EDT Editor Manual* describes EDT in full, and the *PDP-11 Keypad Editor User's Guide* describes KED and K52.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> [$]EDIT *[/editor][/quals]*

FILE? *filespec*

parameter definitions

/editor

One of the following:

/EDI	Invokes the line text editor. This is the default.
/SLIPER	Invokes the batch editor SLIPER.
/EDT	Invokes the DEC standard editor.
/KED	Invokes the keypad editor (for use on VT100 terminals).
/K52	Invokes the keypad editor (for use on VT52 terminals).

filespec

Specification of an existing file to be edited or a new file to be created. You must include a filetype.

/quals

One or more command qualifiers. The qualifiers available with each editor are:

Editor	Qualifier	Default
/EDI		
/EDI or /SLIPER	/OUTPUT[:filespec] /NOOUTPUT	/OUTPUT
	/LIST[:filespec] /NOLIST	/NOLIST
	/AUDIT[:(params)] POSITION:m SIZE:n REPORT_TRUNCATION /NOAUDIT	/AUDIT
	/BLANK /NOBLANK	/BLANK
	/CHECKSUM[:n] /NOCHECKSUM	/NOCHECKSUM
	/DOUBLE /NODOUBLE	/NODOUBLE
	/TRUNCATE[:n] /NOTRUNCATE	/NOTRUNCATE
/EDT	/COMMAND[:filespec] /NOCOMMAND	/COMMAND
	/OUTPUT[:filespec] /NOOUTPUT	/OUTPUT
	/JOURNAL[:filespec] /NOJOURNAL	/JOURNAL
	/RECOVER /NORECOVER	/NORECOVER
	/READONLY /NOREADONLY	/NOREADONLY
/KED or /K52	/OUTPUT[:filespec] /NOOUTPUT	/OUTPUT
	/BLOCKS:n	N/A
	/EMBEDDED_CARRIAGE_CONTROL /IMPLIED_CARRIAGE_CONTROL /FORTRAN_CARRIAGE_CONTROL	/IMPLIED_CARRIAGE_CONTROL
	/INSPECT /NOINSPECT	/NOINSPECT

EXAMPLES

1 Example 1 (SLIPER)

```
PDS> EDI/SLIP/OUTPUT:YURFILE.MAC MYFILE.MAC
@EDI.CMD
/
```

SLIPER opens the input file MYFILE.MAC and executes the SLIPER commands contained in the file EDIT.CMD. The output file is called YURFILE.MAC.

2 Example 2 (EDI)

```
PDS> EDI
FILE? SUPER.ADV
[00055 LINES READ IN]
[PAGE 1]
*EX
PDS>
```

3 Example 3 (EDT)

```
PDS> EDIT/EDT/OUTPUT:PAYROLL.JAM/JOURNAL:REC.ONE PAYROLL.DAT
```

EDT copies the contents of the file PAYROLL.DAT into the main buffer and opens a journal file named REC.ONE. When you end the edit session with the EXIT command, EDT saves the contents of the main text buffer in the file named PAYROLL.JAM. If the editing session ends in other than a normal exit (such as a system failure), EDT retains the journal file REC.ONE.

4 Example 4 (KED)

```
PDS> EDI/KED/INSPECT GAYE.MAC
```

Opens the file GAYE.MAC for inspection only. This command invokes the Keypad editor for the VT100 terminal.

ENABLE

FUNCTION

The ENABLE command reverses the effect of the DISABLE command.

REQUIRED PRIVILEGE

PR.RTC

FORMAT

PDS> ENABLE

TASK? *taskname*

parameter definitions

taskname
Name of the installed task being enabled.

COMMAND VARIATIONS

Not applicable.

EXAMPLES

```
PDS> ENABLE XKE20
```

$EOD

FUNCTION

The end of data ($EOD) command terminates a data stream or the input to a file created by a $CREATE/DOLLARS command.

REQUIRED PRIVILEGE

Not applicable.

FORMAT

$EOD

The $EOD command has no parameters.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

When you create a file in batch stream using the $CREATE command, the end of file indication is taken as the next line beginning with a $ character. Use the $CREATE/DOLLARS command if your file is going to contain a $ character as part of the data. In this case, $EOD is taken as the end of file specification.

EXAMPLES

```
$CREATE/DOLLARS PAYROLL.DAT
; PAYROLL UPDATE FOR 27-JAN
DOE JOHN
$476.32 $46.12 17 P
BLOGGS FRED
$316.41 $96.24 23 R
$EOD
```

This example uses $EOD to terminate a file of data commands. 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.

$EOJ

FUNCTION

The End of Job ($EOJ) command terminates a batch job.

REQUIRED PRIVILEGE

Not applicable.

FORMAT

$EOJ

The $EOJ command has no parameters.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

The $EOJ command must be the last command in a batch job command stream. Anything in the batch job after $EOJ is ignored. When the system encounters $EOJ, the batch job is terminated and, on a timesharing system, any claimed devices are dismounted and released. The batch log and associated files are queued for printing on the default line printer.

EXAMPLES

```
$JOB WILSON TESTRUN 2
$MOUNT DK: TEST DD0:
$ASSIGN DD0: 7
$RUN TEST
$DISMOUNT DD0:
$EOJ
```

FIX

FUNCTION

The FIX command enables you to fix a task in its installed partition.

REQUIRED PRIVILEGE

PR.RTC

FORMAT

PDS> FIX

TASK? *taskname*

[TERMINAL? *terminal]*

parameter definitions

taskname
Name of the installed task to be fixed in memory.

terminal
terminal for which the task is to be It is possible to fix the same task for more than one terminal.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

The advantage you gain by fixing tasks is that there is no delay when you load the task for the first time. Also, you can prevent memory fragmentation by fixing tasks in a system-controlled partition. You can fix a task only if you built it as a fixable and non-checkpointable task. See the *IAS task builder Reference Manual* and the *IAS Executive Facilities Reference Manual*.

EXAMPLES

- Example 1:

```
PDS> FIX MYTSK
```

- Example 2:

```
PDS> FIX MART3 TT4
```

FORTRAN

FUNCTION

The FORTRAN command invokes a FORTRAN compiler to compile a FORTRAN-IV or FORTRAN-IV PLUS source file. Command qualifiers control output file options and subsequent processing.

REQUIRED PRIVILEGE

PR.FOR

FORMAT

PDS> [$]FORTRAN *[/quals]*

FILE? *filespec*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/FOR	Invokes the FORTRAN-IV compiler. Applicable to systems that have both FORTRAN IV and FORTRAN IV PLUS compilers. If omitted, the system invokes its default compiler.
/F4P	Invokes the FORTRAN IV-PLUS compiler. Applicable to systems that have both FORTRAN IV and FORTRAN IV PLUS compilers. If omitted, the system invokes its default compiler.
/F77	Invokes the FORTRAN-77 compiler. If omitted, the system invokes its default compiler.
/LIST[:filespec]	Produces a listing file named as indicated. If the filetype is omitted from filespec, the system defaults it to .LST.
/NOLIST	Does not produce a listing file.
/OBJECT[:filespec]	Produces an object file, named as specified. If the file type is omitted, the system defaults to .OBJ.
/NOOBJECT	Does not produce an object file.

Qualifier	Explanation
/SWITCHES:(/sw1.../swn)	Uses specified FORTRAN IV or FORTRAN IV-PLUS switch options. See Table 14–6.

filespec

Specification of a source program file to be compiled. If you omit the filetype, the system assumes it to be .FTN. Wildcards are not allowed.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

Defaults

1 By default, the compiler produces an object file with the name of the source file and with .OBJ as the file type.

2 A listing file is sent to the line printer if /LIST is specified with no file name. /NOLIST is the default qualifier.

FORTRAN-IV Switches

Table 14–6 and Table 14–7 list the FORTRAN IV and FORTRAN IV-PLUS switches.

Table 14–6 FORTRAN-IV Switches

Switch	Default	Description
/LI:n	/LI:3	Specifies the listing options. The argument n is encoded as follows: /LI:0 or /NOLI — List diagnostics only /LI:1 or /LI:SRC — List source program and diagnostics only /LI:2 or /LI:MAP — List storage map and diagnostics only /LI:4 or /LI:COD — List generated code and diagnostics only You can specify any combination of the above options by summing the numeric argument values for the list options you require. /LI:7 or /LI:ALL — Requests a source listing, a storage map listing, and a generated code listing. If you omit this switch, the default list option is /LI:3, source and storage map.
/CD:xxx	/CD:THR	Selects type of object code to be generated.

Table 14–6 (Cont.) FORTRAN-IV Switches

Switch	Default	Description
		The valid values are: EAE (selects code for EAE hardware) EIS (selects code for EIS hardware) FIS (selects code for EIS and FIS hardware) THR (selects threaded code)
/DE	/NODE	Compiles lines with a D in column 1. Default treats these as comment lines.
/DI	/NODI	Enables expanded listings of compiler internal diagnostic information.
/EX	/NOEX	Reads a full 80 columns of each record in the source file. Only the first 72 columns are read by default.
/ID	/NOID	Prints FORTRAN identification and version number. The default (/NOID) prevents the printing of the identification and version number.
/LO	/LO	Prints on your terminal the names of program units (from PROGRAM, FUNCTION SUBROUTINE, and BLOCK DATA statements) as each program unit is compiled. Note that .MAIN. refers to the main program; and .DATA. refers to an unnamed BLOCK DATA.
/NOOP:xxx	/NOOP:xxx	Disables optimizations from multiple arguments: SPD Disables optimization for program speed. Optimization now takes place for minimal program size. CSE Disables common subexpression elimination. STR Disables strength reduction optimization for in-line code generation. BND Disables global register bindings for in-line code generation.
/OP:xxx	/OP:xxx	Selects optimizations from multiple arguments: SPD Optimizes for speed of object program execution as opposed to minimal program size. CSE Enables common subexpression elimination. STR Enables strength reduction optimization for in-line code generation. BND Enables global register bindings for in-line code generation.
/RO		Causes pure code and pure data program sections to take the RO (read-only) attribute.
/SN	/SN	Includes internal sequence numbers (ISN). Reduces storage requirements for generated code and slightly increases execution speed but disables line number information during traceback.
/SP	/SP	Automatically spools listing file.
/I4	/NOI4	Two-word default allocation for integer variables. Normally, single storage words are the default allocation for integer variables not given an explicit length specification, that is, integer*2 or integer*4. Only one word is used for computation.
/VA	/VA	Enables vectoring of arrays.
/WR	/WR	Enables compiler warning diagnostics.

NOTE: Switch default summary:
(/LI:3/NODE/NOEX/NOID/OP/SN/NOI4/VA/WR)

FORTRAN-IV PLUS Switches

Table 14–7 FORTRAN-IV Plus Switches

Switch	Default	Description
/CK	/NOCK	Generates code to check that all array references are within the array bounds specified by the program. Individual subscripts are not checked against dimensional specifications.
/CO:n	/CO:5	A maximum of n continuation lines is permitted in the program, where n is an integer from 0 through 99. The default value is n=5. Note that you can express n in either octal or decimal radix. If a decimal point follows the number, it is interpreted in decimal radix; otherwise, it is interpreted in octal radix.
/DE	/NODE	Compiles lines with a D in column one. These lines are treated as comment lines by the default /NODE. See the *FORTRAN Language Reference Manual.*
/ID	/NOID	Print FORTRAN IV-PLUS identification and version number.
/I4	/NOI4	Allocates two words for default length of Integer and Logical variables. Normally, single storage words are the default allocation for all integer or logical variables not given an explicit length definition, that is, INTEGER*2, LOGICAL*4. See Section 3.3 of the *FORTRAN IV-PLUS User's Guide.*
/Li:n	/Li:2	Specifies listing options; n is an integer from 0 through 3. The argument is coded as follows: n=0 Minimal listing file: diagnostic messages and program section summary only n=1 Source listing and program section summary n=2 (Default) source listing, program section summary, and symbol table n=3 Source listing, assembly code, program section summary, and symbol table
/TR:XXX	/TR:BLOCKS	Controls the amount of extra code that the compiled output includes for use by the OTS during error traceback. This code produces diagnostic information and identifies which statement in the FORTRAN source program causes the detection of an error condition at execution.
/TR		Same as /TR:ALL
/TR:ALL		Error traceback information is compiled for all source statements, and function and subroutine entries.
/TR:LINES		Same as /TR:ALL option.
/TR:BLOCKS		Traceback information is compiled for subroutine and function entries and for selected source statements. The source statements selected by the compiler are initial statements in sequences commonly called basic blocks. The compiler treats such a sequence of statements as a unit for performing certain types of optimization. Basic blocks usually begin at each labelled statement, each DO statement, and so on.
/TR:NAMES		Traceback information is compiled only for subroutine and function entries.
/TR:NONE		No traceback information is produced.
/NOTR		Same as /TR:NONE.

NOTE: The switch setting /TR is advisable during program development and testing. The default setting /TR:BLOCKS is advisable for most programs in regular use. You can use the setting /NOTR for obtaining fast execution and smallest code, but it provides no information to the OTS for diagnostic message traceback.

EXAMPLES

- **Example 1:**

```
PDS> FORTRAN NEWFILE
```

- **Example 2:**

```
PDS> FORTRAN/SW:(/CK/CO:7) FILES.FTN
```

- **Example 3:**

```
$FORTRAN/OBJ:YRFILE.OBJ MYFILE
```

GOTO

FUNCTION

The GOTO command transfers control, in a command file or batch stream, to the next occurrence of a command line prefixed by a specified label.

REQUIRED PRIVILEGE

ANY

FORMAT

[$]GOTO *label*

parameter definitions

label

Alphanumeric string that must begin with an alphabetic character. The label must also appear, together with a colon, in front of a command later in the file.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

You can use the GOTO command only in indirect or batch command files; it is ignored if you use it in interactive mode.

You can use the GOTO command by itself or as an action in an ON command. When control is transferred, the system ignores all intervening commands. If no matching label is found, no further processing takes place within the command file or batch job. The GOTO command cannot transfer control to a command labelled earlier in the file.

EXAMPLES

```
$JOB SYSTEM
$ON ERROR GOTO L10
$MACRO MYPROG
$LINK MYPROG
$RUN MYPROG
$GOTO L20
$L10: RUN OLDPROG
$L20: RUN TEST
$EOJ
```

In this example, if an error, (that is, ERROR or SEVERE ERROR) is encountered in the MACRO, LINK or RUN commands, control is passed to LI0 and OLDPROG is run. TEST is always run.

HELP

FUNCTION

The HELP command enables you to display information about PDS commands.

REQUIRED PRIVILEGE

ANY

FORMAT

PDS> HELP *[command] [parameter]*

parameter definitions

command

The particular PDS command you need information about. If you omit the command, HELP lists all available PDS commands.

parameter

Parameter of the specified command you need information about. If you omit the parameter, all parameters and qualifiers of that command are listed.

COMMAND VARIATIONS

On a multiuser system, HELP lists commands that require privilege PR.SCI. These are identified by a + sign placed at the beginning of the command.

TECHNICAL NOTES

The precise information displayed depends on your current state as a user. Before you are logged in, you can type HELP to display information on how to log in.

When you are logged in, the HELP command provides help at a number of levels:

1 To obtain a terminal listing of all PDS commands, type a HELP command with no parameters.

2 To obtain information on the format of a specific command, supply the required command name as a parameter to the HELP command. For example:

```
PDS> HELP LIBRARIAN
```

The format of the command (in this example the LIBRARIAN command) and a list of the relevant qualifiers and parameters is listed.

You can obtain further information about qualifiers and parameters for the command by supplying the qualifier or parameter name as an additional parameter to the HELP command. For example:

```
PDS> HELP LIBR EXTRACT
```

This command provides full details of the EXTRACT feature of the LIBRARIAN command.

You can supply only those qualifiers and parameters that HELP flags with two asterisks (**) as the additional parameter.

EXAMPLES

- Example 1:

```
PDS> HELP
PLEASE LOGIN-
PDS> LOGIN<CR>
USER NAME? USER-NAME<CR>
PASSWORD? PASSWORD<CR>
YOUR USER-NAME AND PASSWORD WILL BE SUPPLIED BY THE SYSTEM MANAGER
```

This example shows that typing HELP before you log in displays information to help you log in.

- Example 2:

```
PDS> HELP DISMOUNT
  DISMOUNT [/QUALIFIER] DEV-NAME [VOLUME-LABEL]
     /KEEP     - DO NOT DEALLOCATE DEVICE
     /GLOBAL   - DISMOUNT SPECIFIED "GLOBAL" VOLUME
     /REALTIME - DISMOUNT VOLUME MOUNTED FOR REALTIME ACCESS
```

This example displays information about the DISMOUNT command.

IDENTIFY

FUNCTION

The IDENTIFY command displays the identity of the system utility task on your terminal.

NOTE: Not all system utilities can identify themselves.

REQUIRED PRIVILEGE

ANY

FORMAT

PDS> [$]IDENTIFY

UTILITY? *taskname*

parameter definitions

taskname
Three-character utility name (for example: PIP, KED, K52, FLX, and so on).

COMMAND VARIATIONS

Not applicable.

EXAMPLES

- Example 1:

```
PDS> IDENTIFY
UTILITY? PIP
PIP -- PIP VERSION D1332
```

- **Example 2:**

```
PDS> IDENTIFY FLX
FLX -- FLX VERSION M11
```

INITIALIZE

FUNCTION

The INITIALIZE command enables you to initialize a volume.

REQUIRED PRIVILEGE

PR.DEV

FORMAT

PDS> [$]INITIALIZE *[/quals]*

DEVICE? *devicename*

[VOLUME-ID? *volumeid]*

parameter definitions

/quals

One of the following:

Table 14–8 DOS and RT11 Initialization Qualifiers

Qualifier	Explanation
/DOS	Initializes volume in DOS format.
/RT11[:n]	Initializes volume in RT11 format. n specifies the number of extra words required per directory entry.

If you do not specify either of these qualifiers, the file is initialized in FILES-11 format.

The following qualifier applies to RT11 only.

Table 14–9 RT11 Qualifier

Qualifier	Explanation
/NUMBER:n	Specifies the number of directory segments to allocate to the RT11 volume. The default is 4.

The following file qualifiers apply to Files-11 only.

Table 14–10 RT11 Qualifier

Qualifier	Explanation
/ACCESSED:n	Specifies the number of directories to be kept pre-accessed while the volume is in use. See the *IAS Performance and Tuning Guide* for a description of pre-accessed directories. The default is 3. This qualifier is for disk and DECtape only.
/BAD:option	Specifies that the bad block file is to be initialized. You can specify known bad blocks by either of the following options. AUTOMATIC Use bad block data left on volume by the BAD BLOCK system task MANUAL Specify bad blocks after the command. This qualifier is for disk and DECtape only.
/DENSITY:n	Specifies magnetic tape density to be either 800 or 1600 bpi. The default is 800 bpi. This qualifier is for magtape only.
/EXTENSION:n	Specifies default file extension size in blocks; that is, the amount of space allocated when you extend a file on a volume. For further information, see the *IAS Performance and Tuning Guide*. The default is 5. This qualifier is for disk and DECtape only.
/FILE_PROTECTION:(code)	Specifies the default protection for files created on the volume. See Section 6.2 for a description of protection codes. This qualifier is for disk an DECtape only.
/HEADERS:n	Specifies the number of file headers to be allocated in the initial index file. The default is 16 headers. This qualifier is for disk and DECtape only.
/INDEX:option	Specifies the position of the index file on the volume. Option can be BEGINNING, MIDDLE, END, or n (where n is a logical block number). The default is MIDDLE. This qualifier is for disk and DECtape only
/MAXIMUM_FILES:n	Specifies maximum number of files allowed on the volume. See Table 14–11 for details of the maximum number of files per volume for each Files-11 device and for details of the default allocations. This qualifier is for disk and DECtape only.
/OWNER:uic	Specifies the owner of the volume. The default is [1,1].
/PROTECTION:(code)	Specifies the volume access privilege. See Section 6.2.2 for a description of protection codes.
/WINDOW:n	Default window size for file access; that is, the number of retrieval pointers. For further information, see the *IAS Performance and Tuning Guide*. The default is 7.

devicename

Name of the device where the volume to be initialized is loaded. The unit number must be specified.

volumeid

Optional volume identification.

COMMAND VARIATIONS

On a full timesharing system, you can initialize a Files-11 volume if the device where the volume is loaded is allocated to you. You must not mount the volume.

On a multiuser system, only privileged users (that is, those with a group code less than 10 or PR.RTC privilege) can initialize a Files-11 volume. The volume must not be mounted.

Table 14–11 shows the maximum files per volume for each Files-11 device.

Table 14–11 Maximum Files Per Device

Disk Name	Volume Size	Maximum Files	Default Files	Maximum Indx Hdrs	Default Indx Hdrs	Default Allocation
DECTAPE	576	278	34	1	1	16
DECTAPE II	512	247	30	1	1	16
RA60	400,176	65,500	24,617	3	1	12,308
RA80	236,964	65,500	14,629	3	1	7,314
RA81	891,072	65,500	54,815	3	3	51,699
RF	1,024	499	62	1	1	16
RK05[1]	4,800	2,357	294	1	1	147
RK06	27,126	13,344	1,668	1	1	834
RK07	53,790	26,466	3,308	2	1	1,654
RL01	10,240	5,034	629	1	1	314
RL02	20,480	10,074	1,259	1	1	629
RM02	131,680	64,798	8,099	3	1	4,049
RM03	131,680	64,798	8,099	3	1	4,049
*RM05	500,384	65,500	30,781	3	2	25,593
RP02	40,000	19,680	2,460	1	1	1,230
RP03	80,000	39,365	4,920	2	1	2,460
RP04	171,798	65,500	10,567	3	1	5,283
RP05	171,798	65,500	10,567	3	1	5,283

[1]This device is not backwards-compatible by default.

- MAXIMUM FILES—means the maximum value that can be specified for /MAXIMUM_FILES:n
- DEFAULT FILES—means the default value for /MAXIMUM_FILES:n
- DEFAULT ALLOCATION—means the default value for /HEADERS:n

Table 14–11 (Cont.) Maximum Files Per Device

Disk Name	Volume Size	Maximum Files	Default Files	Maximum Indx Hdrs	Default Indx Hdrs	Default Allocation
RP06	340,670	65,500	20,956	3	1	10,478
RS03	1,024	499	62	1	1	16
RS04	2,048	1,003	125	1	1	16
RX01	494	238	29	1	1	16
RX02	988	481	60	1	1	16

- MAXIMUM FILES—means the maximum value that can be specified for /MAXIMUM_FILES:n
- DEFAULT FILES—means the default value for /MAXIMUM_FILES:n
- DEFAULT ALLOCATION—means the default value for /HEADERS:n

TECHNICAL NOTES

To initialize a Files-11 volume, you must physically load it on a device but you must not mount it.

To initialize an RT-11 or DOS volume, you must mount the volume as foreign (MOUNT/FOREIGN).

You can specify a volume-id only when initializing a Files-11 volume. For disk or DECtape, the volume-id is 1 to 12 alphanumeric characters. For magnetic tape, the volume-id is 1 to 6 alphanumeric characters.

Protection codes applicable to /PROTECTION and /FILE_PROTECTION are described in Chapter 6.

EXAMPLES

- Example 1:

```
PDS> MOUNT/FOREIGN DK0: MYDOSDISK
PDS> INIT/DOS DK0:
```

- Example 2:

```
PDS> MOUN/FOR DT RT11SOURCE RT1:
PDS> INIT/RT11:6/NUMBER:3 RT1:
```

- Example 3:

```
PDS> MOUNT/FOR/NOOPER DT0: DOSDECTAPE MY0
PDS> INIT/DOS MY0:
```

- Example 4:

```
PDS> ALLOCATE DK0:
PDS> INIT DK0: CARFRABLU
```

- **Example 5:**

```
PDS> ALLOC/DEVICE DK1: DD1:
PDS> INIT/OWN:[200,200]/PROT:(SYS:RWED WO:)/IND:BEG
DEVICE? DD1: MYDISK1
```

- **Example 6:**

```
PDS> ALLOCATE MM0:
PDS> INIT/DENS:1600 MM0 MYTAPE
```

INSTALL

FUNCTION

The INSTALL command enables you to install a task, a resident library SGA, a common area SGA, or a region SGA.

REQUIRED PRIVILEGE

PR.RTC

FORMAT

PDS> [$]INSTALL *[/quals]*

FILE? *filespec*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/TASK[:name]	Installs a task and optionally assigns the task name. If you do not specify the name, the task is installed with the name specified in the task build (LINK), or the name is taken from .TITLE in the source. /TASK is the default.
/COMMON[:name]	Installs an SGA as a common area. name is a 1- to 6-character name that overrides the name set at link time.
/LIBRARY[:name]	Installs an SGA as a resident library. name is a 1- to 6-character name that overrides the name set at link time.
/REGION[:name]	Installs an SGA as a region. name is a 1- to 6-character name that overrides the name set at link time.
/SYSTEM:tname	Installs a system library task with the name $$$tname, where tname is a 1-3 character mnemonic. For example INS/SYS:F4P [11,1]F4P installs F4P as $$$F4P.

You can specify the following qualifiers with any of the above:

Qualifier	Explanation
/PARTITION:name	Installs a task or SGA in the specified partition.
/UIC:uic	Changes the task UIC or the owning UIC of an SGA.
	Defaults for the above are the values as set at link time. See the *IAS task builder Reference Manual.*

You can use the following qualifiers when installing a task (/TASK,/SYSTEM).

Qualifier	Explanation
/POOL:number	Sets the pool limit of the task to be installed. The pool limit value can range from 0 to 255 decimal, and represents the maximum number of 8-word nodes that the task is allowed to use at any one time.
/PRIORITY:number	Sets the execution priority to be assigned to the task. Priority ranges from 1 to 250.
/INCREASE: tasksizeincrement	Overrides the EXTTSK option specified in the LINK command. This qualifier specifies the decimal number of words by which the upper read/write area of the task being installed is to be extended. The value specified is rounded up to the next 32-word boundary.

You can use the following qualifier when installing an SGA (/LIB,/COM,/REG).

Qualifier	Explanation
/ACCESS:access	Nonowner access permitted to the SGA being installed. You can express the nonowner access in the usual form for protection. See Section 6.2.2 for a description of file protection. Alternatively, you can specify the following abbreviated protection codes: NA No access by non-owners, equivalent to (D,RWED,,). This is the default. RO Read-only access by non-owners, equivalent to (RD,RWED,R,R). RW Read-write access by non-owners, equivalent to (RWD,RWED,RW,RW). The system adds delete access to the system and owner protection groups, if you do not specify it. This prevents an SGA being created that cannot be deleted when required. The owner always has RW access.

filespec

Specification of the file containing the task image to be installed. If you omit the file type, a default of .TSK is assumed.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

The INSTALL command causes the system to find and note the physical position of a task or shareable global area (SGA) held on disk. This enables fast loading into memory. A task cannot be run in real-time processing (see the RUN command) unless it has been installed. A task cannot be installed until all the SGAs it uses have been installed. The effect of INSTALL is reversed by the REMOVE command.

At install time, you can take the opportunity to override certain task attributes set at link time, and can specify non-owner access rights to an SGA. These changes affect only the installed version, not the task or SGA task image file.

To install a task, you must have extend access to the file containing the task image.

EXAMPLES

- Example 1:

```
PDS> INSTALL [11,1]PIP
```

- Example 2:

```
PDS> INSTALL/TASK:JK304/PRIORITY:200 DK1:COMMS.TSK;3
```

  Install the task image held in file COMMS.TSK;3 on DK1. Give it the installed name JK304 and priority 200.

- Example 3:

```
PDS> INSTALL/COMMON:COMLOL/ACCESS:RO JK61.TSK;4
```

  Install the SGA held in file JK61.TSK;4 on your default device. Give it the installed name COMLOL and give Read-Only access to non-owners of the SGA.

- Example 4:

```
PDS> INSTALL/LIBRARY:SYSRON/ACCESS:RW DK2:JOHN4
```

  Here a library SGA is given read-write access for nonowners.

- Example 5:

```
PDS> INSTALL/SYSTEM:SSS/INCREASE:2048 SYSTSK.TSK
```

  Here the amount of extra task virtual address space n, specified at link time by EXTTSK=n, is replaced by an allocation of 2048 words for this installed version. The task is installed as $$$SSS.

- Example 6:

```
PDS> INSTALL/REGION:MYREG CLC25
```

  Install the SGA CLC25 as a region called MYREG.

$JOB

FUNCTION

The $JOB command initiates a batch job.

REQUIRED PRIVILEGE

Not applicable.

(PR.MCR is required to issue the MCR qualifier).

FORMAT

$JOB *[/mode][/PASSWORD:password] username jobname timelimit*

parameter definitions

/mode

/MCR—Batch job contains MCR mode commands.

/mode

/DCL—Batch job contains DCL commands (used if the default mode is MCR).

password

Alphanumeric string 1 to 6 characters long that is your batch password.

username

Alphanumeric string 1 to 12 characters long that is unique to you. Your user name must be valid (such as the one you use in LOGIN).

jobname

Alphanumeric string 1 to 12 characters long that identifies the job. The system prints the job name at the beginning and end of the job's printed output.

timelimit

Time limit in minutes the batch job is to run. timelimit has a maximum value of 1440 (that is, 24 hours). If you omit this field, the job receives the installation default batch time limit (usually eight minutes).

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

The $JOB command must be the first command in a batch job command stream.

A batch log is created for every batch job run and is given the name LP.SPR. The batch log is one (or more) concatenated file that results from running a batch job. The log is automatically spooled to CL:. You can identify which job a batch log refers to by looking at the banner pages that include the job name and user name.

You must not specify /PASSWORD if there is no batch password associated with you.

You can change MCR or DCL mode within the batch job (using $MCR or $DCL). However, if a batch job contains mostly MCR mode commands, you can use the /MCR qualifier. For example:

```
$JOB/MCR SYSTEM WAGE 30
$MAC WAGE1,WAGE/-SP/CR=PRE1,WAGE1
$TKB @WAGE.CMD
$RUN WAGE
$EOJ
```

has the same effect as:

```
$JOB SYSTEM WAGE 30
$MCR
$MAC WAGE1,WAGE/-SP/CR=PRE1, WAGE1
$TKB @WAGE.CMD
$RUN WAGE
$EOJ
```

or:

```
$JOB SYSTEM WAGE 30
$MCR MAC WAGE1,WAGE/-SP/CR=PRE1, WAGE1
$MCR TKB @WAGE.CMD
$MCR RRT WAGE
$EOJ
```

EXAMPLES

- Example 1:

```
$JOB PIERCE JOBONE
```

- Example 2:

```
$JOB/PASSWORD:SECRET SYSTEM ACCOUNTS 30
```

LIBRARIAN

FUNCTION

The LIBRARIAN command enables you to create, delete and maintain object module libraries, MACRO-11 macro libraries, and UNIVERSAL libraries. The Technical Notes list operations you can perform with the LIBRARIAN command.

REQUIRED PRIVILEGE

PR.LIB

FORMAT

Compress Format:

PDS> [$]LIBRARIAN

OPERATION? COMPRESS *[/quals]*

LIBRARY? *libspec*

[NEW LIBRARY? *newlibspec]*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/SIZE:n	Size in 256 word blocks of the compressed file. Default is 100.
/EPT:n	Number of entries to allocate in the entry point table (not greater than 4096). A macro library has no entry point table, so n is set to 0 even if specifically defined. n is rounded up to the nearest multiple of 64. Defaults are 512 (object),0(macro).
/MNT:n	Number of entries to allocate in the module name table (not greater than 4096). n is rounded up to the nearest multiple of 64. Default is 256.

libspec

Specification of the library file to be compressed (no wildcards are allowed).

newlibspec
Specification of the compressed library file (no wildcards are allowed).

EXAMPLES

```
PDS> LIBRARIAN COMPRESS/SIZE:150
LIBRARY? PEEK.OLB [ESC]
NEW LIBRARY? PEEK2.OLB
```

The object library file PEEK.OLB is compressed to 150 blocks with 512 EPT entries and 256 MNT entries by default. The compressed file is called PEEK2.OLB.

FORMAT

Create Format:

PDS> [$]LIBRARIAN

OPERATION? CREATE *[/quals]*

LIBRARY? *libspec*

[FILE? *[infile-1,...infile-n]]*

parameter definitions

/quals
One of the following:

Qualifier	Explanation
/SIZE :n	Size in 256-word blocks of the library file to be created. Default is 100.
/EPT:n	Number of entries to allocate in the entry point table (not greater than 4096). A macro library has no entry point table. n is rounded up to the nearest multiple of 64. Defaults are 512 (object),0 (macro and UNIVERSAL).
/MNT:n	Number of entries to allocate in the module name table (not greater greater than 4096). n is rounded up to the nearest multiple of 64. Default is 256.
/TYPE:type	Type of library being created. type is either OBJECT, MACRO, or UNIVERSAL.
/SELECT	LINK command uses the file to define required global symbols at task build. (Object files only).
/SQUEEZE	Reduces the macro file by erasing all trailing blanks and tabs, blank lines and comments from the source text. (Macro files only).

Qualifier	Explanation
/NOENTRY_POINTS	Library modules are stored in the library, omitting definitions of entry point symbols.
FILETYPE_DEFAULT:type	Type is the 3-character default input file type, for the universal library created. If you do not specify this qualifier, the input file type for universal libraries is .ULB. The /FILETYPE_DEFAULT:type qualifier does not apply to object or module libraries.

libspec
Specification of the library file to be created (no wildcards allowed).

infile
Specification of a file to be input to the new library file. If you do not supply any input files, an empty library file is created as the qualifiers dictate.

EXAMPLES

- Example 1:

```
PDS> LIBRARIAN
OPERATION? CREATE/SI:200/EP:1024/MN:512/TYPE:OBJ
LIBRARY? MYLIB.OLB [ESC]
FILE? ONE.OBJ,TWO.OBJ,THREE.OBJ
```

  Creates an object library file named MYLIB.OLB with a size of 200 blocks with 1024 EPT entries and with 512 MNT entries, and inserts the three input files.

- Example 2:

```
PDS> LIBRARIAN
OPERATION? CREATE/TYPE:UNIVERSAL/FILETYPE:TXT
LIBRARY? CHRIS.ULB [ESC])
FILE? GARY, CLIVE
```

  Creates a universal library named CHRIS.ULB. By default the universal library CHRIS.ULB is 100 blocks long, with a module name table for 256 entries. Input file specifications for modules inserted or replaced in this library have a default file extension of .TXT. Inserts the modules in the input files GARY.TXT and CLIVE.TXT into the universal library CHRIS.ULB.

FORMAT

Delete Format:

PDS> [$]LIBRARIAN

OPERATION? DELETE *[/qual]*

LIBRARY? *libspec*

ENTRIES? *name1[,...namen]*

parameter definitions

/qual

One of the following:

Qualifier	Explanation
/MODULES	Deletes the specified module (the default qualifier).
/GLOBAL_SYMBOLS	Deletes the EPT entries specified.

libspec

Specification of the library file that contains the modules or entry points to be deleted.

name1,namen

Module name or the name of an entry in the entry point table.

EXAMPLES

- Example 1:

```
PDS> LIB DELETE/MODULES
LIBRARY? MYLIB.MLB
ENTRIES? NAMEA,NAMEB,NAMEC
```

  Deletes the macros NAMEA, NAMEB, and NAMEC from the macro library file MYLIB.MLB.

- Example 2:

```
$LIBRARIAN DELETE/GLOBAL MACLIB.OLB NAMEX
```

  Deletes the EPT entry named NAMEX contained in the library file MACLIB.OLB.

FORMAT

Extract Format:

PDS> [$]LIBRARIAN

OPERATION? EXTRACT/OUTPUT: *filespec*

LIBRARY? *libspec*

MODULES? *modulelist*

parameter definitions

filespec
File specification of the file to be created. If the output file does not have an explicit file type, the file type .MAC is assigned if the modules are extracted from a MACRO library, and .OBJ is assigned if from an object library.

modulelist
List of up to 8 modules to be extracted.

EXAMPLES

```
PDS> LIBR EXTR/OUT:AB MYLIB.MLB A B
```

This command causes the two modules A and B to be extracted from the Macro library MYLIB.MLB and placed in a single file called AB.MAC.

FORMAT

Insert Format:

PDS> [$]LIBRARIAN

OPERATION? INSERT *[/qual]*

LIBRARY? *libspec*

FILE? *[infile1[/qual2][,...infile n[/qual2]]*

parameter definitions

/qual
One of the following:

Qualifier	Explanation
/SELECT	LINK command uses the file to define required global symbols at Task Build time. (Object files only.)
/SQUEEZE	Reduces the macro file by eliminating all trailing blanks and tabs, blank lines and comments from the source text. (Macro files only).

Qualifier	Explanation
/NOENTRY_POINTS	Inserts modules without the definitions of the symbols that are entry points.

libspec
Specification of the library file where modules are to be inserted (no wildcards allowed).)

infile
Specification of a file to be inserted into libspec.

qual2
One of the following, but can only be specified when modules are to be inserted into a universal library:

Qualifier	Explanation
/MODULE:mod	mod specifies the module name (up to 6 Radix-50 characters). The default is the first 6 characters of the file name specified in infile.
/USER_INFORMATION: [(op:op:op:op)]	op specifies optional user descriptive information (up to 6 Radix-50 characters) to be stored in the module header. The default is null. If only part of the information set is specified, all preceding colons must be specified.

EXAMPLES

- Example 1:

```
PDS> LIBRA
OPERATION? INSERT/SQUEEZE
LIBRARY? MACLIB.MLB
FILE? ONE.MAC,TWO.MAC
```

  Inserts the modules contained in the files ONE.MAC and TWO.MAC into the library file name MACLIB.MLB, eliminating blanks and comments.

- Example 2:

```
$LIBRARIAN INSERT MYLIB.OLB MODULE.OBJ
```

  Inserts the file MODULE.OBJ into the library file named MYLIB.OLB.

- Example 3:

```
PDS> LIBRARIAN INSERT UNIV.ULB
FILE? CAROL.TXT/MODULE:FRANK/USER: (THIS IS:AUG15:TXT)
```

  Inserts the file CAROL.TXT into the universal library UNIV.ULB as FRANK (module name). THIS IS, AUG15, and TXT are stored in the module header.

FORMAT

List Format:

PDS> [$]LIBRARIAN

OPERATION? LIST *[/quals]*

LIBRARY? *libspec*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/OUTPUT:outfile	Send the output to the specified file.
/ENTRIES	Produce a directory of all modules and list the entry points for each.
/FULL	Produce a directory of all modules, giving full module descriptions: size, date of insertion and version.
/PRINT	Send the output to the line printer.

libspec

Specification of the library file to be listed (no wildcards allowed).

EXAMPLES

- Example 1:

```
PDS> LIBRARIAN LIST MYLIB.MLB
```

  Lists at your terminal a directory of all the modules contained in MYLIB.MLB.

- Example 2:

```
$LIBRARIAN LIST/FULL/PRINT MODLIB.OLB
```

  Lists at the line printer a directory of all the modules and their descriptions contained in the library file MODLIB.OLB.

FORMAT

Replace Format:

PDS> [$]LIBRARIAN

OPERATION? REPLACE *[/qual]*

LIBRARY? *libspec*

FILE? *infile1[/qual2][,...infilen[/qual2]]*

parameter definitions

/qual

One of the following:

Qualifier	Explanation
/SELECT	LINK command uses the file to define required global symbols at task build. (Object files only.)
/SQUEEZE	Reduces the macro file by eliminating all trailing blanks and tabs, blank lines and comments from the source text. (Macro files only.)
/NOENTRY_POINTS	Replaces modules, omitting definitions of symbols that are entry points.

libspec

Specification of the library file containing the modules to be replaced (no wildcards allowed).)

infile

Specification of a file containing the replacement modules (no wildcards allowed).)

qual2

One of the following, but can be specified only when modules are to be replaced in a universal library:

Qualifier	Explanation
/MODULE:mod	mod specifies the module name (up to 6 Radix-50 characters). The default is the first 6 characters of the file name specified in infile.
/USER_INFORMATION: [(op:op:op:op)]	op specifies optional user descriptive information (up to 6 Radix-50 characters) to be stored in the module header. The default is null. If only part of the information set is specified, all preceding colons must be supplied.

EXAMPLES

- Example 1:

```
PDS> LIBRARIAN
OPERATION? REPLACE
LIBRARY? MODLIB.OLB
FILE? NEWMOD.OBJ
```

  Replaces the module in the library file MODLIB.OLB with a module of the same name as that contained in NEWMOD.OBJ.

- Example 2:

```
$LIBRARIAN REPLACE OLDLIB.OLB ONELIB.OBJ,TWOLIB.OBJ
```

Replaces modules in the library OLDLIB.OLB with modules of the same name in the files ONELIB.OBJ and TWOLIB.OBJ.

- Example 3:

```
PDS> LIBRARIAN REPLACE
LIBRARY? BLUND
FILE? CHORL.TXT/USER_INFO:(THIS:IS:SEP4:UPDATE)
```

Replaces the module CHORL in the universal library BLUND with the updated module from the file CHORL.TXT. The optional user information THIS, IS, SEP4, and UPDATE, is stored in the module header. If the module name is not specified, the default file is taken as the file name CHORL.

FORMAT

Modify Header Format (Universal Libraries Only):

PDS> [$]LIBRARIAN

OPERATION? MODIFY *[/qual]*

LIBRARY? *libspec*

MODULE? *modname*

USER_INFORMATION? *(op:op:op:op)*

parameter definitions

/qual

One of the following:

Qualifier	Explanation
/HEADER	The only qualifier to the MODIFY operation and is, therefore, the default.
libspec	Specification of the universal library file containing the module.
modname	Name of the module whose descriptive information is to be specified. (This information is contained in the module header.)
(op:op:op:op)	Specifies the user information (up to 6 Radix-50 characters) to be stored in the module header. The default for each of the four fields is null, indicating that the corresponding field is not to be changed. Entering # as op clears the corresponding field.

EXAMPLES

```
PDS> LIBRARIAN
OPERATION? MODIFY
LIBRARY? BERT.ULB
MODULE? ACCNTS
USER INFORMATION? (#::FEB21)
```

Changes the optional descriptive information in the module header for module ACCNTS, in universal library BERT.ULB, to contain the following information:

default value	that is, null
no-change	that is, previous contents
FEB21	
no-change	that is, previous contents

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

Library Types

Libraries can be of the following type:

1 Those containing object modules (object module libraries).

2 Those containing macros (macro libraries).

3 Those containing any type of file (universal libraries).

Object module libraries have a default file type of .OLB. Each object module inserted into the library has its module name (taken from the .TITLE statement) added to the module name table (MNT) and its entry points (global symbols) added to the entry point table (EPT). task builder uses these libraries.

Macro libraries have a default file type of .MLB. Each macro inserted into the library has its module name (taken from the .MACRO statement) added to the module name table (MNT). MACRO uses these libraries.

Universal module names are derived from file names that exist when the module is inserted into the library. You can access a universal library module by using the $ULA routine (see the *IAS System Library Routines Reference Manual*).

Restrictions

The following restrictions apply to the handling of object modules:

1 The size of a module is limited to 65,536 words.

2 The size of the library file is limited to 65,536 words.

3 You must allocate the maximum anticipated size to tables and contiguous space. Expanding space allocations requires the COMPRESS operation to copy the entire file.

4 A fatal error results if an attempt is made to insert a module into a library that contains a differently named module with the same entry point.

Compress

The COMPRESS operation physically deletes modules that were logically deleted (by the DELETE operation) from the library specified. COMPRESS rearranges the file, putting all free space at the end of the library file, where it is available for the insertion of new modules.

Create

The CREATE operation allocates a contiguous library file on a direct access device (for example, a disk) and initializes the library header and tables.

Delete

The DELETE operation performs two delete operations:

1 It deletes modules, and all their associated entry points, from the library file tables MNT and EPT.

2 It deletes specified entries in the entry points table (EPT).

You can delete any number of modules in one DELETE operation. If no module of the specified name exists in the library, DELETE has no effect on the library. A deleted module is marked as deleted but remains physically in the file until a COMPRESS operation is performed.

Extract

The EXTRACT operation extracts modules from a library and generates a new file that is the concatenation of the named modules. You can extract up to eight modules at any one time. The original library remains unaltered.

Insert

The INSERT operation inserts modules into the specified library file. You can have any number of input files, and any of these can contain concatenated object modules.

List

The LIST operation has the following functions:

- Causes a library file directory to print on your terminal by default.
- Sends a library file directory to an output file.

The operation qualifier also determines the amount of detail contained in the directory. By default, the directory lists all the modules in the library.

Replace

The REPLACE operation replaces old modules in the library with new modules of the same name. That is, a new module that has the same name as a module already contained in the library replaces the existing module. The old module remains physically in the file until it is compressed.

Modify Header

This modifies the optional user-specified information stored in the header of universal library modules.

LINK

FUNCTION

The LINK command links object files, compiled or assembled modules, to form an executable task and produces output as directed by command qualifiers.

The *IAS task builder Reference Manual* describes the task builder procedures and options in full.

REQUIRED PRIVILEGE

PR.LIN

FORMAT

PDS> [$]LINK *[/quals]*

FILE? *infile1[/filequal][,...,infilen]*

parameter definitions

/quals

One of the following:

Qualifier[1]	Default
/ABORT /NOABORT	/ABORT
/CHECKPOINT /NOCHECKPOINT	/CHECKPOINT
/CROSS_REFERENCE /NOCROSS_REFERENCE	/NOCROSS_REFERENCE
/DEBUG[:filespec] /NODEBUG	/NODEBUG
/DEFAULT_LIBRARY:f-s /NODEFAULT_LIBRARY	/NODEFAULT_LIB:LB:[1,1]SYSLIB.OLB
/DISABLE /NODISABLE	/DISABLE

[1]Each qualifier is described in the Technical Notes under Command Qualifiers.

Qualifier[1]	Default
/EXIT:n /NOEXIT:n	/EXIT:1
/FIX /NOFIX	/NOFIX
/FLOATING_POINT /NOFLOATING_POINT	/FLOATING_POINT
/FLUSH_RECEIVE_QUEUES /NOFLUSH_RECEIVE_QUEUES	/FLUSH_RECEIVE_QUEUES
/FULL_SEARCH /NOFULL_SEARCH	/NOFULL_SEARCH
/HEADER /NOHEADER	/HEADER
/LARGE_SYMBOL_TABLE /NOLARGE_SYMBOL_TABLE	/NOLARGE_SYMBOL_TABLE
/MAP[:filespec] /NOMAP	/NOMAP
/MAP:(filespec/qualifier) /NOMAP	/MAP:(filespec/WIDE)
/MULTIUSER /NOMULTIUSER	NOMULTIUSER
/OPTIONS /NOOPTIONS	/NOOPTIONS
/OVERLAY_DESCRIPTION:filespec /NOOVERLAY_DESCRIPTION	/NOOVERLAY_DESCRIPTION
/POSITION_INDEPENDENT /NOPOSITION_INDEPENDENT	/NOPOSITION_INDEPENDENT
/PRIVILEGED /NOPRIFILEGED	/NOPRIVILEGED
/READ_WRITE /NOREAD_WRITE	/NOREAD_WRITE
/RECEIVE /NORECEIVE	/RECEIVE
/REQUEST /NOREQUEST	/NOREQUEST
/RESIDENT_OVERLAYS /NORESIDENT_OVERLAYS	/NORESIDENT_OVERLAYS
/RUN_TIME_SYSTEM /NORUN_TIME_SYSTEM	/NORUN_TIME_SYSTEM
/SEQUENTIAL /NOSEQUENTIAL	/NOSEQUENTIAL
/SYMBOLS[:filespec] /NOSYMBOLS	/NOSYMBOLS
/SYMBOLS:(filespec/qual) /NOSYMBOLS	/SYM:(fs/UNDEFINED_SYMBOL)

[1]Each qualifier is described in the Technical Notes under Command Qualifiers.

Qualifier[1]	Default
/TASK[:filespec] /NOTASK	/TASK
/TRACE /NOTRACE	/NOTRACE
/WAIT_FOR_NODES /NOWAIT_FOR_NODES	/WAIT_FOR_NODES

[1]Each qualifier is described in the Technical Notes under Command Qualifiers.

infile

Specification of an input file. See Input Files below for further information. Wildcards are not allowed. (You must not include this parameter if you specify the command qualifier /OVERLAY.)

/filequal

See File Qualifiers for a description of each file qualifier.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

Input Files

You can specify input files to the LINK command in one of two ways:

1 In a list of file specifications as a parameter to the command.

2 From within an overlay description file by means of the /OVERLAY command qualifier.

If you use the /OVERLAY qualifier to specify the input files, you must not specify them as a command parameter (see item 1 above).

The input files can consist of the following elements:

1 Single object modules

2 Concatenated object modules

3 Object module libraries

4 Symbol table files

You must use file qualifiers to identify concatenated module files and library files (see File Qualifiers below). In addition, the /SELECT qualifier can modify symbol table files; the task builder then uses the modified file only to resolve required symbol definitions.

The task builder provides default file types in the following cases. When specifying single or concatenated object modules, you can omit the file type field. The task builder then assumes the file type to be .OBJ. You can also omit the file type field of a library file (a file modified by the /LIBRARY qualifier) in which case the task builder assumes the file type to be .OLB.

Input symbol table file specifications have no default file type. Thus, you must specify the file type. However, symbol table files output from the task builder have the default file type .STB.

Wildcards are not allowed for any type of file specification supplied with LINK.

File Qualifiers

The following list defines all the available file qualifiers:

File Qualifier	Description
/CONCATENATED	Identifies the file as a concatenated object file.
/LIBRARY	Identifies the file as an object module library file.
/LIBRARY:[(]mod-1.....mod-n[)]	Identifies the file as an object module library file where mod-1...mon-n are entries within that library and instructs the task builder to take only the modules named.
/NOCONCATENATED	Instructs the task builder to take only the first module in the file. If it is a concatenated object module file, subsequent modules are ignored.
/SELECT	Instructs the task builder to take only required global symbol definitions from the file. The modified file can be any object file, but it is normally a symbol file.
/MAP	Includes input file in the memory allocation map. The default is /MAP for object module files, and /NOMAP for system library files and SGAs.

Command Qualifiers

You can negate all the command qualifiers in this section by using the prefix NO. For example, the qualifier /TASK instructs the Task Builder to create a task file; whereas the qualifier /NOTASK requests the task builder not to produce a task image file.

Table 14–12 Command Qualifiers and Defaults

Qualifier	Default	Explanation
/ABORT	/ABORT	The task can be aborted.
/CHECKPOINT	/CHECKPOINT	The task can be checkpointed.
/CROSS_REFERENCE	/NOCROSS_REFERENCE	Append a global symbol cross-reference to the end of the memory allocation map.
/DEBUG[:filespec]	/NODEBUG	If filespec is not given, link the task with the system's debugging aid ([1,1]ODT.OBJ). If filespec is given, link the task with the debugging aid contained in the specified file. The debugging aid must be in object format.

Table 14–12 (Cont.) Command Qualifiers and Defaults

Qualifier	Default	Explanation
/DEFAULT_LIBRARY:file-spec	/NODEFAULT_LIBRARY:LB:[1,1]SYSLIB.OLB	Use the named object module library instead of current system library file LB:[1,1]SYSLIB.OLB.
/DISABLE	/DISABLE	The task can be disabled.
/EXIT:[n]	/EXIT:1	Task builder stops executing after n (decimal) errors. If you do not specify /EXIT, the task builder continues to find all errors. If you specify /EXIT without a value, the task builder finishes after one error.
/FIX	/NOFIX	The task can be fixed in memory.
/FLOATING_POINT	/FLOATING_POINT	The task uses the floating point processor.
/FLUSH_RECEIVE_QUEUES	/FLUSH_RECEIVE_QUEUES	The task is to have its receive queues (data and references) flushed each time it exits.
/FULL_SEARCH	/NOFULL_SEARCH	This controls the symbol table search for matching definition or references in overlaid tasks having co-trees.
/HEADER	/HEADER	The task includes a header. Use /NOHEADER to produce a non-executable task image (for example, a shareable global area—resident library, common area, or region).
/LARGE_SYMBOL_TABLE	/NOLARGE_SYMBOL_TABLE	Select a version of the task builder that has a large internal symbol table. (Considerably slower than the default task builder.)
/MAP[:filespec] or /MAP[:(filespec/qualifier)]	/NOMAP	Produce a memory allocation map. If you do not specify filespec after /MAP, the map file is sent to the line printer. If you specify filespec, you can omit the file type field, in which case the task builder assumes it to be .MAP. If you qualify the map filespec then you must enclose the filespec and file qualifiers in parentheses, for example: /MAP:(MYMAP/SHORT) You can attach the following qualifiers to the map filespec: /FULL = Includes all modules in map /FILES = Includes file-by-file breakdown /NARROW = Makes map in 72-column format /SHORT = Makes only summary of map /WIDE = Makes map in 132-column format /NOUNDEFINED_REFERENCES = Do not print undefined references on TI: **NOTE:** defaults are: /NOFULL/WIDE/SHORT /NOFILES/UNDEFINED_REFERENCES

Table 14–12 (Cont.) Command Qualifiers and Defaults

Qualifier	Default	Explanation
/MULTIUSER	/NOMULTIUSER	Build a multiuser task so that more than one copy of the task can be run at one time.
/OPTIONS	/NOOPTIONS	Apply task builder options specified after the command string (see the section Task Builder Options). In interactive mode, the /OPTIONS qualifier causes the Task Builder to prompt OPTIONS? after the input files have been specified. For example: PDS> LINK/OPTIONS FILE? PROG, REPORT OPTIONS? You then enter the options described in the list below. A slash (/) as the first character in a line terminates the list of options and the task builder begins executing. See the *IAS task builder Reference Manual* for details of individual option syntax. For example: PDS> LINK/OPTIONS FILE? MAIN.OBJ, PROG.OBJ OPTIONS? ACTFIL=8 OPTIONS? MAXBUF=160 OPTIONS? UNITS=9 OPTIONS? ASG=DT1:7:8:9 OPTIONS? / In batch mode, if the command qualifier list includes /OPTIONS, you must specify at least one option. Specify options on lines immediately following the command string. A line containing a slash (/) in the first character position terminates the list of options. If /NOOPTIONS is specified (default) the system automatically includes the option SGA=SYSRES:RO, that is, the task is mapped onto the system library SYSRES.
/OVERLAY_DESCRIPTION: filespec	/NOOVERLAY_DESCRIPTION	Link the task according to the overlay structure defined in the given file, the name of which you must include with the /OVERLAY_DESCRIPTION qualifier. If you omit the file type field of filespec, the task builder assumes it to be .ODL. You specify the input files to LINK within the overlay description file; therefore, you must not specify the input file parameter list. See the *IAS task builder Reference Manual* for details of ODL files.
/POSITION_INDEPENDENT	/NOPOSITION_INDEPENDENT	The task code is position independent.
/PRIVILEGED	/NOPRIVILEGED	The task is executive-privileged.

Table 14–12 (Cont.) Command Qualifiers and Defaults

Qualifier	Default	Explanation
/READ_WRITE	/NOREAD_WRITE	Gives Read/Write access to the Read-Only code.
/RECEIVE	/RECEIVE	The task is able to receive data sent to it by the SEND DATA and SEND BY REFERENCE directives.
/REQUEST	/NOREQUEST	Data can be sent to the task and can be requested or resumed by a non-directive-privileged user.
/RESIDENT_OVERLAYS	/NORESIDENT_OVERLAYS	The task is to be built with resident overlays.
/RUN_TIME_SYSTEM	/NORUN_TIME_SYSTEM	The task includes the overlay run time system and its associated control area if it is overlaid.
/SEQUENTIAL	/NOSEQUENTIAL	Program sections within the task are to be linked in the order they appear. Otherwise, they are linked in alphabetical order.
/SYMBOLS[:filespec] or /SYMBOLS:(filespec/NOUNDEFINED_SYMBOLS)	/NOSYMBOLS Default filespec qualifier: /UNDEFINED_SYMBOLS	Produces a symbol table file. Unless you specify filespec, the symbol table file takes the name of the first input file, except that the file type is .STB. If you specify filespec, you can omit the file type field; in which case, the task builder assumes it to be .STB. If you qualify the STB filespec then you must enclose the filespec and qualifier in parentheses. For example: /SYMBOL:(MYPROG/NOUNDEFINED) You can supply the following qualifier to the STB filespec: /NOUNDEFINED_SYMBOLS If you use this qualifier, undefined symbols are ignored. The default is /UNDEFINED_SYMBOLS.
/TASK[:filespec]	/TASK	Produces a task image file. Unless you specify filespec, the task file takes the name of the first input file (or the name of the overlay descriptor file) except that the file type is .TSK. If you specify filespec, you can omit the file type field, in which case the task builder assumes it to be .TSK.
/TRACE	/NOTRACE	The task is traceable.

Table 14–12 (Cont.) Command Qualifiers and Defaults

Qualifier	Default	Explanation
/WAIT_FOR_NODES	/WAIT_FOR_NODES	Tasks can use certain directives that require data from buffers. Normally, if no nodes are available, the directive fails and an error message is returned. The /WAIT_FOR_NODES qualifier enables the directive (and hence the task) to attempt to get a node a number of times before it fails.

Task Builder Options

Table 14–13 lists the task builder options.

Table 14–13 Task Builder Options

Option	Meaning	Interest[1]
ABSPAT	Declares absolute patch values.	M
ACTFIL	Declares number of files open simultaneously.	FM
ASG	Declares device assignment to logical units.	FM
ARTG	Declares the number of attachment descriptor blocks to be created in the task header.	FM
BASE	Defines lowest virtual address.	FM
EXTSCT	Declares extension of a program section.	FM
EXTTSK	Extends task memory allocation at install time.	FM
FMTBUF	Declares extension of buffer used for processing format strings at run time.	F
	In CORAL, set to blkmax*8, where blkmax is the maximum number of LUNs used for concurrent asynchronous block I/O at any one time.	C
GBLDEF	Declares a global symbol definition.	M
GBLPAT	Declares patch values relative to a global symbol.	MC
GBLREF	Declares a global symbol reference.	FM
MAXBUF	In FORTRAN, declares an extension to the FORTRAN record buffer.	F
	In CORAL, set to strmax*140 (decimal), where strmax is the maximum number of LUNs associated with stream I/O at any one time.	C
MAXEXT	Declares maximum extendable task size.	FMC
ODTV	Declares the address and size of the debugging aid SST vector.	M
PAR	Declares partition name and dimensions.	FM
POOL	Declares pool usage limit.	FM
PRI	Declares priority.	FM

[1]F indicates FORTRAN, MACRO, and CORAL; M indicates FORTRAN and MACRO; C indicates CORAL.

Table 14–13 (Cont.) Task Builder Options

Option	Meaning	Interest[1]
RESAPR	Reserves APRs for use by memory management directives	FM
RESSGA	Declares task's intention to access an SGA.	FM
SGA	Declares task's intention to access an SGA.	FM
STACK	Declares the size of the task's stack.	FM
SYMPAT	Declares a patch using task symbols.	M
TASK	Declares the default installed name of the task.	FM
TOP	Defines highest virtual address.	FM
TSKV	Declares the address of the task SST vector.	M
UIC	Declares the user identification code under which the task runs.	FM
UNITS	Declares the maximum number of logical units.	FM
VSECT	Declares the virtual base address and size of a program section.	FM

[1]F indicates FORTRAN, MACRO, and CORAL; M indicates FORTRAN and MACRO; C indicates CORAL.

NOTE:

1 **The SGA option supersedes the COMMON and LIBR options in previous versions of IAS. COMMON and LIBR are still recognized by the Task Builder for compatibility. Their effect is identical to specifying SGA.**

2 **The RESSGA option supersedes the RESCOM and RESLIB options in previous versions of IAS. RESCOM and RESLIB are still recognized by the task builder for compatibility. Their effect is identical to specifying RESSGA.**

EXAMPLES

- **Example 1:**

```
$LINK/OPTIONS/PRIVILEGED A.OBJ/CONCATENATED
UNITS=9
/
```

- **Example 2:**

```
PDS> LINK/OVERLAY:STRUCTURE/MAP:ROUTE
```

  The system does not prompt FILE? as /OVERLAY has been specified.

- **Example 3:**

```
PDS> LINK/DEFAULT_LIBR:DK1:[1,1]SYSLIB
FILE? A.OBJ,B.OBJ
```

LOGOUT

FUNCTION

The LOGOUT command terminates your interactive session and on a full timesharing system releases any devices and mounted volumes allocated to you.

REQUIRED PRIVILEGE

Not applicable.

FORMAT

PDS> LOGOUT *[/HOLD]*

parameter definitions

/HOLD

If the terminal is connected by a dialup line, this line is not disconnected when you log out. When you next want to log in, you need not dial in again.

The default is that the line is disconnected when you log out.

The LOGOUT command has no parameters.

COMMAND VARIATIONS

On a multiuser system, mounted volumes are not released on logout.

TECHNICAL NOTES

Provided QUIET mode has not been set, the following information is printed when you log out:

1 The volumes and devices dismounted and deallocated (timesharing only).

2 Your user name, UIC, terminal number, and Job-id.

3 The logout time.

4 The connect time.

5 CPU utilization.

For QUIET mode, see the PDS command SET QUIET.

If you are using the SET PRINTING DEFERRED command, any spooled files generated from tasks run from your terminal are printed when you log out.

The message BYE appears and indicates that the terminal is inactive. You must type Ctrl/C to reactivate the terminal for another session.

EXAMPLES

```
PDS> LOGOUT

USER CAROL UIC [200,60]TT03: 11:19:30 15-MAY-78
CONNECT TIME 36M SYSTEM UTILIZATION 4 MCTS

BYE
```

MACRO

FUNCTION

The MACRO command assembles one or more ASCII source files containing MACRO-11 statements into a single relocatable binary object file. The output optionally consists of a binary object file, an assembly listing, a cross-reference listing, and the symbol table listing.

REQUIRED PRIVILEGE

PR.MAC

FORMAT

PDS> [$]MACRO *[/quals1]*

FILE? *filespec[/quals2][+...]*

parameter definitions

/quals1

One of the following:

Qualifier	Explanation
/OBJECT[:filespec]	Produces an object file (the default condition), named according to the filespec supplied (wildcards are not allowed). Otherwise the file is named by default (see Defaults below).
/NOOBJECT	Does not produce an object file.
/LIST[:filespec]	Produces a listing file (the default is /NOLIST), named according to the filespec supplied (wildcards are not allowed). Otherwise the file is named by default (see Defaults below).
/NOLIST	Does not produce a listing file.
/CROSS_REFERENCE [:(keyword-list)]	Produces a cross-reference listing, where keyword-list is any of the following: USER_SYMBOLS—Shows user-defined symbols (default) MACRO_SYMBOLS—Shows macro symbols (default) REGISTER_SYMBOLS—Shows register symbols PERMANENT_SYMBOLS—Shows permanent symbols

Qualifier	Explanation
/NOCROSS_REFERENCE	Does not produce a cross-reference listing (default).
/SWITCHES:(swlist)	Uses the list of switches swlist to control the contents or format of the output files. See MACRO Switches, below.
filespec	Specification of a file that contains MACRO source code. Multiple input file specifications must be concatenated with a plus sign (+). No wildcards are allowed. Specifications must include a file name. If you omit the file type, the system assumes it is .MAC.

/quals2

One of the following:

Qualifier	Explanation
/LIBRARY	Indicates that the file is a macro library file. You must specify a user macro library file in the command line prior to the source files that reference the library.
/PASS:n	Assembles the associated file during assembly pass 1 or assembly pass 2 only (where n equals 1 or 2).
/SWITCHES:(swlist)	Uses the list of switches swlist to control the contents or format of the input files. See the section MACRO Switches below.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

Defaults

Object File—By default the assembler produces an object file with the name of the last source file specified and .OBJ as the file type. If you define a filespec without a file type, then .OBJ is assumed.

Listing File—If you specify /LIST with no filespec, a listing file is sent to the line printer. If you define a filespec without a file type then .LST is assumed.

MACRO Switches

The following MACRO switches are available by means of the /SWITCHES:(swlist) command qualifier or file qualifier:

Switch	Explanation
/DS	Disable. Equivalent to the .DISABL directive in the source program.
/EN	Enable. Equivalent to the .ENABL directive in the source program.
/LI	List. Equivalent to the .LIST directive in the source program.
/NL	No List. Equivalent to the .NLIST directive in the source program.

/DS and /EN apply to the object file specification and the source file specifications, and you can specify them as command qualifiers or file qualifiers. You can follow /DS and /EN by the switch values shown in Table 14–14. Colons separate the switches from the values, and colons separate the values from each other (:). For more detailed information, see the *PDP-11 MACRO-11 Reference Manual*.

Table 14–14 Values for MACRO Switches /DS and /EN

Value	Default	Meaning
ABS	Disable	Produces absolute binary output in Files-11 format.
AMA	Disable	Assembles all relative addresses as absolute addresses.
CDR	Disable	Treats source columns 73 and greater as a comment.
CRF	Enable	Produces cross-reference output.
FPT	Disable	Causes floating-point truncation.
LC	Disable	Accepts lower case ASCII input.
LSB	Disable	Permits the enabling or disabling of a local symbol block.
PNC	Enable	Produces binary output. Disabling this function inhibits binary output until a .ENABL PNC statement is encountered within the same module.
REG	Enable	Applies normal MACRO-11 default register definitions.
GBL	Enable	Treats all symbol references as default global references.

/LI and /NL apply to the listing file specification, and you specify them as command qualifiers only. You can follow /LI and /NL by the switch values shown in Table 14–15. Colons separate the switches from the values, and colons separate the values from each other (:).

Table 14–15 Values for MACRO Switches /LI and /NL

Value	Default	Meaning
BEX	List	Binary extensions
BIN	List	Generated binary code
CND	List	Unsatisfied conditional coding
COM	List	Comments
LD	No list	Listing directives that alter the listing level count
LOC	List	Current location counter field

Table 14–15 (Cont.) Values for MACRO Switches /LI and /NL

Value	Default	Meaning
MC	List	Macro calls and repeat expansions
MD	List	Macro definitions and repeat expansions
ME	No list	All macro expansions
MEB	No list	Only macro expansions that generate binary code
SEQ	List	Sequence numbers of source lines
SRC	List	Source lines
SYM	List	Symbol table of assembled source program
TOC	List	Table of contents during assembly pass 1
TTM	List	Listing output format: /LI:TTM 80-column output /NL:TTM 132-column output Default: installation-dependent

For further information on the use of MACRO-11, refer to the *PDP-11 MACRO-11 Reference Manual*.

EXAMPLES

- Example 1:

```
PDS> MACRO
FILE? A.MAC+B.MAC;3
```

  Assembles the source files A.MAC and B.MAC;3 to produce an object file named B.OBJ.

- Example 2:

```
$MACRO/NOLIST FILEA
```

  Assembles the source file FILEA to produce an object file named FILEA.OBJ. No listing file is produced.

- Example 3:

```
PDS> MAC/OBJ:C D.MAC+E.MAC
```

  Assembles the source files D.MAC and E.MAC to produce an object file named C.OBJ.

- Example 4:

```
PDS> MAC MYFILE.MAC/PA:1
```

  Assembles the source file MYFILE.MAC during assembly pass 1 only.

- Example 5:

```
PDS> MAC/LIST MACLIB.MLB/LIB+MYFILE
```

  Assembles the source file MYFILE and the macro library file MACLIB.MLB. A listing file is produced on the line printer.

- Example 6:

```
PDS> MAC/SW:(/DS:LSB/LI:ME) TEST.MAC
```

  Assembles the source file TEST.MAC to produce an object file TEST.OBJ. Local symbol blocks are disabled, and macro expansions are listed.

- Example 7:

```
PDS> MAC/LI:FILE/SW:(/LI:TTM) TEST
```

  Assembles the source file TEST to produce an object file TEST.OBJ. A listing file FILE.LST is produced with 80-column output.

- Example 8:

```
PDS> MAC/CROSS:(REGISTER_SYMBOLS) TEST.MAC
```

  Assembles the source file TEST.MAC to produce an object file TEST.OBJ. A cross-reference listing is also produced, showing cross-references to register symbols in the source program.

- Example 9:

```
PDS> MAC/NOOBJ/SW:(/DS:LSB) TEST.MAC
```

  This example gives the error message:

```
QUALIFIER VALUE INVALID HERE
```

  because the switch /DS:LSB applies to object file specifications, and in this example no object file is produced.

- Example 10:

```
PDS> MAC FRED/SW:(/DS:GBL)
```

  Assembles the source file FRED to produce an object file FRED.OBJ. All undefined symbol references are flagged with an error code (U) in the assembly listing.

MCR

FUNCTION

The MCR command enables you to enter MCR mode. See Chapter 5 of the *IAS MCR User's Guide* for a description of MCR mode.

REQUIRED PRIVILEGE

PR.MCR

FORMAT

Format 1:

PDS> [$]MCR

PDS>>

Remains in MCR mode until you issue the DCL command.

Format 2:

PDS> [$]MCR *[MCR command string]*

parameter definitions

MCR command string

A valid MCR command. This has to be less than or equal to 79 characters.

This format enables you to issue a single MCR command without entering MCR mode.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

The prompt PDS>> indicates MCR mode. You can enter DCL commands in MCR mode. If the DCL command has the same name as an MCR command, for example, DIS (DISMOUNT and DISABLE) and SET, the command is taken as an MCR command. Take care when using DCL commands in MCR mode.

All MCR mode commands are interruptable except INS, MOU, and DMO.

MCR mode commands terminated with an ESC suppress the PDS prompt, which you can reactivate by typing Ctrl/C.

EXAMPLES

- Exampale 1:

```
PDS> MCR
PDS>> PIP DK1:/LI
PDS>>
```

- Example 2:

```
PDS> MCR TKB @MYBUILD.CMD
PDS>
```

- Example 2:

```
PDS> MCR
PDS>> SHOW STATUS
    .
    .
    .
PDS>>
```

MERGE

FUNCTION

The MERGE command merges records from a sequential, indexed or relative file (the transaction file) with an indexed or relative file (the target file).

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> [$]MERGE[/LOG[:filespec]]

FILE? *transfile/[/qual1]*

INTO? *targetfile/qual2*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/LOG	If specified, sends an error log to filespec or by default to your terminal. The log gives details of records that cannot be merged. If you specify a filespec, you must include a file name.
transfile	Specification of the file to be merged. You must specify the file name and file type.

/qual1

One of the following:

Qualifier	Explanation
/SEQUENTIAL	Transaction file is sequential (the default).
/INDEXED [/KEY:NUMBER:n]	Transaction file is an indexed sequential (ISAM) file. The order of record extraction can be specified by the /KEY qualifier and key number. Default is /KEY:NUMBER:1 (the primary key). You can omit /INDEXED if you specify /KEY:NUMBER:n.
/RELATIVE	Specifies a relative structured file.

targetfile
Target file specification. You must specify the file name and file type.

/qual2
Must be specified and is one of the following:

/INDEXED
/RELATIVE

COMMAND VARIATIONS

Not applicable.

EXAMPLES

```
PDS> MERGE/LOG:FRED.LOG ALF.DAT SID.DAT/INDEX
```

MESSAGE

FUNCTION

The MESSAGE command enables you to send a message to another terminal or terminals.

REQUIRED PRIVILEGE

ANY

FORMAT

PDS> [$]MESSAGE *[/quals]*

MESSAGE? *message*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/ALL	Sends the message to all timesharing terminals.
/ACTIVE	Sends the message to all active timesharing terminals.
/CLI:cliname	Sends the message to the terminals allocated to the CLI, specified as the three character cliname.
/USER:username	Sends the message to the terminals logged in as the specified user.
/TERMINAL:list	Sends the message to the terminals specified in list. The list of terminals can have the form: TTn
/OPERATOR	Sends the message to the system console (default).

message

String of 1 to 65 characters terminated by RET. In batch mode, message is a string written on the same line as the $MESSAGE command.

COMMAND VARIATIONS

The qualifiers /ACTIVE, /CLI:cliname, and /USER:username are not available on multiuser systems.

EXAMPLES

- Example 1:

```
$MESSAGE/USER:ARTHUR THIS JOB WILL REQUIRE 2 TAPE DRIVES
```

- Example 2:

```
PDS> MESSAGE/OPERATOR SWITCH ON LINE PRINTER
```

- Example 3:

```
PDS> MESSAGE/TERMINAL:(TT1,TT2,TT3)
MESSAGE? PLEASE DISMOUNT DK0: IF YOU HAVE FINISHED WITH IT.
```

MOUNT

FUNCTION

The MOUNT command makes a volume available to you, and optionally associates a logical name with the volume.

REQUIRED PRIVILEGE

PR.DEV

FORMAT

PDS> [$]MOUNT *[/quals]*

DEVICE? *devicename*

VOLUME-ID? *volumeident*

[LOGICAL NAME? *logicalname]*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
devicename	Device name or logical name of the device where you are to mount the volume. You can omit the device unit number, except when you use the /NOOPERATOR qualifier or when the device name is a logical name. The system does not prompt for a logical name if you mount the device using the logical name you assigned to it using an ALLOCATE command.

Qualifier	Explanation
volumeident	If you are mounting the volume as FOREIGN or if you use the /OVERRID:VOLUME qualifier, the name you supply here identifies the volume for handling by the operator, (for example, a label written on the volume container). For disk and DECtape, the volume identification is 1 to 12 characters long. For ANSI labelled magnetic tape, the identification (ANSI label) is 1 to 6 characters long. For magtape only, you can replace volume identification with volume label list. This enables you to specify multiple volume names for multivolume magtapes.
logicalname	Logical name to be associated with the physical device.

COMMAND VARIATIONS

On a multiuser system, any mounted device is available for all users; that is, no device management or device protection is enforced by the system. No operator intervention is required—/NOOP is assumed and cannot be overriden. When mounting a device on a multiuser system, you must ensure that the volume is physically loaded on the device before issuing the MOUNT request. In this case, the /GLOBAL, /REALTIME, /NOSHARE, /NOOPERATOR, /DEVICES, and /NOWRITE qualifiers are illegal. You must always specify a unit number when mounting a volume on a multiuser system.

To mount a multivolume magtape set on a multiuser system, use the following format:

```
PDS> MOUNT[/qual] (devl:....devn:) (volidl:....volidn)
```

Where:

devl:....devn:	Devices to be assigned to the multivolume magtape set. If you specify only one device, you can omit the parentheses. You must, however, specify the unit number.
volid1:....volidn	List of volume labels that constitute the magtape volume set.
/qual	Any of the valid, magtape command qualifiers listed in Command Qualifiers, below.

TECHNICAL NOTES

1 You obtain exclusive access to magnetic tape volumes, and to any volumes mounted as foreign. You can share Files-11 and DECtape volumes, that is, once the volume has been mounted, other users can also MOUNT and use it unless you specify /NOSHARE.

2 You can omit the unit number from the device specification (timesharing only). The operator can then select the appropriate unit.

You can qualify the MOUNT command only in the following circumstances:

a. when a specified Files-11 disk or DECtape volume is not already mounted in the system.

b. when you mount a magnetic tape or foreign volume.

Command Qualifiers

Table 14–16 lists the command qualifiers.

The system rejects the mount if you specify the command qualifiers in a command that tries to mount a previously mounted Files-11 disk or DECtape.

Table 14–16 MOUNT Command Qualifiers

Qualifier	Description
/ACCESSED:n[1]	Number (octal) of preaccessed directories to be kept (Files-11 disk and DECtape only). See the *IAS Performance and Tuning Guide* for a description of preaccessed directories.
/CONTROL_FUNCTIONS	Enable logical and positioning operations on the volume.
/DENSITY:n[1]	Set magnetic tape density, where n = 800 or 1600.
/DEVICES:n	Allocate the stated number of device units for a multivolume magnetic tape unit set, you can not specify a unit number on the device_name parameter in this case.
/EXTENSION:n[1]	Set default file extension to n (octal) blocks. For further information, see the *IAS Performance and Tuning Guide*.
/FILE_PROTECTION:(code)[1]	Overrides default file protection code to be given to new files. See Section 6.2.2.
/FOREIGN	The volume to be mounted is not to be considered as a Files-11 structured volume and cannot be shared by other users. The default is Files-11 format. You can not specify this qualifier with /GLOBAL.
/GLOBAL	The volume mounted on the device is designated as global. This is a volume characteristic. The system considers the volume mounted. However, a timesharing user must then explicitly MOUNT this volume in order to access it. The system operator is not requested to unload the volume unless an explicit DISMOUNT/GLOBAL command is issued (provided no users currently have it mounted). A volume that is mounted globally reserves the device even if no users have it currently mounted. **NOTE:** Use this qualifier only when you mount a shareable volume. Do not use it when you mount a magtape.
/NOOPERATOR	Mounts without operator intervention. You must specify the device number and you should ensure that the required volume is physically loaded on the specified device.
/NOSHARE	Mounts a Files-11 volume for exclusive use.
/NOWRITE	Write protected. The system operator is requested to physically write protect the volume when it is loaded. The default is write permitted. This qualifier has no effect if you specify /NOOP, you must make sure that you enforce hardware write protect.
/OVERRIDE:(items)	Where items are one or more of the following separated by commas and spaces. You can omit parentheses if you only specify one item. EXPIRATION_DATE—Enables you to overwrite an unexpired magnetic tape volume. IDENTIFICATION—Enables you to override the volume identification. As a result, you can mount a volume without specifying its volume identification. SET_IDENTIFICATION—Enables you to process tapes with inconsistent file set identifiers. VOLUME_IDENTIFICATIONEnables you to override the volume identification. As a result, you can mount a volume without specifying its volume identification.

[1]This qualifier enables the first user to override parameters set when the volume was initialized.

Table 14–16 (Cont.) MOUNT Command Qualifiers

Qualifier	Description
/PROCESSOR:ACPtask[1]	Specifies the ancillary control processor (ACP) to be used for processing file accesses to the volume. The ACP specified by this qualifier overrides the default ACP.
/PROTECTION:(code)[1]	Replaces volume protection with code specified. See Section 6.2.2.
/REALTIME	Mounts volume for access by real-time tasks only. You must explicitly MOUNT (timesharing) the volume in order to access it.
/UNLOCKED	Leaves index file unlocked (Files-11 disk and DECtape only). The default is to leave index file locked.
/WINDOW:n[1]	n is the number (octal) of retrieval pointers to be kept in each window block for each open file on the volume. Increasing this number speeds access, especially to randomly accessed files, at the expense of system dynamic memory. See the *IAS Performance and Tuning Guide* for further information. You set the volume default when initializing the volume.

[1]This qualifier enables the first user to override parameters set when the volume was initialized.

EXAMPLES

- **Example 1:**

```
PDS> MOUNT
DEVICE? DT2:
VOLUME-ID? RISE RETURN
```

- **Example 2:**

```
$MOUNT/FOREIGN MT0: TESTER CF0:
```

- **Example 3:**

```
PDS> MOU DK:
VOLUME-ID? SAM AL1:
```

- **Example 4:**

```
$MOUNT/DEN:800/NOOPER MT0: VOL163 TA0:
```

- **Example 5:**

```
PDS> ALLOC/DEVICE
DEVICE? DT ESC
LOGICAL-NAME? XX0
PDS> MOU/FORXX0 DOSVOL2
```

- **Example 6:**

```
$MOUNT/DEVICES:4 MM (VOL1,VOL2,VOL3,VOL4)
```

- **Example 7:**

```
PDS> MOU/NOOP/GLOBAL DK0: UPDATEMM28B

PDS> MOU DK0: UPDATEMM28B

The volume is now accessible to the user.
```

ON

FUNCTION

The ON command specifies the action to be taken if, in a batch or indirect command file, the completion of a command returns an error. You must fully specify the ON command on one line.

REQUIRED PRIVILEGE

ANY

FORMAT

[$]ON *error-severity [THEN] action*

parameter definitions

error-severity

Any error up to the severity stated, and one of the following:

WARNING
ERROR
SEVERE_ERROR

action

One of the following:

Action	Function
CONTINUE	Ignores the error and continues processing commands in the command file.
GOTO label	Alphanumeric string beginning with an alpha character, that must appear together with a colon in front of a later command in the file. If the ON condition is satisfied, the GOTO causes all commands to be ignored until the label is found.
STOP	Terminates execution of command file; that is, ignores all further commands. In interactive mode, control is returned to PDS.
Any valid DCL command note	MCR-mode commands are not valid as an action.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

An ON command remains in force until the next ON command, and is then superseded entirely. On exit from the command (indirect or batch) file, the default error condition is set (see default below).

The action specified in the ON statement is not checked for syntax errors until the required error-severity is encountered. When this occurs and the action is attempted but found to be in error then the default condition (ON ERROR STOP) is enforced.

If the ON command itself is in error, that is, ON error-severity, this command is ignored and any previous error condition remains in force. See Section 8.7.1 for a description of ON and associated commands.

Default

[$] ON ERROR STOP is assumed by default at the beginning of a terminal session (LOGIN) or the beginning of a batch job ($JOB).

EXAMPLES

- Example 1:

```
$ON ERROR STOP
$MACRO MYPROG
$LINK MYPROG
$RUN MYPROG
```

  In this example, $ON has no effect on the MACRO assembly itself. If the assembly is completed with nothing worse than a warning, the job proceeds to $LINK. If the linking is completed with nothing more severe than a warning, the job proceeds to $RUN. If, at any stage, an ERROR is received (for example, ERROR or SEVERE ERROR) the job terminates.

- Example 2:

```
$JOB ENGINE3 RUN1 30
$ON WARNING GOTO ELSE
$LINK MYPROG
$RUN MYPROG
$STOP
$ELSE: LINK OLDPROG
$RUN OLDPROG
$EOJ
```

  In this example, if the $LINK of MYPROG produces any errors (warning or worse) then OLDPROG is linked and run. If no errors are found, MYPROG is run and the job terminates.

- Example 3:

```
ON WARNING GOOTO ALTER
MOUNT/NOOP DKO: MYDISK
COPY DKO:[200,200]*.* *.*
GOTO COMMON
ALTER: MESSAGE/OPER PLEASE LOAD MYDISK ON DKO:
MOU DKO: MYDISK
COPY DKO:[200,200]*.* *.*
COMMON: DISM DKO:
```

In this example there is an error in the ON action. Thus, if the MOUNT or COPY command fails - the command file is terminated by the default condition ON ERROR STOP.

- Example 4:

```
$JOB/MCR FRANK RUN2 30
$ON SEVERE_ERROR CONTINUE
$PIP [200,200]=[300.300]*.MAC
$ON SEVRER GOTO FINIT
$MAC @FULLMAC.CMD
$TKB @FULLBLD.CMD
$RUN TESTPROG
$FINIT: PRINT *.LST
$EOJ
```

In this example, the second ON statement is in error and so the previous ON SEVERE_ERROR CONTINUE remains in force throughout the job. Errors are ignored and all commands attempted.

PRINT

FUNCTION

The PRINT command enables you to queue one or more specified files for output on the line printer. You can optionally delete the file or files after they have been printed.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> [$]PRINT*[/quals]*

FILE? *filespec1[,...filespecn]*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/DELETE	Instructs the system to delete the file or files after they have been printed.
/FORMS:n	Indicates the type of form where the specified files are to be printed. The association of a value of n with a particular form is installation dependent. n is a digit from 0 to 6. The current forms type is set by the system operator. Default is /FORMS:0.
/COPIES:n	Determines the number of file copies to be printed. n is an integer from 1 to 32. Default is /COPIES:1.
/PRIORITY:pri	Enables you to request that a file be printed at a low priority (for example, priority 1). pri must be between 1 and n, where n is a value defined by the system manager at system generation time. The default for this value is 100. Default is /PRIORITY:100.
/AFTER:[(date]time[)]	Enables you to specify a time, or a date and time, when the file is printed. The system keeps the file in the spooler queue until the specified time. The file is then queued for immediate processing. You specify time in the form hh:mm. You specify the date in the form dd-mmm-yy. Default is /AFTER:00:00 (that is, immediately).
/NOBANNERS	Suppresses the printing of the file identification banner pages.

Qualifier	Explanation
/NOTRANSFER	Inhibits the copying of the queued file(s) to the spooling device. The file(s) are printed direct from the volume where it resides. Take care when using this option to ensure that the device is not dismounted before printing is complete.
filespec	Specifies the file to be printed. Wildcards are allowed. The file type is optional and defaults to .LST.

COMMAND VARIATIONS

On a multiuser system, files are never transferred to the spooled device (/NOTRANS implied).

TECHNICAL NOTES

If files are queued with a different forms type, a message is sent to the operator when a change of forms type becomes necessary, in order that the remainder of the queue can be printed.

On a timesharing system, take care when specifying /NOTRANS. Do not dismount the volume from which the files are printed and do not log out the terminal as this causes the volume to be automatically dismounted.

EXAMPLES

- Example 1:

```
PDS> PRINT
FILE? MACLIST
```

- Example 2:

```
$PRINT FREAN.MAC;3,PEEK.LST;*
```

- Example 3:

```
PDS> PRI/DE B4.MAC
```

- Example 4:

```
PDS> PRI/AFTER:10:30 FRED
```

QUEUE

FUNCTION

The QUEUE command enables you access to the queue to accomplish the following operations:

1 Interrogate the queue (/LIST).

2 Remove an entry from the queue (/REMOVE).

3 Add to the queue (/ADD). This is the default.

4 Display the status of all queue entries (/ALL).

5 Modify the current status or attributes of a file that is queued for printing (/MODIFY).

You can also use the PRINT and SUBMIT commands to add files to the line printer and batch queues.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

List Format:

PDS> QUEUE */LIST*

[QUEUE? *devicename]*

parameter definitions

/LIST

Displays the status of your queue entries.

devicename

If you specify a device, only entries in a queue for that device are listed. If a device is not specified, user entries in queues for all devices are listed.

FORMAT

Remove Format:

PDS> QUEUE/REMOVE

SEQUENCE? *seqno*

parameter definitions

/REMOVE
Removes the queue entry specified by a sequence number. You display the entries using the QUEUE/LIST command.

seqno
Sequence number of a queue entry to be removed. To find out the sequence number of a queue entry, enter the QUEUE/LIST command.

FORMAT

Add Format:

PDS> QUEUE/ADD *[/quals]*

[QUEUE? *devicename]*

FILE? *filespec [,....]*

parameter definitions

/ADD
Default operation. The /ADD qualifier adds the specified file to the named queue. You can modify the ADD operation by using the following qualifiers.

/quals
One of the following:

Qualifier	Explanation
/FORMS:n	Indicates the type of form where the specified files are to be printed. The association of a value of n with a particular form type is installation dependent. n is a digit from 0 to 6. Default is /FORMS:0 .
/COPIES:n	Determines the number of copies to be printed. n is an integer from 1 to 32. Default is /COPIES:1.
/DELETE	Requests the system to delete the specified files after they have been processed.
/PRIORITY:pri	Enables you to request that a file be queued at a low priority. pri must be between 1 and n, where n is a value defined by the system manager at system generation time. Default is /PRIORITY:100.
/AFTER:[(date]time[)]	Enables you to specify a time, or a date and time, when the file is to be printed. The system keeps the file in the queue until the specified time. The file is then queued for immediate processing. You specify time in the form hh:mm. You specify the date in the form dd-mmm-yy. Default is /AFTER:00:00 (that is, immediately).
/NOBANNERS	Suppresses the printing of the file identification banner pages.
/NOTRANSFER	Inhibits the copying of the queued file to the spooling device. The file is printed directly from the volume where it resides. Take great care when using this facility on a timesharing system, as, if the volume is dismounted before the queue entry is processed, you can not access the file.

devicename
Specifies the relevant queued device. Default is LP0.

filespec
Specification of one or more files to be added to the queue specified. The filespecs must contain a file name. Wildcards are allowed. The file type is optional and is defaulted to .LST.

FORMAT

All Format:

PDS> QUEUE/ALL

[QUEUE? *devicename]*

parameter definitions

/ALL
Display status of entries in all queues or the specified queue.

devicename
Specifies the relevant queued device. If you do not specify it, all entries in all queues are listed.

FORMAT

Modify Format:

PDS> QUEUE/MODIFY *[/quals]*

SEQUENCE? *seqno*

parameter definitions

/MODIFY

Enables you to change the current status or attributes of a file queued for printing or batch job execution that has not yet started printing or executing.

/quals

One of the following:

Qualifier	Explanation
/FORMS:n	Indicates the type of form where the specified files are to be printed. The association of a value of n with a particular form type is installation dependent. n is a digit from 0 to 6. Default is /FORMS:0.
/COPIES:n	Determines the number of copies to be printed. n is an integer from 1 to 32. Default is /COPIES:1.
/DELETE	Requests the system to delete the specified files after they have been processed.
/PRIORITY:pri	Enables you to request that a file be queued at a low priority. pri must be between 1 and n, where n is a value defined by the system manager at system generation time. Default is /PRIORITY:100.
/AFTER:[(date]time[)]	Enables you to specify a time, or date and time, when the file is to be printed. The system keeps the file in the queue until the specified time. The file is then queued for immediate processing. You specify time in the form hh:mm. You specify the date in the form dd-mmm-yy. Default is /AFTER:00:00 (that is, immediately).
/DEVICE:dev	Dev is the new device where the entry is to be queued.
seqno	Sequence number of the queue entry to be modified. To find out the sequence number of a queue entry, enter the QUEUE/LIST command.

COMMAND VARIATIONS

On a multiuser system, files are never transferred to the spooled device (/NORTRANS implied).

TECHNICAL NOTES

All QUEUE functions are interruptable except QUEUE/REMOVE.

EXAMPLES

- **Example 1:**

```
PDS> QUE/LIST

DEV ACT ACCOUNT FILE SPECIFICATION  SEQ PRI FO CO PBCA

LP0 * [123,22] DR0:[123,22]IUG4.DOC;1  100 0 1
TT35 [123,22] DR0:[123,22]IPDS14.DOC;1 1 100 0 1 *
```

- **Example 2:**

```
PDS> QUEUE/ALL LP0

DEV ACT ACCOUNT FILE SPECIFICATION  SEQ PRI FO CO PBCA

LP0 * [123,22] DR0:[123,22]IUG4.DOC;1  100 0 1
LP0 [122,6] DB0:[122,6]JB11.DOC;1  1 100 0 1
LP0 [122,6] DB0:[122,6]26770.DOC;1  2 100 0 1
LP0 [200,130] DR0:[1,4]TCPDEVT.LST;1  3 100 0 1
LP0 [23,10] DB0:[23,10]GAMMV4.DOC;1 4 100 0 1
LP0 [123,40] DR0:[123,40]PRLUG.DOC;1 5 100 0 1 *
AFTER 17:00
```

The headings shown in examples 1 and 2 have the following meaning:

Heading	Meaning
DEV	Device
ACT	Active. An asterisk (*) in this column indicates that the file is currently being printed.
ACCOUNT	Account
FILE SPECIFICATION	File specification
SEQ	Sequence
PRI	Priority
FO	Forms Type
CO	Copies
PBCA	Preserve Banners Concatenated After
	An asterisk (*) in the P column indicates that the system preserves the file after printing (that is, does not delete the file).
	A letter N in the B column indicates that the sytem will not print any banners.
	A letter C in the C column indicates that the system will concatenate output.
	An asterisk (*) in the A column indicates that the system will hold the file until the time shown on the line after the queue entry has passed.

- **Example 3:**

```
PDS> QUEUE/REMOVE 2
```

- **Example 4:**

```
PDS> QUEUE/ADD/COPIES:4/DELETE [ESC]
QUEUE? LP0:
FILE? LIST.MAP;4
```

- **Example 5:**

```
PDS> QUEUE/ADD/PRIO:10 LP3 LISTFILE
```

- **Example 6:**

```
$QUEUE/PRIORITY:40 LP3 ADD.MAC
```

- **Example 7:**

```
PDS> QUEUE/DELETE
FILE? MYFILE.MAC
```

- **Example 8:**

```
PDS> QUEUE/MODIFY/AFTER:09:30
SEQUENCE? 4 [ESC]
QUEUE? LP0
```

REMOVE

FUNCTION

The REMOVE command enables you to remove an installed task, or shareable global area (SGA), from the system.

REQUIRED PRIVILEGE

PR.RTC

FORMAT

PDS> REMOVE *[/quals]*

TASK? *name*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/COMMON	Removes the common area SGA named.
/LIBRARY	Removes the resident library SGA named.
/REGION	Removes the installed region SGA named.
/TASK	Removes the installed task named (default).

You can also specify the following qualifier when removing a task (/TASK):

Qualifier	Explanation
/NOHEADER	Removes a task whose header has been corrupted.

name

Installed name of the task, common area SGA, resident library SGA, or installed region SGA.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

Removing a task or SGA reverses the effect of the INSTALL command. You cannot remove a task if it is active, fixed, or has nodes accounted to it. If there is any outstanding data from SEND directives to the task, it is returned to the pool.

You cannot remove an SGA until you have removed all the tasks that map onto it.

EXAMPLES

- Example 1:

```
PDS> REMOVE MYLOL
```

  Removes the task with installed task name MYLOL.

- Example 2:

```
PDS> REM/COM SYST20
```

  Removes the common area SGA with installed name SYST20.

- Example 3:

```
PDS> REM/NOHEAD MYCLIB
```

  Removes the task MYCLIB even though its task header has been corrupted.

RENAME

FUNCTION

The RENAME command enables you to rename an existing file.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> RENAME

OLD? *oldspec*

NEW? *newspec*

parameter definitions

oldspec
Specification of an existing file.

newspec
New name for oldspec.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

Both oldspec and newspec must contain a file name and file type. Wildcards are allowed. The device field in both file specifications must be the same, because you cannot rename files from one device to another. If you omit the version field, the normal defaults apply, see Section 6.3.1.

EXAMPLES

- Example 1:

```
PDS> RENAME
OLD? MYFILE.OBJ;1
NEW? BACKUP.OBJ;1
```

- Example 2:

```
$RENAME MYFILE.OBJ;1,BACKUP.OBJ;1
```

- Example 3:

```
PDS> RENAME
OLD? MYFILE.OBJ;1,BACKUP.OBJ;1
```

- Example 4:

```
PDS> RENAME CAROL.*;*
NEW? FRED.CBL;*
```

RUN

FUNCTION

The RUN command enables you to execute an executable task. You can issue RUN for a timesharing task (Format 1) or for a real-time task (Formats 2 to 6). See the Technical Notes for the functions that correspond to these formats.

REQUIRED PRIVILEGE

RUN/TIMESHARING—PR.RUN

RUN/REAL-TIME—PR.RUN,PR.RTC

FORMAT

1. Timesharing:

PDS> [$]RUN *[/quals]*

FILE? *filespec*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/TIMESHARING	Runs the task in the specified file as a runs the task in the specified file as a timesharing task. This is the default and you can omit it.
/INCREASE:tasksize-increment	Overrides the EXTTSK option specified in the LINK command. You specify the decimal number of words by which the upper read/write area is to be extended. The value you specify is rounded up to the next 32-word boundary.
filespec	Specification of a file that contains the task image. The specification must include a file name. If you omit the file type field, .TSK is assumed.

EXAMPLES

- Example 1:

```
PDS> RUN [200,40]PASCAL.TSK;4
```

- Example 2:

```
$RUN PASCAL
```

FORMAT

2. Memory:

PDS> RUN */MEMORY[/options]*

TASK? *taskname*

[DEVICE? *terminal]*

parameter definitions

/MEMORY
Indicates that the task is to be run immediately. If sufficient memory is not available to run the task, and error message is returned.

/options
One of the following:

Option	Explanation
/UIC:[m,n]	Specifies the UIC.
/PARTITION:par	Specifies the name of the partition where the task is to run.
/PRIORITY:pri	Specifies the task priority (decimal).
/NOPROMPT	Suppresses the PDS prompt.
/INTERVAL:interval	Specifies the time interval at which the task is to be periodically rerun, in the form: xxt where:

Option	Explanation
	• xx = Number of hours, minutes, seconds, or clock ticks. • t = is one of: H–hours M–minutes S–seconds C–clock ticks
/REALTIME	Indicates that the task is to be run as a real-time task (implied by /MEMORY).

taskname
Name of the task to be run immediately.

terminal
TI for which the real-time task is run. If you do not specify TI, the terminal where the RUN command was issued is used.

EXAMPLES

- Example 1:

```
PDS> RUN/MEMORY/PART:FILE JK03
```

- Example 2:

```
PDS> RUN/MEM/UIC:[100,10] MART6
```

- Example 3:

FORMAT

3. Synchronize:

PDS> RUN */SYNCHRONIZE:unit[/DELAY:delay][/options]*

TASK? *taskname*

[DEVICE? *terminal]*

parameter definitions

/SYNCRONIZATION:unit

Synchronization clock unit, as follows:

HOURS–hours
MINUTES–minutes
SECONDS–seconds
TICKS–clock ticks

/DELAY:delay

Delay period after synchronization, in the form: xxt (as in Format 2)

/options

One of the following:

Option	Explanation
/UIC:[m,n]	[m,n] is user identification code.
/PARTITION:par	par is partition name.
/PRIORITY:pri	pri is priority number (decimal)
/NOPROMPT	Suppresses PDS prompt.
/INTERVAL:interval	Time interval, as in Format 2.
/REALTIME	Task is to be run as a real-time task (implied by /SYNC).

taskname

Name of the task to be synchronized.

terminal

TI where the real-time task is run. If you do not specify TI, the terminal where the RUN command was issued is used.

EXAMPLES

- Example 1:

```
PDS> RUN/SYNC:HOUR/DELAY:10M/INTERVAL:25M CAROL
```

When the time is next an exact number of hours, wait ten minutes, then run task CAROL every twenty-five minutes.

If the time is now 10.15, then task CAROL runs at 11.10, 11.35, 12.00, 12.25 and so on.

- Example 2:

```
PDS> RUN/SYNCH:HOUR/DELAY:5M/PART:SYSTEM XK3
```

Run task XK3 at 5 minutes past the next hour in the SYSTEM partition.

FORMAT

4. Schedule:

PDS> RUN/SCHEDULE *:time[/options]*

TASK? *taskname*

[DEVICE? *terminal]*

parameter definitions

/SCHEDULE:time
Absolute time of day the task is to begin execution. Time is expressed as: **hh:mm.**

/options
One of the following:

Option	Explanation
/UIC:[m,n]	[m,n] is the user identification code.
/PARTITION:par	par is the partition name.
/PRIORITY:pri	pri is the priority number (decimal).
/NOPROMPT	Suppresses the PDS prompt.
/INTERVAL:interval	Time interval, as in Format 2.
/REALTIME	Indicates that the task is to be run as a real-time task (implied by /SCHED).

taskname
Name of the task to be scheduled.

terminal
Name of the terminal where the task is to be scheduled.

EXAMPLES

- Example 1:

```
PDS> RUN/SCHED:10:23:00/INTER:30S MKLOL
```

  Run task MKLOL at 10:23:00 and every 30 seconds thereafter.

-

```
PDS> RUN/SCHED:10:30:00/PRI:120 MYTSK
```

  Run MYTSK at 10:30:00 at priority 120.

FORMAT

5. Delay:

PDS> RUN/DELAY *:delay[/SYNCHRONIZE:unit][/options]*

TASK? *taskname*

[DEVICE? *terminal]*

parameter definitions

/DELAY:delay
Delay period before the task is to be periodically rerun, in the form: xxt (as in Format 2).

/SYNCHRONIZE:unit
See Format 3.

/options
One of the following:

Option	Explanation
/INTERVAL:interval	Interval at which the task is to run after a delay. See Format 2 for the interval format.
/UIC:[m,n]	User identification code.
/PARTITION:par	Partition name.
/PRIORITY:pri	Priority number (decimal).
/NOPROMPT	Suppresses the PDS prompt.
/REALTIME	Indicates that the task is to be run as a real-time task (implied by /DELAY).

taskname
Name of the task to be run after the delay.

terminal
Name of the terminal where the task is to be run after the delay.

EXAMPLES

- Example 1:

```
PDS> RUN/DELAY:30M/INTERVAL:20S/UIC:[30,2] MYTSK
```

  Wait 30 minutes, then run [30,2]MYTSK every 20 seconds.

- Example 2:

```
PDS> RUN/DELAY:2H/PART:GLOBZ XKEE9
```

Wait 2 hours, then run task XKEE9 in the partition GLOBZ.

FORMAT

6. Real-time:

PDS> RUN */options*

TASK? *taskname*

[DEVICE? *terminal]*

parameter definitions

This format enables you to run an installed task as soon as memory is available.

/options

One of the following:

Option	Explanation
/INTERVAL	Interval at which the task is to run after a delay. See Format 2 for the interval format.
/UIC:[m,n]	User identification code.
/PARTITION	Partition name.
/PRIORITY:pri	Priority number (decimal).
/NOPROMPT	Suppresses the PDS prompt.
/REALTIME	Indicates that the task is to be run as a real-time task (as implied by any of the previous options).

taskname

Name of the installed task to be run.

terminal

Name of the terminal the task is to run on.

EXAMPLES

```
PDS> RUN/NOPROMPT MYTASK
```

Run MYTASK as soon as memory is available, and suppress the PDS prompt.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

You can issue the RUN command for a timesharing task (Format 1) or for a real-time task (Formats 2 to 6).

To run a real-time task:

1 You must have installed it in the system (see the INSTALL command).

2 You can reset the task's UIC, partition, and priority from those in force at installation.

3 You can suppress the PDS prompt (for example, to enable terminal dialogue with the task). Ctrl/C reactivates the PDS prompt. Under /NOPROMPT, PDS is still running and times out in the usual way unless you type Ctrl/C.

In the command format for a real-time task, taskname refers to the installed taskname. See the INSTALL command for further details.

For running real-time tasks, the RUN command has one of the following formats:

- Format 2—Runs a task immediately (/MEMORY).
- Format 3—Synchronizes the running of a task with a time unit, and optionally reschedules the task after a specified interval (/SYNCHRONIZE).
- Format 4—Schedules the task for running at a specified time, and optionally reschedules the task after a specified interval (/SCHEDULE).
- Format 5—Delays the task for a specified period, and optionally reschedules the task to rerun periodically (/DELAY).
- Format 6—Runs the task as soon as memory is available, (/REALTIME), without any of the other qualifiers stated above.

SET

FUNCTION

The SET command enables you to perform the following operations:

1 Suppress the output of information messages (Format 1—SET QUIET).

2 Establish a new default device and/or UFD for subsequent file specifications supplied by you, available on timesharing systems only. (Format 2—SET DEFAULT).

3 Defer printing of spooled files (Format 3—SET PRINTING DEFERRED).

4 Change your interactive or batch password (Format 4—SET PASSWORD).

5 Change the characteristics of your terminal. If you are logged in under a user name whose UIC is [1,1] you can change the characteristics of any terminal (Format 5—SET TERMINAL).

6 Change file protection code (Format 6—SET PROTECTION).

7 Reset the priority of an active task (Format 7—SET PRIORITY).

8 Set the terminal UIC, available on multiuser systems only (Format 8—SET UIC).

9 Set real-time control, available on timesharing systems only (Format 9—SET REAL_TIME_CONTROL).

10 Write the bootstrap block from the file specified (Format 10—SET BOOTSTRAP).

11 Specify file's end-of-file position (Format 11—SET END_OF_FILE).

12 Supress the output from indirect command files (Format 12—SET NO ECHO).

13 Execute SCI commands from a non-monitor console (Format 13—SET SCI).

REQUIRED PRIVILEGE

SET BOOTSTRAP—PR.DEV
SET END_OF_FILE—PR.FIL
SET DEFAULT—ANY
SET NO ECHO—ANY
SET [NO] QUIET—ANY
SET [NO] REAL_TIME_CONTROL—PR.RTC
SET PASSWORD—ANY
SET PRINTING—ANY
SET PRIORITY—PR.RTC
SET PROTECTION—PR.FIL
SET [NO] SCI—PR.SCI
SET TERMINAL—ANY
SET TERMINAL:[(]TTm[,...,TTn[)]]—[1,1]
SET UIC—PR.RTC or group code less than 10

FORMAT

1. Set Quiet:

PDS> *[$]SET[NO]QUIET*

parameter definitions

QUIET

Suppresses the output of information (usually accounting) messages. NOQUIET is the system default.

EXAMPLES

```
PDS> SET QUIET
```

FORMAT

2. Set Default (timesharing only):

PDS> [$]SET *DEFAULT*

[DEVICE AND/OR UFD? *[devicename:][UFD]]*

parameter definitions

DEFAULT

Changes your default device and/or UFD to the value you specify. If you omit both device name and UFD, the system reestablishes your initial default settings (established at LOGIN) for both values.

EXAMPLES

```
PDS> SET DEFAULT DK0:[200,200]
```

FORMAT

3. Set Printing Deferred:

PDS> [$]SET *PRINTING [NO]DEFERRED*

parameter definitions

PRINTING DEFERRED
Defers the printing of spooled files generated by the timesharing tasks run from your terminal. This defers printing until either you log out or you issue the SET PRINTING NODEFERRED command (the default).

EXAMPLES

```
PDS> SET PRINTING DEFERRED
```

FORMAT

4. Set Password:

PDS> SET PASSWORD *[/qual]*

OLD PASSWORD? *oldpassword*

NEW PASSWORD? *newpassword*

parameter definitions

/qual
/BATCH—enables you to change a batch password.

oldpassword
1- to 6-character alphanumeric password currently associated with your user name.

newpassword
1- to 6-character alphanumeric password that supersedes the old password.

EXAMPLES

- Example 1:

```
PDS> SET PASSWORD
OLD PASSWORD? glove
NEW PASSWORD? mitten
```

This example changes the password from glove to mitten. The response to the OLD PASSWORD? and NEW PASSWORD? prompts are not echoed on the screen.

- Example 2:

```
SET PASSWORD/BATCH
OLD PASSWORD? Sunday
NEW PASSWORD? Monday
```

This example changes the batch password from Sunday to Monday. The response to the OLD PASSWORD? and NEW PASSWORD? prompts are not echoed on the screen.

FORMAT

5. Set Terminal:

PDS> SET

FUNCTION? *TERMINAL[:(termlist)]*

ATTRIBUTE? *attribute*

parameter definitions

termlist

Can be specified only by [1,1] users and is a list of the terminals to be operated on by attribute.

attribute

One of the following:

Attribute	Explanation
terminaltype	One of the following: ASR33 KSR33 ASR35 LA30S LA30P LA36 LA120 VT05 VT50 VT52 VT61 VT100 VT200 VT300 SET TERMINAL terminaltype sets the characteristics other than the speed(s) to the default values listed in the *IAS Device Handlers Reference Manual.* If DS is appended to the command line, the speed is also set to the default value.
optionlist	One or both of: [NO]option option:value separated by spaces.

You can abbreviate each option and any short form listed with it as long as it remains unique within the list of SET TERMINAL options. You can negate each acceptable form of an option by using the prefix NO; for example, NOSCOPE. The options are as follows:

Option	Description
ALTMODE	For an old model Teletype that generates 175 or 176 (octal), you press the ALT key. Either of these characters is treated the same as ESCAPE.
ANSISEQUENCE	Escape sequences for the terminal are to be in ANSI mode.
BACKSPACE	Terminal responds to the backspace character.
BLOCKMODE	Terminal is a VT61 and is to be used in block mode.
BINARY	Terminal is to operate in binary mode.
CARRIAGERETURN or RETURN	Lines exceeding the terminal width as set are continued on the following line(s).
COMPATIBLE	Terminal requires RSX-11M compatible escape sequence handling.
CONTROLCFLUSH or CCF	Flush typeahead when you type Ctrl/C.
CONTROLS or CSQ	When the terminal is in deferred processing mode, terminal is set so that only Ctrl/S and Ctrl/Q are processed immediately.
DEFAULT	Terminal characteristics to system default values as existing at login time. If you have a UIC of [1,1] you can also set new default values.
ESCAPESEQUENCE	Terminal requires escape sequence recognition.
FORMSMODE	Terminal is a VT61 and is to be used in forms mode.

Option	Description
FULLDUPLEX	Invokes full duplex mode, in which input and output operate independently. For use with intelligent terminals. For details, see the *IAS Device Handlers Reference Manual*.
HANGUP	Hangs up dialup line. You cannot negate this.
HARDWAREFORMFEED or HFF	The characters form feed and vertical tab are recognized and do not need software simulation.
HARDWARETAB or HTAB	The character horizontal tab is recognized.
HOLD	(VT5x and VT61 terminals only) used to enter auto-hold mode. Output from the computer then stops automatically when the screen becomes full with output and can be resumed if you press the SCROLL key to enable a further line to be output. Pressing the SHIFT and SCROLL keys simultaneously enables a further page to be output. For this facility to work correctly, the terminal must transmit and receive at the same speed.
KEYBOARD	Terminal is capable of input.
LOCALCOPY	Terminal echoes all characters as you type them.
LOWERCASEKEYBOARD or LCKEYBOARD	Accepts lower case characters. If you use Ctrl/ type-ahead, characters are echoed as lower case, whether or not they are processed as lowercase.
LOWERCASEKEYBOARD	Can be consistently used with NOLOWERCASEINPUT.
LOWERCASEINPUT	Lower case characters are to be passed to a program performing input even if the program (for example, EDI) asks for case conversion.
LOWERCASEOUTPUT or LCOUTPUT LOWERCASEPRINTER or LCPRINTER	Terminal can print lower case characters.
LVF	Requires LA36-type vertical fill for form feed and vertical tab, that is, 66 nulls.
MESSAGES	Messages from other terminals are allowed.
NEWLINE	Terminal sends newline when the carriage return key is pressed.
NONSTANDARDTAB OR NSTAB	Terminal, on receiving tab character, does not space to the next 8-character boundary.
NOPARITY	Does not generate parity bit on character output.
PASSALLBITS	Terminal passes all eight bits of characters read to the user buffer for a READ_PASS_ALL request.
PRINTER	Terminal is capable of output.
PROCESSCONTROLC	When the terminal is in deferred processing mode, terminal is set so that only Ctrl/S, Ctrl/Q, and Ctrl/C are processed immediately.
SCOPE	Terminal is a VDU and the rubout physically erases characters from the screen.
SIMULATEFORMFEED or SFF	Form feed and vertical tab are to be software simulated to start a new page and skip to next six-line boundary respectively.
TAPE	Terminal has a low speed paper tape reader and interprets Ctrl/B and Ctrl/T accordingly.
TWOSTOPBITS or TSB	Terminal requires two stop bits as normally required for mechanical printers, for example, ASR33.
VERTICALFILL or VFILL	Terminal requires VT05-type vertical fill.

The option values are as follows:

Value	Explanation
FILL:n	n is fill required for carriage return. n = 7 supplies LA30S-type fill.
LENGTH:n	n is page length in lines.
NAME:name	name can be one of the following: ASR33, KSR33, ASR35, LA30S, LA30P, LA36, LA120, VT05, VT50, VT52, VT55, VT61, VT100. This option is for use in "deceiving" a program as to the type of terminal where it is running, that is, when you want mixed characteristics. The option sets only the location holding the name of the terminal type; see the *IAS Device Handlers Reference Manual.* **NOTE:** SET TERMINAL NAME:name does NOT set the corresponding characteristics implicitly.
PARITY:type	Type is EVEN or ODD. Set line to generate characters with parity. Note that parity is not checked on input.
READAHEAD:type	Type is one of the following: NONE — No read-ahead allowed. DEFERREDPROCESSING or DP — Read-ahead accepted but not examined until a read that uses it is processed. IMMEDIATEPROCESSING or IP — Read-ahead is processed as it is typed but not echoed until it is read.
SPEED:(m:n)	Sets split-speed line. m is the keyboard (lower) speed. n is the printer or display (higher) speed.
SPEED:n	Sets line speed. n can be one of the following: Speed in baud 134 (meaning 134.5 baud) EXTA (DA11 external speed A) EXTA (DH11 external speed A) EXTB (DH11 external speed B)
WIDTH:n	n is the page width in columns.

EXAMPLES

- Example 1:

```
PDS> SET TERMINAL WIDTH:50 LENGTH:30
```

  The width is set to 50 characters, the length to 30 lines. Lines of more than 50 characters are continued on the following lines.

- Example 2:

```
PDS> SET TERMINAL
ATTRIBUTE? SPEED:(150:9600)
```

  Terminal is to send at 150 baud and receive at 9600 baud.

- Example 3:

```
PDS> SET
FUNCTION? TERMINAL
ATTRIBUTE? VT05 DS
```

Terminal is a VT05 and is to run at the corresponding speed (2400 baud).

- Example 4:

```
PDS> SET TERMINAL NAME:VT61
```

The terminal type is recorded as being VT61 but no characteristics are thereby changed.

- Example 5:

```
PDS> SET TERMINAL:(TT3,TT5,TT6) SPEED:300
```

Terminals TT3, TT5 and TT6 are to send and receive at 300 baud.

FORMAT

6. Set Protection:

PDS> [$]SET PROTECTION *[/OWN]*

FILE? *filespec*

[PROTECTION? *(code)]*

parameter definitions

/OWN

If you specify /OWN, it changes the ownership UIC of the file to be the same as the UFD under which the file is stored.

filespec

Specification of the file to which you apply protection code.

code

Protection code to be applied to filespec. See Section 6.2.2. If you specify /OWN, (code) is an optional parameter.

EXAMPLES

- Example 1:

```
PDS> SET PROTECTION/OWN CATHS.DAT ESC
PROTECTION? (GRO:R, SY:R, WORLD:, O:RWDE)
```

- Example 2:

```
$SET PROTECTION TONY.MAC (OW:RWED,SY:,GR:,W:)
```

- Example 3:

```
PDS> SET PRO MYPROG.COB (SY:RWED,OW:RWDE,WO:DERW,GR:RWED)
```

- Example 4:

```
PDS> SET PRO/OWN FILE.MAC
```

- Example 5:

```
PDS> SET PROTECTION JUD.CBL
PROTECTION? (WO:, GR:)
```

FORMAT

7. Set Priority:

PDS> SET PRIORITY

TASK? *taskname [terminal] priority*

parameter definitions

taskname
Installed name of the task whose priority you are altering.

terminal
Terminal where the task was activated. The default is the current terminal.

priority
New task priority (that is, a decimal number ranging from 1 to 250).

EXAMPLES

- Example 1:

```
PDS> SET PRIORITY SCAN TT4 120
```

  Sets the priority of the installed task SCAN running from terminal TT4 to 120.

- Example 2:

```
PDS> SET PRIORITY XYZ,,130
```

  Sets the priority of the installed task XYZ to 140.

FORMAT

8. Set User Identification Code (multiuser systems only):

PDS> SET UIC

UIC? *uic*

parameter definitions

uic
Sets the terminal UIC to that specified (multiuser systems only).

EXAMPLES

```
PDS> SET UIC
UIC? [200,20]
```

FORMAT

9. Set [No] Real-Time Control (timesharing systems only):

PDS> SET [NO]REAL_TIME_CONTROL

This operation has no parameters. See the TECHNICAL NOTES for a description.

FORMAT

10. Set Bootstrap:

PDS> SET BOOTSTRAP

FILE? *filespec*

parameter definitions

filespec
File with a saved IAS System image. You must specify a device in the filespec, and it must be allocated and mounted for you. See the Technical Notes for a description.

EXAMPLES

```
PDS> SET BOOTSTRAP DK0:[11,17]IAS.SAV
```

FORMAT

11. Set End-of-File:

PDS> SET END_OF_FILE

FILE? *filespec [block:byte]*

parameter definitions

filespec
File for which the end-of-file is to be set.

block
Block number for the placement of the end-of-file pointer. This value is decimal.

byte
Byte location of the end-of-file pointer, or the first unused byte of the block. This value is decimal.

EXAMPLES

```
PDS> SET END_OF_FILE
FILE? GAYE.TMP 22:511
```

NOTE: The EOF pointer cannot be located beyond the highest number of blocks allocated to the file. The maximum value for byte is 511 (decimal). The default is the last byte of the last block allocated to the file.

FORMAT

12. Set Echo:

PDS> SET [NO] ECHO

parameter definitions
This command has no parameters. See Set Echo in Technical Notes for more information.

EXAMPLES

```
PDS> SET ECHO
```

FORMAT

13. Set Sci:

PDS> SET [NO] SCI

parameter definitions

This command has no parameters for this command. See Set Sci in TECHNICAL NOTES for more information.

EXAMPLES

```
PDS> SET NOSCI
```

COMMAND VARIATIONS

SET DEFAULT and SET [NO]REAL_TIME_CONTROL are not available on multiuser systems. SET UIC is not available on timesharing systems.

TECHNICAL NOTES

Set Default (timesharing only)

The system manager allocates a default device to each user. The default takes effect each time you log in. The initial default UFD is equivalent to your UIC. You must issue the SET DEFAULT command to change either or both values for file specifications included in subsequent commands. The SET DEFAULT command does not affect file specifications written in programs. To reestablish the default settings in effect at login, issue SET DEFAULT without any other values.

Set Password

The system does not display either the old or the new password when it is prompted for. This command is not permitted in batch mode.

Set Batch Password

This command enables you to redefine the batch password to be associated with the account. Until this command is issued, any user can submit a batch job that could run for and be charged to the user's account. This command is not permitted in batch mode.

Set Terminal

The SET TERMINAL command enables you to change the characteristics of the terminal. Terminal characteristics revert to the system defaults when a dialup line is disconnected or when you log out.

For details of the software facilities associated with characteristics see the *IAS Device Handlers Reference Manual*. For the setting of characteristics at system generation, see the *IAS Installation and System Generation*.

Set Priority

The SET PRIORITY command enables you to alter the priority of an active task.

Set [No] Real-Time Control (timesharing only)

This command enables CLIs and control commands to run at a high priority. This enables users with PR.RTC privilege to have some control over real-time tasks (for example, so that looping tasks can be aborted).

Set User Identification Code (multiuser only)

This command changes the UIC in subsequent commands. The SET UIC command does not affect file specifications written in programs.

NOTE: Instead of the MCR mode SET /UIC command, use SET DEFAULT on a timesharing system or SET UIC on a multiuser system.

Set Bootstrap

This command writes a bootstrap routine from the specified file onto block 0 of the specified device. When the device is hardware bootstrapped, the system contained in the specified file becomes active.

Set Endoffile

This command enables you to specify the end-of-file pointer for a file. This is normally used when a file has been corrupted. This can happen after a system crash, where the contents of the file are useful, but the EOF pointers are incorrect and thus prevent you from obtaining the information.

If you are a file owner or have a system level UIC, you can read or change this file attribute. You can do this without having read or write access. If you are group or world to the file owner's UIC, you need read-access to read the attribute and write-access to change it.

If you do not specify the EOF position (block:byte), the EOF is placed at the last byte of the last block allocation to the file.

Set Echo

This command enables or disables the echo from the execution of an indirect command file.

Set Sci

This command enables or disables the excution of Sci commands from a non-monitor console.

SHOW

FUNCTION

The SHOW command enables you to display specified information at your terminal. The parameter associated with the SHOW command determines the type of information displayed.

REQUIRED PRIVILEGE

Table 14–17 Required Privileges for the SHOW Commands

Command	Privilege Required
SHOW CLI	Any
SHOW DAYTIME	Any
SHOW DEFAULT	Any
SHOW DEVICES	Any
SHOW GLOBAL AREAS	Any
SHOW LUNS	PR.RTC
SHOW EXTENDED TASKSIZE	Any
SHOW MEMORY	Any
SHOW PARTITIONS	Any
SHOW STATUS	Any
SHOW TASKS	Any
SHOW CLOCK QUEUE	Any
SHOW I/O QUEUES	Any
SHOW SHAREABLE GLOBAL AREAS	Any
SHOW SWITCH REGISTERS	Any

FORMAT

PDS> SHOW

ATTRIBUTE? *parameter*

parameter definitions

parameter

One of the following:

Parameter	Function
CLI [cliname]	Displays information about all or selected Command Language Interpreters (CLIs) currently running in the system.
[DAY]TIME	Displays the current time and date.
DEFAULT	Displays the current user default device and UFD.
DEVICES[/PUD] [device[unit number]]	displays information about all or selected devices known to the system. See the Devices section. With /PUD, displays also the PUD address of the device unit(s).
EXTENDED_TASKSIZE_MAXIMUM	Displays current extension limit as an octal number of 32 word blocks.
GLOBAL_AREAS	Lists the following description of each installed shareable global area: Name Base address (octal) Size UIC Access Position independent or blank Creation date Type codes: TPA—task pure area LIB—resident library SGA COM—common area SGA IRG—installed region SGA DRG—dynamic region or task read/write resident overlay region
LUNS taskname	Displays current assignment of LUNS for an installed task.
MEMORY	Displays the use of the system's memory.
PARTITIONS	Displays information on system memory partitions. The information listed is partition name, size (octal) and type. Partitions can be: SC—system controlled T—timesharing UC—user controlled
STATUS	Displays information about the current status of the user's job.
TASKS[/ACTIVE]/MIDDLE [taskname] [terminal]	Displays task name, task status, task type, run priority, partition, and real memory address and current task size. This is the default. If you specify a task name, only information about the specified task is displayed. If you specify a terminal, information is given for tasks running for that terminal. If you do not specify a terminal the default is the current terminal. To display information about all active tasks, replace the taskname with a comma (,) and specify: PDS> SHO TAS/ACT/MID,,ALL

Parameter	Function
TASKS[/ACTIVE]/FULL taskname [terminal]	Displays a full version of the specified task's status information. Task name is mandatory for the FULL qualifier. The default is /MIDDLE.
TASKS[/ACTIVE]/BRIEF [taskname] [terminal]	Displays a brief version of the active tasks in the system. The information displayed is task name, task status, task type and terminal name. The default is /MIDDLE.
TASKS/FIXED	Displays all currently inactive fixed tasks in the system.
TASKS/CHECKPOINTABLE	Displays all checkpointable real-time tasks.
TASKS/INSTALLED	Displays all tasks currently installed in the system. This command sometimes shows multiple entries for a task, because the System Task Directory (STD) is changing dynamically while the list of tasks is being displayed.
TASKS/TIMESHARING [terminal]	Displays all current timesharing tasks, or all timesharing tasks active for the specified terminal. Timesharing systems only. Note that if a task is not currently in memory, the CPU time is not available and displays as blank.
CLOCK_QUEUE	Displays the system clock queue.
IO_QUEUES	Displays I/O request queues.
SHAREABLE_GLOBAL_AREAS	Displays the name of all tasks in the STD that are linked to one or more shareable global areas, and the SGA to which each is linked.
SWITCH_REGISTERS	Displays the contents of the switch register on the PDP-11/34.

COMMAND VARIATIONS

SHOW CLI, SHOW TASKS/TIMESHARING, and SHOW DEFAULT are not available on multiuser systems.

On timesharing systems, SHOW STATUS displays default information for the system active tasks, and all device information (on multiuser systems the device information is not reported).

TECHNICAL NOTES

Devices

The command SHOW DEVICES causes the system to display the symbolic names of the devices known to the system. You can choose to print information about one particular device (for example, DK0); all devices of that type (for example, DK); or all devices (default). The physical unit directory (PUD) addresses of the units can also be requested. Physical device names are followed by ** if the device handler is resident. System logical device names are followed by the associated physical device names. In the listing are messages giving additional information about particular devices. The messages and their meanings are as follows:

Message	Meaning
GLOBAL	The device is mounted globally. See the MOUNT command.
MOUNTED	The device is mounted.
REALTIME	The device is mounted for real-time activity.
T/S DEVICE	The device is a timesharing device. If followed by an X (see example), the device has been explicitly allocated to a user.
T/S TERMINAL	The terminal is a timesharing terminal.
SYSTEM	The device is a system device.
SPOOLED:n	The device is spooled. n is the current setting of the forms type.
TIMESHARING:n	n is the number of timesharing users accessing the device.

Status

Active timesharing tasks are displayed in the order they are initiated. On a timesharing system, the task name is always that assigned by the system of the form JOBxxx, (see example 4).

On multiuser systems, the task name is the installed task name, or, if you initiated the task by means of the RUN filespec, the name is of the form TTnnx where TTnn is terminal number and x is a unique character, (see example 5).

Memory

The command PDS> SHOW MEMORY displays on a VDU terminal (VT05, VT50, VT52, VT55, VT61, VT100, VT200, VT300) the memory usage and task activity of the system provided that the terminal handler was configured to support escape sequences.

The display appears in two rows of columns (one row on a VT50). Each column refers to a portion of memory.

All types of task area within the occupied memory are displayed by task name. Shareable global areas are displayed by name.

Tasks listed down the right side of the screen are real-time tasks waiting for memory to become available. The number of nodes available and the largest hole are included in the heading information at the top of the screen. The name of the currently active task, and the terminal for which it is running, are also displayed only if the SHOW MEMORY task (...DEM) is run as a high priority real-time task.

On the display, at the bottom of each column,

<-> Indicates a task's read/write (impure) area.

[-] Indicates an inactive fixed task.

<=> Indicates a task's read/only (pure) area.

[=] Indicates a shareable global area (SGA) or dynamic region.

<+> Indicates a fixed or noncheckpointable task.

Once the memory diagram is displayed, you can alter the portion of memory being displayed by using one of the following commands:

NOTE: Do not type Ctrl/C or use the control key with these commands as results are not predictable.

FORMAT (no prompt):

```
B[ASE] base
```

Where:

base	Beginning of the area of memory whose activity is to be displayed. You can enter base either in the form: mK that is, mK words (m decimal), or in the form: n that is, n octal blocks of 32 words or 100 (octal) bytes
G[RAIN] grain	Resets the amount of memory referred to by a single column of the display. Grain has the same syntax as base.
C[LEAR]	Clears the VDU screen and redisplay. You can use this, for example, to clear an external message from the screen.
E[XTENT] nK	Changes extent of display.
E[XTENT] ALL	Displays all memory (initial state).
I[NTERVAL] n	Updates display every n seconds (initially n = 1). n = 0 gives continuous update.
X	Exit to PDS.

WARNING: If you run SHOW MEMORY with I=0, particularly on a 9600 baud terminal, the speed of system performance is reduced. This occurs to a greater degree if you run SHOW MEMORY as a real-time task; the more terminals running, the greater the speed reduction.

EXAMPLES

- Example 1:

```
PDS> SHOW DAYTIME
1-JUN-78 10:53:41
```

- Example 2:

```
PDS> SHO DEV
  TT0   **  T/S TERMINAL
  CI0   TT0
  CO0   TT0
  CL0   LP0
  TO0   TT6
  SP0   SY0
  PI0   **
  MO0   **
  MM0   **  T/S DEVICE X MOUNTED     TIMESHARING:1
  DT1   **  T/S DEVICE
  DT0   **  T/S DEVICE
  LP0   **  SYSTEM       SPOOLED:0   TIMESHARING:8
  TT11  **  T/S TERMINAL
  TT10  **  T/S TERMINAL
  TT7   **  T/S TERMINAL
  TT6   **  T/S TERMINAL
  TT5   **  T/S TERMINAL
  TT4   **  T/S TERMINAL
  TT3   **  T/S TERMINAL
  TT2   **  T/S TERMINAL
  TT1   **  T/S TERMINAL
  DS0   **                MOUNTED       GLOBAL TIMESHARING:1
  DB1   **  T/S DEVICE X
```

```
DB0   **  SYSTEM       MOUNTED      GLOBAL TIMESHARING:8
DK1   **
DK0   **  T/S DEVICE
SY0   DB0
```

- **Example 3:**

```
PDS> SHOW DEV/PUD DB

  DB1 152404 ** T/S DEVICE X   TIMESHARING:1
  DB0 152470 ** SYSTEM    MOUNTED   GLOBAL TIMESHARING:8
```

- **Example 4:**

```
PDS> SHO STA   (timesharing system)
  User SYSTEM UIC [1,1] TT03: 09:52:44 30-MAR-78
  JOB212 Size: 4K CPU: 0.00
  JOB215 Size: 4K CPU: 0.00
  JOB216 Size: 4K CPU: 0.00
  FILES OR DEVICES ASSIGNED
  FILE OR DEVICE   REDIRECTED  LUNS
  TT3:          -    NO LUNS
  DB0: IAS306       -    NO LUNS
  LP0:          -    NO LUNS
```

- **Example 5:**

```
PDS> SHO STA    (multiuser system)
  USER SYSTEM UIC [1,1] TT04: 09:57:16 30-MAR-78
  TT04A
  TT04B
  TT04C
```

- **Example 6:**

```
PDS> SHO TAS/ACT
  DB.... WF0 RT TT00
  TT.... WF0 RT TT00
  DK.... WF0 RT TT00
  DT.... WF0 RT TT00
  MM.... WF0 RT TT00
  MO.... WF0 RT TT00
  PI.... WF0 RT TT00
  F11ACP SUS RT TT00
  ERRLOG ST4 RT TT00
  ...ACT RUN TS TT00
  ...PDS ST0 TS TT00
```

- **Example 7:**

```
PDS> SHO TAS/FU ...PDS
...PDS ST0 TS TT02 001 GEN 00544300 CURRENT RW SIZE 031400
REGS 174000 013466 000033 000000 000015 030022 000000 004736 000356
EV 1-16 100171 EV 17-32 140400 ATL FLGS 040010 STD FLGS 020100
EV MASKS 000006 000000 000401 000000 MKTM CNT 000 ACT VERS 005
ATL ADDR 113740 STD ADDR 104440 TSK SIZE RO 063600 INITIAL RW 031400
I/O PEND 000 I/O PROG 000 POOL LIM 040 POOL USE 005 REQ TASK PI....
REGIONS .PURE. SYSRES
HW PARS 005447 005647 000000 016061 016261 016461 016661 002265
HW PDRS 077406 043406 000000 077402 077402 077402 016402 075002
```

- **Example 8:**

```
PDS> SHO EXT/MAX
Maximum Extend Size = 002000
```

SORT

FUNCTION

The SORT command enables you to sort files into a specified sequence. See the *PDP-11 SORT/MERGE Reference Manual* before using this command.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> [$]SORT */quals1*

FILE? *infilespec/quals2*

parameter definitions

/quals

One of the following:

Qualifier	Explanation
/OUTPUT:outfilespec	Specifies the output file. The default filetype is .SRT. If you omit /OUTPUT and the infilespec contains no version number, the output file is assumed to be the same as the input file, with the version number incremented. If you omit /OUTPUT but give a full infilespec, the output filespec is assumed to be exactly the same as the infilespec. That is, the version number is not incremented.
/ALLOCATION:n	Specifies the initial space allocation for the output file before the sort process begins. n is the number of bytes (decimal).
/BLOCK_SIZE:n	For magtape files only, specifies a non-standard tape block size. n is the number of bytes (decimal).
/BUCKET_SIZE:n	Specifies the RMS bucket size of the output file. n is the number of bytes (decimal).
/CONTIGUOUS	Specifies that the initial space allocation for the output file is to be contiguous.

Qualifier	Explanation
/DEVICE:device or /DEVICE:([device:]/quals)	For applications requiring control of the SORT scratch files, this qualifier specifies the scratch file device. device Scratch file device. /quals One or both of the following: /ALLOCATION:n /CONTIGUOUS
/FILES:n	For special applications, specifies the number of scratch files to be used by SORT (n must be between 3 and 8).
/FORMAT:format[:n]	Specifies the record format of the output file, if you specify /FORMAT, you must also specify format, even if it is unknown. • Format is one of: FIXED VARIABLE UNKNOWN • n is optional and specifies: Record length (with FIXED) Maximum record length (with VARIABLE or UNKNOWN).
/KEYS:(abm.n,.....) or /KEYS:(abm.n .....)	Defines the key fields to SORT, where: • a defines how to treat the data (that is, character, zone, and so on). The default is character. • b is the general sort order, where N is normal (ascending) O is opposite The default is N. m is the first position of key field. You must define this. n is the length of key field. You must define this. You can specify a maximum of 10 keys. The major key is the first in the string, and the minor key is the last. You can not specify this qualifier with /SPEC.
/PROCESS:x	Defines the type of SORT process, where x is one of the following: RECORD (default) TAG ADDRESS_ROUTING INDEX
/SEQUENTIAL	Specifies the file organization of the output file or /RELATIVE as sequential or relative.
/SPECIFICATION:file-spec	Control parameters for SORT are contained in the specified file. You cannot specify this qualifier with /KEYS. The default file type is .SRT.
infilespec	File specification of the file to be sorted. If you omit the file type, the system defaults to .SRT.

/quals2

One or both of the following:

Qualifier	Explanation
/FORMAT:format:n	Specifies the record format and length of input file. format can be FIXED, VARIABLE or UNKNOWN. You must specify this qualifier. n specifies record length for FIXED length records and the maximum length of VARIABLE or UNKNOWN structured records.
/INDEXED_SEQUENTIAL:n	Mandatory for an input file with indexed sequential organization, where n is the number of keys.

COMMAND VARIATIONS

Not applicable.

EXAMPLES

- Example 1:

```
PDS> SORT/KEY:C1.4
FILE? CAROL.DAT/FORMAT:UNKNOWN:130
```

  Sorts the file CAROL.DAT according to the characters in the key. The key is to be taken as characters and is in position 1 of the record and is 4 bytes long. Name the output (sorted) file as CAROL.DAT with incremented version number.

- Example 2:

```
PDS> SORT/SPEC:FRANK.SRT
FILE? MARTIN.DAT;3/FOR:FIXED:124/INDEXED:5
```

- Sorts the file MARTIN.DAT;3 according to the specifications held in FRANK.SRT. Name the output (sorted) file as MARTIN.DAT;3, to replace the input file.

- Example 3:

```
PDS> SORT/KEYS:(BN1.6 C8.2)/REL
FILE? TELEPHONE.LST/FORMAT:FIXED:40
```

- Example 4:

```
PDS> SORT/SPEC:STOCK.SRT/DEV:(/ALL:100/CO)
FILE? P12709.001/FORMAT:VAR:80
```

STOP

FUNCTION

The STOP command enables you to prevent all further processing within a file. You can use the STOP command only in an indirect command file or a batch command file. This command is ignored in interactive mode.

REQUIRED PRIVILEGE

ANY

FORMAT

PDS> [$]STOP *[/JOB]*

parameter definitions

/JOB
The only valid qualifier for PDS users and you can omit it.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

You can use the STOP command by itself or as the action in an ON command.

EXAMPLES

```
$JOB DEMO
$ON WARNING GOTO L10
$RUN JOB1
$GOTO L20
$L10: RUN TEST
$STOP
$L20: ON WARNING STOP
$RUN JOB2
$RUN JOB3
$EOJ
```

SUBMIT

FUNCTION

The SUBMIT command enables you to send a file containing batch commands to the batch processor.

REQUIRED PRIVILEGE

PR.SUB

FORMAT

PDS> SUBMIT *[quals]*

FILE? *filespec1[,....filespec n]*

parameter definitions

quals

One of the following:

Qualifier	Explanation
/AFTER:[(date]time[)]	Requests that the file is held in the spooler queue until the specified time, or date and time. When the specified time has passed, the file is queued for immediate processing. You specify time in the form hh:mm. You specify the date in the form dd-mmm-yy. The default value is 00:00; that is, the file is submitted for immediate processing.
/PRIORITY:pri	Priority at which you submit the file (for example, priority 1). pri must be between 1 and n. The System Manger determines n at system generation. The default value is 100.
/NOTRANSFER	Inhibits the copying of filespec to the spooling device.
filespec	Specification of a file containing batch commands. The specification must contain a file name. The default file type is .BIS.

COMMAND VARIATIONS

On a multiuser system, files are not transferred (/NOTRANS implied).

TECHNICAL NOTES

The system submits the file name of the file of batch commands to a queue of jobs for subsequent processing in batch mode. For every batch job run, a batch log is created and is given the name LP.SPR. The batch log is one (or more) concatenated file that results from running a batch job. The log is automatically spooled to CL:. You can identify which job a batch log refers to by looking at the banner pages that include the job name and user name.

On a timesharing system, unless filespec exists on a system device (that is, available to all timesharing users) or unless you specify /NOTRANSFER, filespec is automatically copied to device SP. You can use SUBMIT/NOTRANSFER only when the device where the filespec exists is still mounted when the job is dequeued.

EXAMPLES

- Example 1:

```
PDS> SUBMIT
FILE? BATCHFILE.BIS
```

- Example 2:

```
PDS> SUBMIT/PRIORITY:6
FILE? BATCHJOB
```

- Example 3:

```
PDS> SUBMIT/NOTRANSFER DK1:MYJOB,HISJOB
```

- Example 4:

```
$SUBMIT MYJOB
```

- Example 5:

```
PDS> SUBMIT/AFTER:14:00 BATCHJOB.BIS
```

TRUNCATE

FUNCTION

The TRUNCATE command enables you to truncate files back to their logical end-of-file point.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> [$]TRUNCATE

FILE? *filespec1[,filespec2...,filespecn]*

parameter definitions

filespec
Specification of the file to be truncated. Wildcards are allowed.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

RMS-11 files other than those that are fixed-length, variable-length, or sequenced cannot be truncated.

EXAMPLES

```
PDS> DIR A.A

DIRECTORY DB0:[1,1]
3-MAR-80 10:54

A.A;2    1.    03-MAR-80 10:53

TOTAL OF 1./10. BLOCKS IN 1. FILE

PDS> TRUNCATE
FILE? A.A

DIRECTORY DB0:[1,1]
3-MAR-80 10:54

A.A;2    1.    03-MAR-80 10:53

TOTAL OF 1./1. BLOCKS IN 1. FILE
```

The file A.A is truncated to the logical end of file. The extra blocks allocated to this file are freed.

TYPE

FUNCTION

The TYPE command enables you to print the contents of one or more specified files at your terminal. In batch, the file is output directly to the batch log.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> TYPE

FILE? *filespec1[,...filespecn]*

parameter definitions

filespec
File specification that must contain a file name and file type. Wildcards are allowed.

COMMAND VARIATIONS

Not applicable.

EXAMPLES

- **Example 1:**

```
PDS> TYPE
FILE? (BARLEY.CBL;2,GRAHAM.CBL;2)
```

- **Example 2:**

```
PDS> TYPE APPLE.DAT
```

- **Example 3:**

```
$TYPE FRED6.CBL
```

UNFIX

FUNCTION

The UNFIX command enables you to free a fixed task from memory.

REQUIRED PRIVILEGE

PR.RTC

FORMAT

PDS> UNFIX

TASK? *taskname* ESC

[TERMINAL? *terminal]*

parameter definitions

taskname
Installed name of the task to be unfixed from memory.

terminal
Terminal where the task is to be unfixed. The default is the current user's terminal.

COMMAND VARIATIONS

Not applicable.

EXAMPLES

- **Example 1:**

```
PDS> UNFIX JK03
```

- **Example 2:**

```
PDS> UNF FRED9 TT6
```

UNLOCK

FUNCTION

The UNLOCK command enables you to unlock a file that was locked as a result of being improperly closed.

REQUIRED PRIVILEGE

PR.FIL

FORMAT

PDS> [$]UNLOCK *[/FILE]*

FILE? *filespec1[,...,filespecn]*

parameter definitions

/FILE
The only valid qualifier for ordinary PDS users, and you can omit it.

filespec
Specification of the file you want to unlock. Wildcards are allowed.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

If a program using File Control Services (FCS) has a file open with write access and exits without first closing the file, the file is locked against further access as a warning that the information it contains is possibly invalid. Typically, in such a case, the following information has not been written to the file:

1 The current block buffer being altered.

2 The record attributes that contain the end of file information.

By using the UNLOCK command, you can access the file and can determine the extent of the damage and perhaps take appropriate corrective action.

EXAMPLES

```
PDS> UNLOCK
FILE? THAMES.MAC;7
```

VERIFY

FUNCTION

The VERIFY command enables you to verify the file structure on a device.

REQUIRED PRIVILEGE

PR.DEV

FORMAT

PDS> VERIFY *[/quals]*

DEVICE? *devicename*

parameter definitions

/quals

One of the following:

NOTE: All of the following qualifiers except /OUTPUT, /PRINT and /WORK_DEVICE are equivalent to switches in the file structure verification utility (VFY). The equivalent switch is given in brackets, and VFY is described in the *IAS Utilities Manual*.

Qualifier	Explanation
/OUTPUT:filespec	Specifies the file specification for the output listing.
/PRINT	Prints output on line printer.
/UNDELETE	Resets marked-for-delete indications (equivalent to /DE).
/FREE	Prints free space on volume (equivalent to /FR).
/LIST	Lists index file (equivalent to /LI).
/LOST_FILES	Enters lost files in the directory (equivalent to /LO).
/READ_CHECK[:n]	Reads every allocated block on the device to see if every block can be read. n is the blocking factor that indicates the number of file blocks to be read at a time (equivalent to /RC).
/RECOVER_BLOCKS	Restores blocks that are marked as allocated but not in use (equivalent to /RE).

Qualifier	Explanation
/UPDATE_BITMAP	Updates the bitmap to show all allocated blocks correctly.
/WORK_DEVICE:dev	Specifies the work device.

COMMAND VARIATIONS

Not applicable.

TECHNICAL NOTES

To verify a Files-11 volume, mount it using the /NOSHARED option to ensure that there is no activity on the volume.

EXAMPLES

- Example 1:

```
PDS> VER DB0:
```

- Example 2:

```
PDS> VER/FREE DK0:
```

- Example 3:

```
PDS> VER/UPDATE_BITMAP DK0:
```

Index

A

Abbreviating input • 4–1
Appending files
  extend access • 6–15

B

BASIC
  restrictions • 9–1
Batch job
  components of • 1–1
  submitting • 5–2
  terminating • 5–2
Batch mode • 8–1
Batch processing
  priority • 5–1
Batch queue • 5–1
Batch stream
  free • 5–1

C

COBOL object file
  default name • 10–2
  name specification • 10–2
COBOL object files
  generation • 10–3
Command and task privileges • 4–1
Command description layout
  command variations • 14–3
  examples • 14–3
  function • 14–3
  required privilege • 14–3
  technical notes • 14–3
Command descriptions
  brackets, round in • 14–1
  brackets, square in • 14–1
  brackets in • 14–1
  command names in • 14–1
  concatenation character in • 14–2
  continuation characters in • 14–2
Command descriptions (Cont.)
  dollar sign in • 14–1
  ellipses in • 14–1
  ! in • 14–2
  parameters in • 14–1
  parentheses in • 14–1
  qualifiers in • 14–2
Command qualifiers
  specification of • 11–2
Commands and parameters • 4–1
Command strings
  batch • 4–1
Comment character • 4–1
COPY command qualifiers • 6–16

D

Default mode • 5–2
Default print queue • 6–19
Defaults
  modifying, multiuser system • 6–6
  modifying, timesharing system • 6–6
Deferring printing • 6–19
Device defaults • 6–6
DISMOUNT
  parameters • 6–12

E

Editor
  batch-oriented • 5–3
Error conditions • 4–1

F

File
  definition of • 6–1
Files-11 • 6–1
File specification • 6–1

I

IAS file system
  default • 6–1
Information
  storing • 6–1
Interactive mode • 8–1
Interactive terminal
  layout • 2–1
  typewriter and • 2–1

L

LIBRARIAN command
  compress operation • 14–106
  create operation • 14–106
  delete operation • 14–106
  extract operation • 14–106
  insert operation • 14–106
  library types • 14–105
  list operation • 14–107
  modify header • 14–107
  replace operation • 14–107
  restrictions • 14–106
LINK command
  command qualifiers • 14–111
  file qualifiers • 14–111
  input files • 14–110
Linked COBOL task
  running • 10–5
Linking CORAL programs • 13–3
Logical device names • 6–5

M

MACRO command
  defaults
    listing file • 14–121
    object file • 14–121
  switches • 14–122
Macros
  using • 8–2
Media
  magnetic • 6–1
MERGE command
  target file • 6–18
MERGE command (Cont.)
  transaction file • 6–18
Mixed-mode processing • 1–1

N

Null device • 6–5

O

ON command
  default • 14–137
  restrictions • 8–7
Options
  termination of • 10–3

P

PDS file organization commands • 6–3
Printing
  deferred • 6–10
Privileges
  command • 4–1
  task • 4–1
Pseudo-devices • 6–5

Q

Qualifiers • 4–1

R

RMS file organization • 6–3

S

SET command
  batch password • 14–171
  bootstrap • 14–171
  default, timesharing • 14–170
  echo • 14–172

SET command (Cont.)
  endoffile • 14–171
  password • 14–170
  priority • 14–171
  real-time control–timesharing • 14–171
  sci • 14–172
  terminal • 14–171
  UIC–multiuser • 14–171
SHOW command
  devices • 14–175
  memory • 14–176
  status • 14–176
Source code
  storing • 8–2
SUBMIT command • 5–1
SYSRES • 10–4
System resident library • 10–4

T

Target file • 6–18
Task execution • 12–4
Terminals
  interactive • 2–1
Transaction file • 6–18

U

UFD
  directory • 6–14
UFD defaults • 6–6
User name
  parameters • 5–2

Reader's Comments

This form is for document comments only. Digital will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well organized? Please make suggestions for improvement.

Did you find errors in this manual? If so, specify the error and the page number.

Please indicate the type of user/reader that you most nearly represent:

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify)_____

Name_____Date_____

Organization_____

Street_____

City_____State_____Zip Code_____
or Country

Do Not Tear - Fold Here and Tape

digital™

No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO 33 MAYNARD MASS

POSTAGE WILL BE PAID BY ADDRESSEE

IAS Engineering/Documentation
Digital Equipment Corporation
5 Wentworth Drive GSF/L20
Hudson, NH 03051-4929

Do Not Tear - Fold Here