CVSI

CVSI, Inc.
4G Crosby Drive
Bedford, MA 01730

Operator's Guide to the Spooler Subsystem

Revision 23.3

DOC9303-5LA

Operator's Guide to the Spooler Subsystem

Fifth Edition

Rhonda Bengiovanni

This manual documents the software operation of the PRIMOS operating system on 50 Series computers and their supporting systems and utilities as implemented at Master Disk Revision Level 23.3 (Rev. 23.3).

Prime Computer, Inc., Prime Park, Framingham, Massachusetts 01701

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

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

PRIME, PR1ME, PRIMOS, and the Prime logo are registered trademarks of Prime Computer, Inc. 50 Series, 400, 750, 850, 2250, 2350, 2450, 2455, 2550, 2655, 2755, 2850, 2950, 4050, 4150, 4450, 5310, 5320, 5330, 5340, 6150, 6350, 6450, 6550, 6650, 9650, 9655, 9750, 9755, 9950, 9955, 9955II, DISCOVER, PRIME EXLNET, Prime INFORMATION CONNECTION, PRIME/SNA, PRIME EXL, PRIME EXL MBX, INFO/BASIC, MIDAS, MIDASPLUS, PERFORM, PERFORMER, PRIFORMA, PRIMAN, Prime INFORMATION, INFORM, PRISAM, Prime INFORMATION PLUS, PRIMELINK, PRIMIX, PRIMENET, PRIMEWAY, PRODUCER, PRIMEWORD, Prime INFORMATION EXL, Prime INFORMATION/pc, PRIME TIMER, RINGNET, SIMPLE, PT25, PT45, PT65, PT200, PT250, and PST 100 are trademarks of Prime Computer, Inc. PrimeService is a service mark of Prime Computer, Inc.

Adobe and PostScript are registered trademarks of Adobe Systems Incorporated.
Apple, AppleTalk, LaserWriter, and Macintosh are registered trademarks of Apple Computer, Inc.
Datapac is a registered trademark of Bell Canada.
Dataproducts is a trademark of Dataproducts Corporation, 6200 Canoga Avenue, Woodland Hills, CA 91365.
Diablo is a registered trademark of Diablo Systems, Inc., a subsidiary of Xerox Corp.
Dynapac is a trademark of Dynapac Systems, 5350 Shawnee Road, Alexandria, VA 22312.
Ethernet is a registered trademark of Xerox Corp., 800 Long Ridge Road, Stamford, CT 06904.
FUJITSU is a trademark of FUJITSU America Incorporated, San Jose, CA 95131.
Gandalf Pin is a trademark of Gandalf Data, Inc, 1019 S. Noel, Wheeling, IL 60090.
Gould is a trademark of Gould, Inc, 3631 Perkins Avenue, Cleveland, OH 44114.
HP LaserJet is a registered trademark of Hewlett-Packard Company.
NEC is a trademark of NEC Information Systems, 1414 Massachusetts Avenue, Boxborough, MA 01719.
Printronix is a trademark of Printronix, P.O. Box 19559, Irvine, CA 92713.
QMS is a trademark of Quality Micro Systems Inc, P.O. Box 81250, Mobile, AL 36689.
QUME is a trademark of ITT Qume Corp., 2350 Qume Drive, San Jose, CA 95131.
Scribe is a registered trademark of UniLogic Ltd.
TELENET is a registered trademark of GTE Telenet Communications Corp.
Versatec is a trademark of Versatec, Inc, 2710 Walsh Avenue, Santa Clara, CA 95051.

Printing History

First Edition (DOC9303-1LA) February 1985 for Rev. 19.4
Second Edition (DOC9303-2LA) July 1987 for Rev. 21.0
Third Edition (DOC9303-3LA) October 1988 for Rev. 22.0
Fourth Edition (DOC9303-4LA) March 1990 for Rev. 23.0
Fifth Edition (DOC9303-5LA) May 1992 for Rev. 23.3

Credits

Editorial: Barbara Bailey, Irene Rubin
Illustration: Elizabeth Wahle
Production: Judy Gordon
Project Support: Joan Karp, Glenn Morrow
Technical Support: Joe Coelho, George Gove, Don Markuson

Contents

■ ■ ■ ■ ■ ■ ■

About This Book

1 Overview of the Spooler Subsystem

2 Setting Up the Spooler Subsystem

4 Controlling Printer Formatting

5 Tailoring Spooler Information

6 Managing Spooler Operations

7 Resolving Operational Problems

Appendices

A *SPOOL and PROP Commands*

B *Writing a Print Handler*

C Spooler Subsystem Messages

D Defining and Using PostScript Printers

E Defining HP LaserJet Printers

F PDN Support Directive

Index

About This Book

The *Operator's Guide to the Spooler Subsystem* is one of a set of Operator's Guides intended for Operators, System Administrators, and other people with responsibility for managing the operations of a computer system.

The Spooler subsystem enables users to send files to designated printers, and enables the Spool Administrator to manage printers and printing capabilities efficiently. This book provides information about

- Spooler subsystem components
- Setting up the Spooler subsystem
- Managing the Spooler subsystem
- Defining Spooler tasks, procedures, and schedules
- Using the Spooler subsystem
- Defining printer environments
- Tailoring Spooler information
- Controlling printer formatting
- Writing print handlers

This book assumes that you are familiar with Prime® systems and the PRIMOS® operating system. If you have not used Prime systems before, read the *PRIMOS User's Guide* which explains the PRIMOS file management system and introduces essential commands and utilities. The *PRIMOS Commands Reference Guide* also contains introductory information, and is a dictionary of PRIMOS user commands. These manuals contain information for users on using the SPOOL and PROP commands.

Once you become more familiar with Prime systems, read the *Operator's System Overview* which outlines information in the Operator's Guide series. You can then choose to read other books in the Operator's Guide series as they apply to your system responsibilities.

Organization of This Book

This book contains seven chapters and six appendices.

- Chapter 1, Overview of the Spooler Subsystem, describes the Spooler subsystem and its components. It introduces the commands you use to print files, cancel or change print requests, and manage and operate the Spooler subsystem and printers. Examples of how you use these commands to perform basic Spooler operations are included.
- Chapter 2, Setting Up the Spooler Subsystem, describes how to install and secure the Spooler subsystem directories and files, set up and secure the spool queue and spool data directories, determine your printing requirements, consider alternative methods of setting up and using printers, and suggests ways of managing printer operations. This chapter also provides information on using pre-Rev. 21.0 spool queues and converting pre-Rev. 21.0 printer environments.
- Chapter 3, Defining Printer Environments, explains how to create printer environment files for different types of printers and their printing capabilities, and explains how to use printer environments to match requests with the correct printers.
- Chapter 4, Controlling Printer Formatting, explains how to print files in different character sets and how to use the Electronic Vertical Forms Unit (EVFU) to print information on special forms.
- Chapter 5, Tailoring Spooler Information, explains how to create custom formats for displaying spool queue information, expand despooler log information, create user address information for distributing printed output, and design custom header pages. Information on using multilanguage despoolers is also included in this chapter.
- Chapter 6, Managing Spooler Operations, provides procedures and examples for common Spooler operations that you perform using the SPOOL and PROP commands.
- Chapter 7, Resolving Operational Problems, explains how to identify and resolve hardware and software problems that may occur.
- Appendix A, SPOOL and PROP Commands, is a reference of the SPOOL and PROP commands and their options.
- Appendix B, Writing a Print Handler, explains how to add support for nonstandard output devices.
- Appendix C, Spooler Subsystem Messages, lists and explains messages that the Spooler subsystem displays.
- Appendix D, Defining and Using PostScript Printers, describes the PostScript® fonts, printing options, and printer environments.

- Appendix E, Defining HP LaserJet Printers, describes the HP LaserJet® fonts and printer environment directives.
- Appendix F, PDN Support Directive, describes how you can use the Spooler subsystem to send files to printers that are attached to a Public Data Network (PDN). The software that enables you to print files over a PDN is available as a separately priced option.

Changes at Rev. 23.0

- The module for displaying the spool queue entries can be replaced by your own routine. If you do not create your own routine, the existing routine called by SPOOL –LIST is used.
- You can create ASCII files containing entries that define the spool queue information displayed by SPOOL –LIST –BRIEF operations.

Changes at Rev. 23.1

- A new MPC4000 environment directive enables you to configure the data line connection from the new MPC4000 high-speed print controller to a Dataproducts™, Centronics, or PostScript printer.
- A new option to the ASYNC data line configuration directive, –MAXIMUM_SHUTDOWN_TIME *nn*, enables you to configure the number of seconds the despooler waits before shutting down the Spooler when the print buffer is full.
- A new print handler routine, S$START_COPY, is called by the despooler phantom before it prints each copy of a file. This allows you to control laser printers that are capable of double-sided printing among other things.

Changes at Rev. 23.2

- Font options for PostScript printers are available.
- Attributes are available with the –SET_LANDSCAPE and –SET_PORTRAIT options of the SPOOL command (and environment file directives) for PostScript printers only. The new attributes enable *n*-up printing and draft copy printing. A subset of the *n*-up attributes can be used with the –PROC command for text files.

- An option to the SPOOL –NO_FORMAT option, –AT TEXT, allows you to disable PostScript page formatting when spooling text files to PostScript printers.

Changes at Rev. 23.3

- Support for the HP LaserJet family of printers, including a new DEVICE directive, and new attributes for setting paper type, fonts, and page sizes is available.
- A new environment directive, ERROR_NOTIFY, directs the Spooler to send printer error messages to the user names listed in the ERROR_NOTIFY directive (up to eight users), on both local and remote systems. This directive is for use only with printers connected over ASYNC and TCP/IP.
- Up to eight aliases can be listed within printer environment files for MANDATORY attributes, and more than one alias can be included on an attribute line.
- The despoolers now utilize large packet sizes for TCP/IP instead of the one line per packet. Although this feature is not user-visible, you should see a decrease in the use of TCP/IP packets.
- New options to the PROP command, –MAX_SIZE and –MIN_SIZE, enable Spool Administrators (or .SPOOL_ADMINISTRATOR$ group members) to control the size of print jobs allowed into the local spool queue. The new PROP option, –REPORT_SIZE, displays information on minimum size and maximum size settings. A new suboption to the PROP –COLDSTART command, –NO_QUERY, bypasses the query message response that is required before shutting down an environment.
- The new environment directives MAX_SIZE and MIN_SIZE enables Administrators to control the size of jobs processed by particular printer environments.
- Spool Administrators can now give access rights to specific users to use the PROP command to abort, back up, continue, drop, hang, line up, release, reset, restart, start, stop, and suspend printer environments.
- Two new DEVICE directive options, –POSTSCRIPT_CHECK and –POSTSCRIPT_OVERRIDE, perform the following functions:

  –POSTSCRIPT_CHECK appends the –NO_FORMAT option of the SPOOL command when spooling PostScript language files to PostScript printers.

–POSTSCRIPT_OVERRIDE removes a –NO_FORMAT option given with the SPOOL command when spooling text files (non-PostScript) to PostScript printer, thus preventing PostScript errors.

- Two new TCP/IP directive options, –DEFER_TIME and –RELEASE, alternate printing of jobs from two environments that share the same printer.

Documentation for System Administrators and Operators

The following Prime documentation contains detailed information for System Administrators and Operators:

- *System Administrator's Guide, Volume I: System Configuration* (DOC10131-3LA) explains how to set up a system and allocate system resources.
- *System Administrator's Guide, Volume II: Communication Lines and Controllers* (DOC10132-2LA with RLN10132-21A) explains how to configure communications lines.
- *System Administrator's Guide, Volume III: System Access and Security* (DOC10133-3LA) explains PRIMOS security features and how to prevent unauthorized use of your system.
- *Operator's System Overview* (DOC9298-3LA) introduces the Operator's Guide series and describes computer-room operation of Prime systems.
- *Operator's Guide to System Monitoring* (DOC9299-3LA) describes how to monitor system usage, activity, and messages.
- *Operator's Guide to File System Maintenance* (DOC9300-6LA) describes the PRIMOS file system, and explains how to format disk partitions, run the disk partition maintenance program, determine physical device numbers, and interpret disk error messages.
- *Operator's Guide to Data Backup and Recovery* (DOC10324-1LA with UPD10324-11A and UPD10324-12A) describes how to save information on disk or tape, restore that information when it is needed, generate indexes, create and use a boot tape, and how to shut down the system before starting the save or restore, and restart the system once the save or restore is complete.
- *Operator's Guide to System Commands* (DOC9304-6LA) is a reference guide containing most of the operator commands and their options in alphabetical order.
- *Operator's Guide to the Batch Subsystem* (DOC9302-3LA) describes how to set up, monitor, and control the Batch subsystem.
- *Programmer's Guide to BIND and EPFs* (DOC8691-1LA with UPD8691-11A and UPD8691-12A) describes the BIND linker and Executable Program Formats (EPFs).

Other Prime User and Hardware Documentation

The following documentation is listed below for your reference:

- The documentation for your CPU, and the Using Your series of books.

- The documentation for letter-quality printers, *LQP 3185 Guide* (DOC7634-001L) and *LQP 3175 Guide* (IDR5024).
- *PRIMOS Commands Reference Guide* (DOC3108-8LA) provides detailed information on user commands.

Printer Documentation

The following printer documentation is listed below for your reference:

- *Prime 3200 Series High Performance Band Printers Operator's Manual* (MAN11318-1LA)
- *Laser Printer Operator's Manual, Model 3410* (MAN10069-1XA)
- *Laser Printer Quick Reference Guide, Model 3410* (MAN10079-1XA)
- *Prime 3450 Laser Page Printer User's Guide* (MAN11277-1LA)
- *Prime 3451 PostScript Laser Page Printer User's Guide* (MAN11276-1LA)
- *Diablo 630 and IBM Graphics Printer Personality Reference Manual, Model 3410* (MAN10072-1XA)

Prime Documentation Conventions

The following conventions are used throughout this document. The examples in the table illustrate the uses of these conventions.

Convention	*Explanation*	*Example*
Uppercase	In command formats, words in uppercase bold indicate the names of commands, options, statements, and keywords. Enter them in either uppercase or lowercase.	**PROP –VERIFY**
Italic	Variables in command formats, text, or messages are indicated by lowercase italic.	**SPOOL** ***myfile***
Abbreviations	If a command or option has an abbreviation, the abbreviation is placed immediately below the full form.	**–ATTRIBUTE** **–ATT**
Brackets	Brackets enclose a list of one or more optional items. Choose none, one, or several of these items.	$\textbf{–STATUS} \begin{bmatrix}\textbf{–ALL}\\ \textbf{–NO\_WAIT}\end{bmatrix}$

Convention	*Explanation*	*Example*
Braces	Braces enclose a list of items. Choose one and only one of these items.	PROP $\left\{\begin{array}{l}\text{–MAX } [\textit{nnnn}]\\ \text{–MIN } [\textit{nnnn}]\end{array}\right\}$
Braces within brackets	Braces within brackets enclose a list of items. Choose either none or only one of these items; do not choose more than one.	–HANG $\left[\left\{\begin{array}{l}\text{–FINISH}\\ \text{–NOW}\\ \text{–IDLE}\end{array}\right\}\right]$
Monospace	Identifies system output, prompts, messages, and examples.	`-- More --`
Underscore	In examples, user input is underscored but system prompts and output are not.	`OK, PROP -STATUS`
Hyphen	Wherever a hyphen appears as the first character of an option, it is a required part of that option.	SPOOL –LIST
Ellipsis	An ellipsis indicates that you have the option of entering several items of the same kind on the command line.	–ATT *name* [*name* . . .]
Octal	The symbols ' or ^ indicate that the number is given as an octal value.	'240 or ^240 is eqivalent to 160 (base 10).

1 Overview of the Spooler Subsystem

Introduction

This chapter provides an overview of the Spooler subsystem and its components. The following topics are introduced in this chapter and explained in more detail in the remaining chapters:

- Spooler subsystem components
- Setting up and using printers
- Printing files with the SPOOL command
- Checking the progress of spool requests
- Canceling a spool request
- Modifying a spool request
- Controlling printer operations with the PROP command
- Overview of Spooler subsystem tasks

Spooler Subsystem Components

This section describes the components of the Spooler subsystem and how you use them to set up and manage Spooler operations and your systems printers. The Spooler subsystem consists of three top-level directories which must be protected by ACLs to secure your data. These directories and their files are described briefly below, and in more detail in Chapter 2, Setting Up the Spooler Subsystem.

The SPOOL_DATA* Directory

This directory contains spooled files. When you issue the SPOOL command, the spooled file is copied to this directory (unless you use the –NO_COPY or –SPOOL_WHILE_OPEN option with the SPOOL command).

The SPOOL_QUEUE* Directory

This directory contains the QUEUE file, which is not an ASCII file. This file contains a list of spool requests waiting to be printed. The SPOOL_QUEUE* directory may also contain two optional files:

FULL_LIST_USERS

Contains a list of users to whom the Administrator has given rights to view all spool queue entries using the SPOOL –LIST command. The Administrator may give these rights to all users by adding the entry .ALL_USERS. in the FULL_LIST_USERS file, or to specific users by adding their user IDs to the file.

DATA_PARTITIONS

Is a file that you create to enter the names of disk partitions containing a SPOOL_DATA* directory. You can have more than one SPOOL_DATA* directory if needed. The SPOOL command copies spooled files to the SPOOL_DATA* directory residing on the first partition listed in the DATA_PARTITIONS file. If the first partition becomes full, or if an error occurs that prevents the file from being copied to the first partition, the SPOOL command copies files to the next partition, and so on. (Refer to the section Using Multiple SPOOL_DATA* Directories in Chapter 2.)

The SPOOL* Directory

This directory contains all other Spooler subsystem files and subdirectories that you use to define the operation of and services performed by each printer. This directory also contains files for custom spool list displays and files used for monitoring Spooler activities. The SPOOL* directory contains the following files and subdirectories:

Printer environment files

Contain entries that define the operation and setup of each printer. Printer environment filenames have a .ENV extension.

RESTRICTED_USERS subdirectory

Contains the SWO_VALID_USERS file. You create this subdirectory and its file. The file lists privileged users to whom you have given access rights to spool a file while it is open.

ATTRIBUTES subdirectory

Contains attributes files that you create to enable the SPOOL command to verify attributes given with spool requests and supply default attributes when none are given on the command line. You can create attributes files for individual users and projects, and for all users by creating a .DEFAULT_ATTRIBUTES. file.

LOG subdirectory

Contains printer environment log files. Each log file lists the printer environment information, and logs despooler entries processed by that environment, when LOG is enabled.

CONFIG_LIST subdirectory

Contains the binary version of ASCII files that users create using the CREATE_CONFIG_LIST program to produce custom SPOOL –LIST displays. Refer to the section Creating a Custom Spool List Display in Chapter 5 for more information.

USER_ADDRESSES file

Contains the physical address locations of users for distributing printed output.

Setting Up and Using Printers

Before you can use your printers, you must first determine the printing needs of the users on the system and decide how best to use each printer connected to the system. Once you gather this information, you will use it to create one or more environment files for each type of printer.

An environment file contains **environment directives** which are entries that define the type of printer, the hardware interface, queue search intervals, and printing capabilities that you wish to implement on a particular printer.

Each printer and its associated environment are controlled by a process called a despooler phantom. A **despooler phantom** searches the spool queues andprocesses requests whose options and attributes match those defined in itsassociated printer environment file. There can be 128 despooler phantoms running on a system but each phantom can service only one printer and printer environment at any one time. Refer to Chapter 3, Defining Printer Environments, for more information

Environment files can contain any of the following five types of directives:

- **Interface definition directives** define the data line connection hardware.
- **Scheduling requirement directives** establish the order in which jobs are printed and by which despooler phantoms. Scheduling requirement directives also list the attributes supported by a particular printer.
- **Queue search directives** determine which spool queues are searched for print jobs and how often they are searched.
- **Formatting option directives** establish page format, font, and other print formatting options.
- **Logging despooler directives** determine the level of despooler activity and error reporting.

Attributes are one of the scheduling requirement directives that you use to define the type of printer and the printer setup such as the paper type, formatting options, and other capabilities supported by a particular printer. The despooler phantom checks the attributes given with each spool request and processes the requests whose attributes match those in its associated environment file.

For some printing environments, you may determine that certain features of the printer or its setup are *required* for spooling to that printer. You can define **mandatory attributes** using the –MANDATORY option of the ATTRIBUTE directive. If the environment file contains any mandatory attributes, users must include all of the mandatory attributes when spooling to that environment or the file will not print. One example of a mandatory attribute might be a special form type loaded on the printer or the name of a print handler program that controls the printer.

For more information on creating environment files, refer to Chapter 3, Defining Printer Environments.

Printing Files With the SPOOL Command

This section gives an overview of the SPOOL command and how you use it to spool files for printing, check the progress of requests, cancel spool requests, and modify requests in the queue. This section also describes how the Spooler processes requests for different printers.

Appendix A describes the SPOOL command options. The *PRIMOS User's Guide* and the *PRIMOS Commands Reference Guide* contain information for users on how to use the SPOOL command and its options.

To spool a file for printing, you issue the SPOOL command at the PRIMOS command line prompt using the following format:

SPOOL *pathname* [*options*]

pathname is the qualified name of the file you want to print (or a simple filename if the file is located in your current directory). The SPOOL command has many options that you can use to select the type of paper, page format, and other capabilities that you may need to use.

The SPOOL command options enable you to

- Submit a spool request.
- Cancel a spooled file.
- Modify a spool request.
- List your print requests in the spool queue.
- Defer printing of spool requests to some other time.
- Request multiple printed copies of the file.

Selecting a Printer

When spooling a file, you must include at least one attribute that selects a particular printing device. You may include other attributes to use other printing options, provided that they are defined in the environment file for that printer.

In the example below, the SPOOL command includes an attribute that selects a printer named LQP1:

```
OK, SPOOL REPORT1 -ATTRIBUTE LQP1
```

The SPOOL command verifies the attributes and notifies the user of an error if you have created special files called attributes files. **Attribute files** are files that you create to establish default attributes for all users, specific users, and projects, and which the SPOOL command supplies if no attributes are given with the SPOOL command.

Refer to Chapter 2, Setting Up the Spooler Subsystem, for more information on creating attribute files.

Entering Spool Requests in the Spool Queue

When you issue the SPOOL command with an existing filename or pathname, the despooler phantom copies the spooled file into the SPOOL_DATA* directory and enters the spool request into the spool queue (in the SPOOL_QUEUE* directory). Each system has one spool queue that lists the details of spool requests waiting to be printed. When the spool request is entered into the spool queue, the SPOOL command informs the user that the request was added to the spool queue and displays a number identifying the request.

If you spool a file using either the –NO_COPY or –SPOOL_WHILE_OPEN option, the spooled file is not copied to the SPOOL_DATA* directory. (Refer to Appendix A for more information on the SPOOL command options.)

The following example shows the notification message displayed upon entering a spool request:

```
OK, SPOOL RUMCAKE -ATTRIBUTE TP_QUM
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
Request 56 added to queue, 1 records: <CRUMB>RHONDA>RECIPE>RUMCAKE
OK,
```

Processing Requests in the Spool Queue

Once files are placed in the spool queue, they are printed by one of several despooler phantoms running concurrently on the system. Each despooler phantom searches the spool queues at predefined intervals for requests that it can print. The despooler phantoms can search both local spool queues and those on other systems in the same network. Each despooler phantom determines which files to print by checking the attributes of the spool requests against those in its associated environment file.

The SPOOL command also checks the active environment files and notifies the appropriate despooler phantom of requests that it can print. If the appropriate despooler phantom is not currently printing a file, the SPOOL command wakes up the despooler phantom and it prints the file immediately. If the despooler phantom is processing a file, the request waits in the queue until the despooler phantom is finished.

You can use the –ON *sysname* option with the SPOOL command to wake up a remote despooler phantom that is idle. Figure 1-1 illustrates how spool requests are entered into the spool queue.

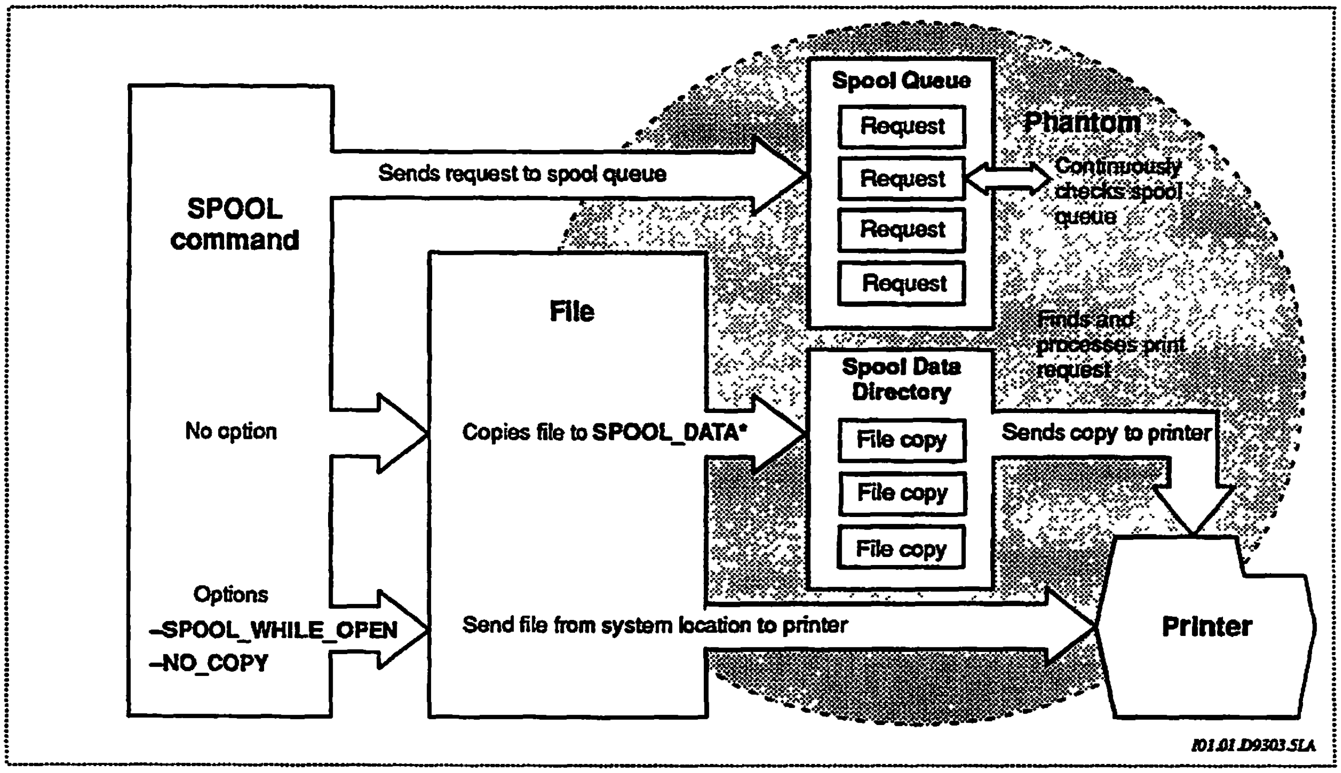

Figure 1-1. Spooling a File

Recognizing SPOOL Command Errors

The SPOOL command checks the filename or pathname you enter to determine if it exists. The SPOOL command also verifies the attributes given with the spool request (if attributes files have been created). If the filename or pathname does not exist, or if any attributes supplied with the SPOOL command are not valid, an error message displays immediately and the request is not entered into the spool queue.

The SPOOL command does not verify all attributes that you either include or omit in your spool request. If you include attributes that are not defined in the environment file for the selected printer or if you do not supply any mandatory attributes defined in the environment file, the request is entered into the queue but the file is not printed.

Each despooler phantom searches the spool queues for requests whose attributes match those defined in its associated environment file. If a spool request contains attributes that cannot be processed by the despooler phantom controlling the selected printer, the request remains in the queue until you cancel or modify it, as explained later in this chapter. The file may be processed by another despooler phantom controlling a printer of the same type if the attributes in its associated environment file match those given with the request.

Because not all mistakes are recognized as errors by the SPOOL command, the Spool Administrator must inform users how to use the printers and provide them with the correct printer environment names, acceptable attributes, and any mandatory attributes for each printer. Users should also know how to check their spool queue entries and modify them if necessary. Refer to the section in Chapter 2 called Using Attributes.

The following sections provide an overview of the SPOOL command options for checking the progress of requests in the spool queue, canceling a spooled file, and modifying a spool request.

Checking the Progress of Spool Requests

To check the progress of requests in the spool queue, issue the SPOOL –LIST command as shown below:

```
OK, SPOOL -LIST
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]

System SYSA
Request  Time  User             File                   No Size  State
-------- ----- ---------------- ---------------------- -- ----- -----
12 March 92
909      07:34 BOBBY            MONTH.END.RPT          1  6     Print
1013     09:18 KARENC           KAREN.MAIL             1  2
1017     09:47 SDO              LISTING.3              1  3
1018     09:47 SDO              PHONE.CALLS            1  3
1046     13:44 IRENE            MAIL.SPOOL.TEMP        1  3
OK,
```

The SPOOL –LIST command displays the number assigned to the spool request, the time it was entered into the spool queue, the user ID under which the requested was spooled, the filename, the number of copies (one copy by default), the file size in records, and the status of the spool request. Notice in this example that the requests are listed in order of the date and time they were entered into the queue and that the first file is printing.

The following example shows the SPOOL –LIST display using the –DETAIL option. The –DETAIL option gives you more information about each request, including the attributes given with each request.

```
OK, SPOOL -LIST -DETAIL
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]

System SYSA
Request  Time       User                     Copies Size  State
-------- ---------- ------------------------ ------ ----- -------------
1065     15:27:32   RHONDA                            1     1

   File name   <CRUMB>RHONDA>RECIPE>FUDGE
   Attributes  TP_QUM

1066     15:28:08   RHONDA                            1     1
   File name   <CRUMB>RHONDA>RECIPE>RUMCAKE
   Attributes  TP_QUM
   Deferred until 17:00:00 on 13 March 92

1067     15:28:32   RHONDA                            1     1
   File name   <CRUMB>RHONDA>RECIPE>BREAD
   Attributes  TP_QUM, WHITE

OK,
```

In this example, request 1066 was issued with the –DEFER option of the SPOOL command to defer printing until 17:00:00 (or 5:00 PM). Each request in this example was spooled with the device attribute TP_QUM. Request 1067 lists an additional attribute, WHITE, which requests a particular paper type.

Canceling a Spool Request

This section explains how to cancel a spool request and delete it from the spool queue. The Spool Administrator, privileged users (the supervisor terminal), and members of the .SPOOL_ADMINISTRATOR$ ACL group, can cancel any pending print requests and stop any file that is currently printing on the local system. General users can cancel only their own requests.

Users may wish to cancel a spool request for any of the following reasons:

- The file was spooled to the wrong printer.
- The file needs additional changes before printing.
- The file is spooled with the wrong format.
- The wrong file was spooled.

To cancel a request and delete it from the spool queue, enter the following:

SPOOL –CANCEL *request-number* [*request-number* . . .]

request-number is the number of the request you wish to cancel. You obtain the request number by issuing the SPOOL –LIST command, as explained in the previous section and shown below:

```
OK, SPOOL -LIST
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]

System SYSA
Request  Time   User             File                     No Size  State
-------- ----- ---------------- ------------------------ -- ----- -----
56       11:13 RHONDA           RUMCAKE                  1      1
57       11:18 RHONDA           MUSHROOM.SOUP            1      2   Print
58       11:19 RHONDA           FUDGE                    1      1
59       11:19 RHONDA           PUNCH                    1      1
OK,
```

The following example shows the use of the SPOOL –CANCEL command to cancel request 59 from the spool queue listing above. A message displays confirming that request 59 is canceled.

```
OK, SPOOL -CANCEL 59
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
Request 59 cancelled
OK,
```

The example below shows the spool queue listing after request 59 is canceled. Note also that request 57 finished printing and is removed from the spool queue.

```
OK, SPOOL -LIST
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]

System SYSA
Request  Time   User             File                     No Size  State
-------- ----- ---------------- ------------------------ -- ----- -----
56       11:13 RHONDA           RUMCAKE                  1  1
58       11:19 RHONDA           FUDGE                    1  1
OK,
```

Note that the remaining spool queue request numbers are not renumbered. Each spool request entered into the queue is given the next consecutive number. You can cancel multiple spool requests on the same command line by separating each request number with a space. Users can cancel their own requests that have already started printing on the local system. Refer to Chapter 4, Controlling Printer Formatting, for more information.

Modifying a Spool Request

This section explains how to modify a request using the –MODIFY option of the SPOOL command. The SPOOL –MODIFY command is useful if you need to change some part of a spool request after it has been entered into the spool queue. You may wish to defer printing to a later time or include another attribute, for example.

To modify a spool request, issue the SPOOL –MODIFY command in the following format:

SPOOL –MODIFY ***request-number*** **[*options*]**

request-number is the number of the spool request you wish to modify. *options* are any of the SPOOL printing options that you wish to add or change in your spool request (except –SPOOL_WHILE_OPEN, –NO_COPY, –OPEN, or –TUNIT).

The detailed spool queue listing below shows the attributes of each spool request in the queue. The requests in this example have a valid device attribute, TP_QUM, but they will not print without the mandatory attribute CARBON for this particular printer.

```
OK, SPOOL -LIST -DETAIL
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]

System SYSA
Request  Time       User                          Copies Size  State
-------- ---------- ----------------------------- ------ ----- -------------
56       11:13:48   RHONDA                        1      1
   File name    <CRUMB>RHONDA>RECIPE>RUMCAKE
   Attributes   TP_QUM

58       11:19:36   RHONDA                        1      1
   File name    <CRUMB>RHONDA>RECIPE>FUDGE
   Attributes   TP_QUM

87       14:02:44   RHONDA                        1      2
   File name    <CRUMB>RHONDA>RECIPE>MUSHROOM.SOUP
   Attributes   WHITE, TP_QUM
OK,
```

The following example uses the SPOOL –MODIFY command to modify request 87 to add the mandatory attribute CARBON and defer printing to a later time.

```
OK, SPOOL -MODIFY 87 -ATT CARBON -DEFER 15:00
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
OK, SPOOL -LIST -DETAIL
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]

System SYSA
Request  Time       User                       Copies Size  State
-------- ---------- -------------------------- ------ ----- -------------
39       10:13:24   JOAN                       1      5
   File name    <CRUMB>JOAN>MAIL.SPOOL.TEMP
   Attributes   Q, ST2

56       11:13:48   RHONDA                     1      1
   File name    <CRUMB>RHONDA>RECIPE>RUMCAKE
   Attributes   TP_QUM

58       11:19:36   RHONDA                     1      1
   File name    <CRUMB>RHONDA>RECIPE>FUDGE
   Attributes   TP_QUM

87       14:02:44   RHONDA                     1      2
   File name    <CRUMB>RHONDA>RECIPE>MUSHROOM.SOUP
   Attributes   CARBON
   Deferred until 15:00:00 on 16 March 92
OK,
```

Notice that request 87 no longer has the attributes TP_QUM or WHITE, as shown in the previous example. If you modify the attributes of a request with the –ATT option, you must reissue all of the attributes for the request.

Controlling Printer Operations With the PROP Command

This section gives an overview of how you use the PROP command to control the operation of printers and printer environments. The acronym PROP is derived from printer operations. Chapter 4 describes the use of PROP in more depth for managing most Spooler and printer operations.

The PROP command is primarily used by the Spool Administrators, Operators, and other privileged users to perform the following Spooler operations:

- Monitor the status of printers and environments.
- Control despooler phantoms.
- Start, stop, display, and perform other printer environment operations.
- Interrupt printing jobs and restart them later.
- Control the size of spool requests entered into the queue.

The PROP command is used most frequently for starting and stopping printer environments, displaying the contents of a particular environment file, and checking the status of available printers. The following sections explain how to use the PROP command for displaying, checking, stopping, and starting the status of printer environments.

Displaying a Printer Environment

Any user can display a printer environment by entering the following command:

PROP ***environment-name*** **–DISPLAY**

The following example shows the use of PROP to display the printer environment TP_QUM:

```
OK, PROP TP_QUM -DISPLAY
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
TP_QUM       Printing (MAIL.SPOOL.TEMP: Page 4, Copy 1 of 1
                           Request: 89)

/* QUME (TPUBS/CARBON) PRINTER FOR SYSA
/* Name of environment file is SPOOL*>TP_QUM.ENV
/*
Attribute TP.QUM
Attribute TP_QUM
Attribute TPUBS
Attribute CARBON -Mandatory   /* CARBON must be specified.
Device QUME -Evfu_off
Priority_size 10              /* Print files less than 10 records first.
Node -Local                   /* Scan only SYSA's spool queue.
Format -Width 84 -Length 66 -Bottom_margin 4  /* Sets format of pages.
Header -RBC                   /* Use different characters for header page.
Message TPUBS Qume Printer
Warning -On
Async -Line 108 -Protocol TTY -Speed 1200
Log -Off
OK,
```

Notice that the –DISPLAY option also shows you the status of the printer in the first line. Here, TP_QUM is printing a file called MAIL.SPOOL.TEMP, and is currently on Page 4. Copy 1 of 1 indicates that only one copy of the printed file was requested. Refer to Chapter 3 for more information on printer environment files.

Checking the Status of Printer Environments

The –STATUS option of PROP displays the current state of an active printer environment only, such as whether it is idle, printing, and so on. For a list of other possible printer states, refer to Chapter 4, Controlling Printer Formatting.

Any user can check the status of the printers by entering:

```
PROP -STATUS [-ALL    ]
             [-NO_WAIT]
```

PROP –STATUS alone displays only the active printer environments, which are those that have been started. The –ALL option displays the status of all printer environments, including those that are not started or in some other state.

The following example shows a sample PROP –STATUS display:

```
OK, PROP -STATUS
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
SPWT            Idle
WIDE            Idle
INFO            Idle
OPS             Idle
QMST2          Printing (HELPLIST.233: Page 2, Copy 1 of 1
                          Request: 613)
TP_QUM          Idle
TRAIN           Idle
PLANS           Idle
OK,
```

In the example above, the environment called QMST2 is printing a file. All other printer environments are idle and available for use.

The example below shows the status display using the –ALL option:

```
OK, PROP -STATUS -ALL
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
SPWT            Idle
WIDE            Idle
CLAM            Not Started
MEDIA           Not Started
INFO            Idle
RASLW           Not Started
PDE2            Not Started
OPS             Idle
QMST2           Idle
TP_QUM          Idle
LABEL           Not Started
OK,
```

The previous display shows existing printer environments that are not started. If you have created several environments for a single printer, and you wish to use a different environment, you must first stop the current environment before starting an alternate one. You do not have to stop the current environment for printers connected to TCP/IP, which accepts requests from more than one environment. If a stopped environment is associated with a particular printer, you can easily start it up again. (See Starting a Printer Environment later in this chapter.)

Stopping a Printer Environment

If you wish to start a new environment for a particular printer, or make any physical adjustments to the printer itself, you need to stop the despooler phantom that controls the current printer environment so that it does not continue processing requests from the queue..

To stop a printer environment, issue the following command:

PROP ***environment-name*** **–STOP [*options*]**

Depending on the status of the printer environment, you may need to include one of the following three options:

- –FINISH allows a currently printing file to finish printing before stopping the environment.
- –NOW stops the despooler phantom immediately.
- –IDLE stops the despooler phantom when there is nothing left in the queue to be printed by that despooler phantom.

If you do not include one of these options, the despooler phantom stops after it finishes printing the current file.

In the following example, PROP –STOP is used to stop the environment TP_QUM:

```
OK, PROP TP_QUM -STOP -FINISH
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
OK,
*** TP_QUM (user 266 on ENB) at 13:21
Despooler for TP_QUM stopping.
```

When you issue PROP –STOP, the despooler phantom finishes printing whatever remains in the print buffers, which means there may be some delay between issuing PROP –STOP and the display of this message.

In addition to the message displayed at the terminal, PROP prints a message at the printer itself, indicating that the printer is stopping (if the environment file has WARNING set to ON).

Starting a Printer Environment

Start a printer environment by using the following command:

PROP *environment-name* –START

When you start a printer environment, you start a despooler phantom process which controls the environment and the printer itself. If a printer is already running an environment, and you wish to start a new environment, you must first stop the current environment before you start a new one.

The following example shows the use of PROP to an environment called TP_QUM:

```
OK, PROP TP_QUM -START
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
OK,
*** TP_QUM (user 266 on ENB) at 13:21
Despooler for TP_QUM ready.
```

Note If you are starting a PostScript printer environment, the despooler phantom loads the dictionaries first, which means the despooler ready message may not display for several minutes.

At Rev. 23.3, the Spool Administrator can give RX access rights to specific users allowing them to abort, back up, continue, drop, hang, line up, release, reset, restart, stop, start, and suspend printer environments.

Unless the Spool Administrator gives access rights to specific users, nonprivileged users can use only the PROP –DISPLAY, –STATUS, –VERIFY, –REPORT_SIZE, and –HELP commands.

Overview of Spooler Subsystem Tasks

This section lists the tasks you perform to set up the Spooler subsystem. The instructions and procedures for performing the following tasks are given in Chapter 2, Setting Up the Spooler Subsystem.

Setting Up Printers

- Determine the number of printers and their printing capabilities.
- Determine the printing needs of users.
- Create one or more printer environments for each printer.

- Create attributes files and other directories and files for optional features that you plan to use.
- Develop schedules and procedures for running printer environments.
- Decide the locations and procedures for distributing printed output.
- Decide the level of information you want printed on header and trailer pages, such as pathnames and user IDs, and modify the default settings in the environment file as needed.
- Create custom SPOOL –LIST displays (optional).

Depending on the size of your operation, and the types of printing services you need, you may need to gather additional information before setting up the Spooler subsystem.

For example, if all printing is performed at a central location, you may need to maintain copies of the different types of forms used for printing, schedules for running environments, and instructions for loading and lining up each type of form or paper on the printer, so that an Operator can find them easily.

Operating and Maintaining Printers

The following tasks are generally performed by Operators and other people with responsibility for operating and maintaining printers:

- Stop, start, and restart printer environments.
- Change to alternate printer environments to print special requests.
- Monitor the local and remote spool queues periodically to check for special requests.
- Help users modify spool requests when necessary.
- Print special forms on a regularly scheduled day and time.
- Load and unload paper and special forms according to schedule (and when changing to another environment).
- Change printer ribbons, print wheels, and perform other printer services as scheduled, and according to the manufacturer's warranty and instructions.
- Drop currently printing jobs that were spooled by mistake.

Setting Up the Spooler Subsystem

2

Introduction

This chapter describes how to install, set up, and secure the Spooler subsystem. Before setting up the Spooler subsystem, you should read Chapter 1, which presents an overview of the information and terms described in this chapter. This chapter provides instructions for performing the following tasks:

- Installing the Spooler subsystem
- Preparing the Spooler subsystem for use
- Setting up the Spooler subsystem directories and files
- Cold starting the Spooler subsystem
- Using spool queues
- Securing the Spooler subsystem
- Using pre-Rev. 21.0 Spoolers
- Determining printing requirements
- Using attributes

The Spooler subsystem includes optional features for controlling printers with special formatting, such as PostScript, and contains features that you can tailor to meet your particular requirements. For example, you can replace the standard spool queue display with one that you create. Chapters 4 and 5 contain instructions for setting up and using these and other optional Spooler subsystem features.

Installing the Spooler Subsystem

The top-level directories, SPOOL*, SPOOL_QUEUE*, and SPOOL_DATA* should exist on your system. If they do not exist on your system, you will need to create them and follow the installation and setup procedure in the next section. (Refer also to the online INFO file called SPOOL.RUNO located in SPOOL*>INFO for up-to-date installation information.)

If you need to install the Spooler subsystem, for example to install a rapid-repair release, run the SPOOL.INSTALL.CPL program located in the top-level directory SPOOL supplied with the release. The SPOOL.INSTALL.CPL program copies the Spooler subsystem components to the appropriate directories. The top-level directory SPOOL is supplied only with rapid-repair releases. It is not supplied as part of the Spooler subsystem.

Protecting Spooler Subsystem Files and Directories

To protect your system from unauthorized use, you must set the correct ACLs (Access Control Lists) on the Spooler subsystem directories and files by running the following utility that Prime supplies:

```
OK, RUN SYSTEM>SPOOL.INSTALL_ACL.CPL
```

This utility sets the correct ACLs automatically. For more information, refer to Securing the Spooler Subsystem later in this chapter.

Preparing the Spooler Subsystem for Use

This section gives an overview of the steps you perform to install and set up both the required and optional features of the Spooler subsystem. Each step in the general procedure below refers you to the chapter that contains complete instructions for performing that task.

1. Read the INFO file for up-to-date installation instructions and for information about changes to the Spooler subsystem. This file is called SPOOL.RUNO and resides in the directory SPOOL*>INFO.
2. Create environment files for printers and plotters. Environment files define the type of printer and its setup. Despooler phantoms use environment files to control printers and direct spool requests to the correct printers. Printer environments are discussed in this chapter and Chapter 3, Defining Printer Environments.
3. Optionally, create attributes files to verify SPOOL command attributes and provide default attributes for all users, specific users, and projects. Refer to Using Attributes later in this chapter for more information.

4. Create additional SPOOL_DATA* directories, if you need them, and create the DATA_PARTITIONS file in the SPOOL_QUEUE* directory. Refer to Creating Additional Directories for Spooled Files later in this chapter for more information.
5. Review whether any users need the ability to view all spool queue entries, and create a FULL_LIST_USERS file in the SPOOL_QUEUE* directory.
6. Assign privileged users by adding their user IDs to the ACL group .SPOOL_ADMINISTRATOR$ and to the file called SPOOL*>RESTRICTED_USERS>SWO_VALID_USERS. Refer to SPOOL* Subdirectories later in this chapter for more information.
7. Determine if any users need to control certain printer environments and add their user IDs with RX access to the Access Control List of those environment files. See the PROP Command section in Appendix A.
8. Create Electronic Vertical Format Unit (EVFU) files for printers that require special formatting. Chapter 4 explains how to set up and use EVFU files.
9. Create a cold-start procedure for the Spooler subsystem within the PRIMOS.COMI file that automatically starts the appropriate despooler phantoms when the system itself is cold started.
10. Optionally, create a file of user address information for distribution purposes. Refer to Chapter 5 for more information.
11. Optionally, create custom SPOOL –LIST displays. See Chapter 5 for more information.

Setting Up the Spooler Directories and Files

This section describes the Spooler subsystem directories and files and provides information on the ACL settings that must be established on these directories and files to ensure the security of spooled data. If you are using or upgrading from a pre-Rev. 21.0 Spooler, read the section called Using Pre-Rev. 21.0 Spoolers later in this chapter.

Spooler Subsystem Directory Structure

Figure 2-1 shows the organization of the Spooler subsystem directories and files. Their contents and the file system protection (ACLs) that should be set for each directory are described.

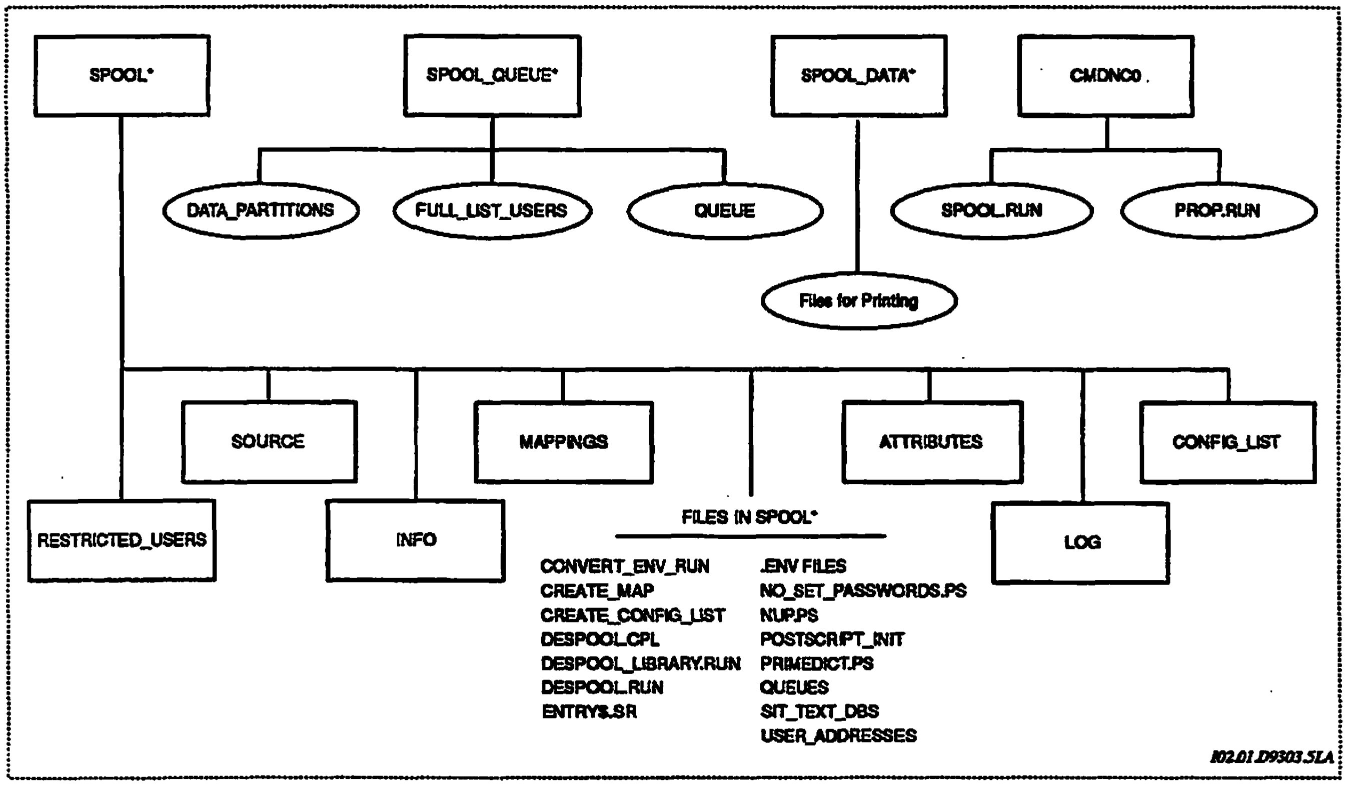

Figure 2-1. Directories and Files Used by the Spooler Subsystem

The following directories reside at the top of the directory structure, as shown in Figure 2-1:

- SPOOL* contains the files and subdirectories that the Spooler subsystem uses to control printers and their environments, spool queues, log file entries, and so on.

- SPOOL_QUEUE* contains the QUEUE file, which is a list of print requests and contains the two optional files DATA_PARTITIONS and FULL_LIST_USERS.

- SPOOL_DATA* is the directory to which the SPOOL command copies files for printing. (Files spooled with the –NO_COPY or the –SPOOL_WHILE_OPEN option are not copied to this directory. See Appendix A for information on the SPOOL command options.)

- CMDNC0 is the system top-level command directory containing the system's program files and containing the runfiles for the SPOOL and PROP commands.

SPOOL* Directory

The SPOOL* directory contains the following files and subdirectories. You must protect this directory by giving LUR access to nonprivileged users and ALL access to members of the .SPOOL_ADMINISTRATOR$ group, System Administrators, and other privileged users.

Set the Access Control List (ACL) as follows:

```
.SPOOL_ADMINISTRATOR$:ALL
$REST:LUR
```

The SPOOL* directory contains the following files:

DESPOOL.CPL

The PROP command uses this file to activate the despooler program. You can modify this file to include ADD_REMOTE_ID (ARID) commands for network access. You should prevent users from seeing the ARID passwords by setting an ACL that permits only the .SPOOL_ADMINISTRATOR$ group to access this file.

DESPOOL.RUN

DESPOOL.RUN is the runfile for the despooler phantoms. This runfile is invoked each time a despooler phantom is started.

DESPOOL_LIBRARY.RUN

This runfile contains the library of print handler routines supplied by Prime.

ENTRY$.SR

The ENTRY$.SR file, which Prime supplies, is required by the Spooler subsystem. This file sets the ENTRY$ search rules for a despooler phantom and includes the following lines:

```
SPOOL*>DESPOOL_LIBRARY.RUN
-SYSTEM
```

If you modify this file, you must follow the instructions in Appendix B, Writing a Print Handler, and Chapter 3, Defining Printer Environments, specifically, the TCP environment directive. If you supply your own accounting routine, banner page routine, or print handlers, you must add the location of the EPF library entrypoints to the search rules in this file.

POSTSCRIPT_INIT

This file is the default file used when you initialize a PostScript printer. Use this file to enter the pathnames of files containing PostScript procedures.

QUEUES

This is an optional file that you create for entering queue search directives that are common to multiple environment files designed for the same printer. It is similar in structure to an environment file and can contain the following directives:

DISK
DISK_SCAN
NODE
POLL_RATE
PRIORITY_TO_LOCAL_QUEUE

The QUEUES file has the following two uses:

- The SPOOL –LIST –ALL command reports on all the queues defined in this file. If there is no SPOOL*>QUEUES file, then SPOOL –LIST –ALL reports on the local queue only.
- The despooler phantom uses the information in SPOOL*>QUEUES if you have included the CONFIG QUEUES directive in the environment file. Using the CONFIG QUEUES directive enables easier queue management and avoids duplicating information in each environment file.

USER_ADDRESSES

This is an optional file that you create to enter address information for each user ID. User address information is printed on the header page of each printed document for distribution purposes.

environment-name.ENV

The SPOOL* directory also contains environment files that define the operation and printing capabilities of different types of printers.

environment-name is the name of the printer environment. Environment filenames must have the extension .ENV, and must follow the same rules for defining a PRIMOS user ID. That is, the environment filename must be 1 through 32 characters long; begin with a letter; and contain only letters, digits, periods (.), underscores (_), and dollar signs ($). Printer environments are explained in the next section of this chapter, and in Chapter 3, Defining Printer Environments.

When you start a despooler phantom for the first time, it creates a file with the pathname SPOOL*>*environment-name*.SEM. The SPOOL command uses this file to start the despooler phantom for the named environment when requests for that environment exist in the spool queue for processing. You must not alter files with the .SEM extension because they have special meaning to the operating system for handling inactive processes.

Any environment files that you create must also reside in the SPOOL* directory. Refer to Chapter 3 for more information on creating and naming environment files.

SPOOL* Subdirectories

The following subdirectories reside below the SPOOL* directory:

MAPPINGS

This subdirectory contains the character mapping files ECS_TO_ASCII_MAP and QUME_OAS_MAP. The first file maps ECS (Extended Character Set) to ASCII and the second file is a map for QUME printers that require OAS support and ECS to ASCII. This subdirectory can also contain the STANDARD_MAP file, although this file is not placed there by the PRIMOS default build.

RESTRICTED_USERS

This subdirectory contains the SWO_VALID_USERS file that lists authorized SPOOL_WHILE_OPEN users. Each user is entered on a separate line which can be a user ID, an ACL group, or the .ALL_USERS. category.

ATTRIBUTES

This subdirectory contains attributes files that you create to validate SPOOL command attributes and supply default attributes for all users, specific users, and project users. The attributes listed in the attributes files are the same attributes that users issue with the SPOOL command to select a printer and its capabilities, which are also defined in printer environment files.

SOURCE

This subdirectory contains source code and insert files that you can use to create a custom spool queue list routine, custom print handlers, and other routines.

LOG

This subdirectory contains log files that record despooler activities. For more information on the LOG environment directive, refer to Chapter 3, Defining Printer Environments.

CONFIG_LIST

This subdirectory contains binary files that produce the default SPOOL –LIST and custom SPOOL –LIST displays that have been created for individual users and project users.

INFO

This subdirectory contains information files, such as INFO.RUNI, which contains the most up-to-date information on changes, enhancements, and installation instructions for the latest release of the Spooler subsystem.

SPOOL_QUEUE* Directory

The SPOOL_QUEUE* directory contains a file called QUEUES, which is not an ASCII file. This file contains the list of waiting print requests.

Set the ACL for the SPOOL_QUEUE* directory as follows:

```
.SPOOL$$:ALL
$REST:NONE
```

The SPOOL_QUEUE* directory can contain two optional files:

FULL_LIST_USERS

This file contains the list of users who can view the entire spool queue with the SPOOL –LIST command.

DATA_PARTITIONS

This file lists the names of disk partitions on the local machine that contain a SPOOL_DATA* directory. Refer to the section Creating Additional Directories for Spooled Files later in this chapter.

SPOOL_DATA* Directory

The SPOOL_DATA* directory contains copies of spooled files. Files spooled with the –NO_COPY or –SPOOL_WHILE_OPEN option are not copied to this directory. –NO_COPY is useful when spooling very large files that would occupy excessive disk space in SPOOL_DATA*. –SPOOL_WHILE_OPEN allows privileged users to spool a file while it is open for writing to disk.

Set the ACL for the SPOOL_DATA* directory as follows:

```
.SPOOL$$:ALL
$REST:NONE
```

Cold Starting the Spooler Subsystem

The Spooler subsystem is cold started as part of the system cold-start routine, which must occur after setting the system date and time.

To cold start the Spooler subsystem, follow these steps:

1. Share the Spooler subsystem data area.
2. Issue the PROP –COLDSTART command.
3. Start up despooler phantoms (optional).

The first two steps are included in the PRIMOS.COMI file (a command input file supplied by Prime for starting the system). You must perform these two steps whether or not you start any despooler phantoms. The PRIMOS.COMI file includes the following command line that shares the Spooler subsystem data and performs a cold start of the Spooler subsystem:

```
COMI SYSTEM>SPOOL.SHARE.COMI
```

The SPOOL.SHARE.COMI file contains the PROP –COLDSTART command that initializes all the Spooler subsystem data structures and resets the queue control information.

If you must reinitialize the Spooler subsystem for some reason, the Spool Administrator and members of the .SPOOL_ADMINISTRATOR$ ACL group may issue the PROP –COLDSTART command at any time. For example, you may need to reinitialize the Spooler subsystem if you have removed a SPOOL_DATA* directory.

Depending on the status of the Spooler subsystem at the time you issue the PROP –COLDSTART command, the following conditions may result.

Spooler Subsystem Status	*Results of PROP –COLDSTART Command*
No active despoolers and queues empty	Shared segment and request numbers are reinitialized.
Queues contain entries	Printing flags are reset, but the count is not reset. Shared segment is reinitialized.
Active despoolers	A dialog begins for shutting down the individual active despoolers. Answer YES to shut down each active despooler. The two results described above occur after shutting down the active despoolers.
Active despoolers and active environments	Entries associated with active environments are left in the shared segment. Entries associated with environments to be shut down (if any) are removed from the shared segment. The spool queue is not reinitialized. PROP –COLDSTART automatically removes any entries listed in the shared segment for despoolers that have logged out.

Starting Despooler Phantoms at System Cold Start

You can start despooler phantoms as part of the system cold-start procedure by entering the PROP commands for the environments you wish to start in the SYSTEM>SPOOL.SHARE.COMI file. For example:

```
PROP DOC -START
PROP LQP -START
```

The SYSTEM>SPOOL.SHARE.COMI file contains a place holder for inserting the PROP commands.

For instructions on how to start, stop, and suspend printer environments at any other time, see Chapter 6, Managing Spooler Operations.

Using Spool Queues

This section explains briefly how despooler phantoms search the spool queues for spool requests, and how you can define the queue search settings as they apply to your system requirements.

Spool queue searching is defined by including queue search directives in the environment files controlled by each despooler phantom. For information on using queue search directives and creating printer environment files, refer to Chapter 3, Defining Printer Environments.

Each system at Rev. 21.0 and later contains a single spool queue in the SPOOL_QUEUE* directory. The SPOOL command adds spool requests automatically to the local spool queue.

The –ON option of the SPOOL command allows you to place requests in the spool queue of other systems in a network. To place requests in the spool queue of pre-Rev. 21.0 systems in a network, use the –DISK option of the SPOOL command. (Refer also to Using Pre-Rev. 21.0 Spoolers later in this chapter for more information on pre-Rev. 21.0 Spoolers.)

Searching Spool Queues on Other Systems

You can instruct the despooler phantoms to search the spool queue of other systems in the network, in addition to searching the local spool queue.

Refer to the descriptions of CONFIG, DISK, and NODE in Chapter 3, Defining Printer Environments, for more information.

Rev. 21.0 and later Spoolers can search remote queues of both Rev. 21.0 and pre-Rev. 21.0 formats.

Searching Spool Queues in a Ring Network

If you have a ring network, and you want the despooler phantoms to process all requests for the environments that they control, no matter which node they are queued on, you need to include the NODE –RING directive in the environment file. The NODE –RING directive instructs the despooler to search spool queues at all the nodes defined in the ring configuration.

Defining Queue Search Frequency

Using queue search directives, you can define how frequently the despooler phantoms check each spool queue for spool requests. For information on setting these parameters, refer to the descriptions of POLL_RATE, NODE, and DISK in Chapter 3, Defining Printer Environments.

Using the Default Queue Search Order

When searching remote spool queues, a despooler phantom processes all requests for its associated environment before it searches the next spool queue. If the remote spool queue contains numerous requests for that particular environment, requests in the local spool queue wait until the despooler phantom is finished. This default queue search order can delay the processing of requests from the local spool queue.

Change the default queue search setting by including the directive PRIORITY_TO_LOCAL_QUEUE in the environment file. This directive gives priority to the local spool queue. If the despooler phantom is processing a request from a remote queue, this directive forces the despooler phantom to search the local spool queue for requests immediately after it finishes processing the request from the remote spool queue.

Securing the Spooler Subsystem

The Spooler subsystem is designed to meet the U.S. Department of Defense C2-certified security.

The SPOOL_QUEUE* and SPOOL_DATA* directories are protected by ACLs (Access Control Lists). You must not use passwords to protect these directories. Set the ACL for the SPOOL_QUEUE* and SPOOL_DATA* directories as follows:

```
.SPOOL$$:ALL
$REST:NONE
```

The SPOOL_QUEUE* and SPOOL_DATA* directories are manipulated only by the queue management routines. When a user issues a SPOOL command, the .SPOOL$$ group is temporarily added to the user ID only until the data is transferred to the protected directories. Users cannot see or modify files that belong to another user.

Caution To ensure the security of the SPOOL_QUEUE* and SPOOL_DATA* directories, do not add any users to the .SPOOL$$ ACL group. In order to determine which users are in a particular ACL group, you must check the profile of each user.

Securing Spooled Data

Set the ACL for the SPOOL* directory as follows:

```
.SPOOL_ADMINISTRATOR$:ALL
$REST:LUR
```

Spooled files are copied to the SPOOL_DATA* directory. The –NO_COPY option of the SPOOL command prints a file from its current file system location; the file is not copied to the SPOOL_DATA* directory. Users must protect files that reside in their own directories.

In order to use the –NO_COPY option of the SPOOL command, ensure that the ACLs are set correctly on the directory containing the file you are spooling, and on the file itself as follows:

1. Set the ACL on the directory containing the spooled file to

   ```
   $REST:LUR
   ```

2. Set the ACL on the spooled file to

   ```
   $REST:R
   ```

Nonprivileged users can spool files to which they have access, view their own spool requests, and cancel or modify their own requests. Nonprivileged users can also use PROP –STATUS to check which environments are active. Nonprivileged users cannot view other users' entries in the spool queues unless the Spool Administrator has created a FULL_LIST_USERS file in the SPOOL_QUEUE* directory containing the names of users who can view all entries in the spool queue.

Only privileged users can use the PROP command options that control printer environments, unless the Spool Administrator gives specific users access rights to the environment file. Only members of the .SPOOL_ADMINISTRATOR$ group are privileged users.

Setting Spooler Subsystem ACLs

Prime supplies a utility that automatically sets the correct ACLs for the Spooler subsystem directories. The SPOOL.INSTALL_ACL.CPL utility is located in the top-level directory SYSTEM. When you run this utility, it sets the ACLs correctly on all the directories and files on the local disk that are used by the Spooler subsystem. You should use this utility to set ACLs automatically, unless you require different ACLs.

If you have created several SPOOL_DATA* directories, you must first create a DATA_PARTITIONS file containing the names of the partitions containing SPOOL_DATA* directories, in the SPOOL_QUEUE* directory, before you run the utility. The utility sets ACLs only for SPOOL_DATA* directories on partitions listed in the DATA_PARTITIONS file. If there is no DATA_PARTITIONS file, the utility assumes that there is only one SPOOL_DATA* directory, and sets the ACL on the first SPOOL_DATA* directory that it finds.

Run the utility program for setting ACLs automatically by issuing the following command:

```
OK, RUN SYSTEM>SPOOL.INSTALL_ACL.CPL
```

This utility does not set ACLs on remote disks. You must set ACLs on remote disks by using the PRIMOS commands SAC and EDAC. You may use these PRIMOS commands to set nonstandard ACLs on any file or directory in the Spooler subsystem, but the security of the subsystem cannot be guaranteed if the recommended ACLs are not used.

Using Pre-Rev. 21.0 Spoolers

The Rev. 21.0 and later Spoolers contain numerous changes and enhancements. If you are upgrading from a pre-Rev. 21.0 Spooler subsystem, you need to set up the SPOOL*, SPOOL_QUEUE*, and SPOOL_DATA* directories as described in this chapter. You must also redefine any pre-Rev. 21.0 environment files and convert them to a new format using a conversion utility provided by Prime and described in the next section.

Running Different Spooler Versions in a Network

If you have a network of systems running different versions of the Spooler, including pre-Rev. 21.0 Spoolers, be aware of the following changes:

- At Rev. 21.0 and later, the Spooler can add spool requests to spool queues on pre-Rev. 21.0 systems and spool requests from pre-Rev. 21.0 nodes.

- Spoolers on pre-Rev. 21.0 systems cannot access the spool queues of Rev. 21.0 and later systems.

Users can spool files from a Rev. 21.0 or later system to a printer on any system in the network. If a system in the network is running a pre-Rev. 21.0 Spooler, users may need to use the –DISK option to place the request in the correct spool queue.

Any changes that affect the use of pre-Rev. 21.0 Spooler functions are documented in the sections that describe those functions.

Converting Pre-Rev. 21.0 Environment Files

This subsection describes the utility that produces a Rev. 21.0 environment file template from an existing pre-Rev. 21.0 environment file. At Rev. 21.0 you may need several slightly different versions of each environment. Use the utility below to produce a template, then modify the template to create each of the environments you need.

To create a Rev. 21.0 environment file template, issue the following command:

```
                     [old_env new_env                                   ]
                     [{-ENVIRONMENT old_name}                           ]
R SPOOL*>CONVERT_ENV [{-ENV old_name        } new_name                  ]
                     [-HELP                                             ]
```

old_env is the pathname of a pre-Rev. 21.0 environment file.

new_env is the pathname of the file in which the Rev. 21.0 environment file is to be stored.

old_name is an environment filename without either the E. prefix or the .ENV suffix. Use the optional *new_name* if you want to rename the environment you are converting.

If you use the –ENVIRONMENT option, the utility looks for the file SPOOLQ>E.*name* and produces the file SPOOL*>*name*.ENV. If you supply pathnames, the old file and the new file are saved to those locations in the file system.

Viewing the Spool Queues

Nonprivileged users can display only their own entries in the spool queue with the SPOOL –LIST command. Privileged users, SYSTEM, and members of .SPOOL_ADMINISTRATOR$ can see the entire queue.

To enable all users to see the whole spool queue, create a file called FULL_LIST_USERS in the SPOOL_QUEUE* directory containing the following entry:

```
.ALL_USERS.
```

The periods at the beginning and end of the .ALL_USERS. keyword are part of the name. To allow only selected users to view the entire spool queue, enter each user ID on a separate line in the FULL_LIST_USERS file or a single ACL group name.

To save a FULL_LIST_USERS file, you need to change the ACL for SPOOL_QUEUE* temporarily to allow you to write to that directory.

Creating Additional Directories for Spooled Files

As explained in Chapter 1, the SPOOL command copies spooled files to the SPOOL_DATA* directory (unless you use the –NO_COPY or –SWO option of the SPOOL command).

One SPOOL_DATA* directory is usually sufficient for most systems, but if your printing volume requires more space, you can create added SPOOL_DATA* directories.

To create another SPOOL_DATA* directory, do the following:

1. Create the SPOOL_DATA* directories you need, each on a different partition on the local system. Name the directory SPOOL_DATA*.

   Do not create a SPOOL_DATA* directory on the same partition containing the SPOOL_QUEUE* directory. If SPOOL_DATA* resides on the same partition as the SPOOL_QUEUE* directory, and the files copied to SPOOL_DATA* fill the disk, the spool queue itself cannot accept any new spool requests.

2. Set the ACL for each SPOOL_DATA* directory as shown below:

   ```
   .SPOOL$$:ALL
   $REST:NONE
   ```

   Do not use passwords to protect these directories. Use the ACL settings shown above to secure the data.

3. Create a file named DATA_PARTITIONS in the SPOOL_QUEUE* directory.

4. Enter the names of the partitions on which you have created a SPOOL_DATA* directory. Enter only the name of the partition on a separate line, without the left and right angle brackets (< and >).
5. Save the DATA_PARTITIONS file.

If you do not create a DATA_PARTITIONS file, the SPOOL command locates and uses the first SPOOL_DATA* directory on the system. You can have a single SPOOL_DATA* directory and operate without a DATA_PARTITIONS file.

Using Multiple SPOOL_DATA* Directories

When you have more than one SPOOL_DATA* directory, the SPOOL command copies spooled files to the first partition name listed in the DATA_PARITIONS file, until the disk becomes full (unless some other problem prevents the SPOOL command from copying files to that partition containing the SPOOL_DATA* directory).

If the disk containing the SPOOL_DATA* directory becomes full, the SPOOL command removes any partially written file, and uses the next partition name listed in the DATA_PARTITIONS file. The SPOOL command copies files to the current partition until the disk is full, and then uses the next partition name in the list, and so on. The SPOOL command also uses the next partition in the list for SPOOL -OPEN requests.

If the SPOOL command is unable to copy files to the last partition name listed in the DATA_PARTITIONS file, it tries the first partition name in the list. If all the partitions containing SPOOL_DATA* directories are full, the SPOOL command reports an error.

Deleting a SPOOL_DATA* Directory

You can add new SPOOL_DATA* partition names to the DATA_PARTITIONS file at any time. However, if you want to delete a SPOOL_DATA* directory and remove the partition name from the DATA_PARTITIONS file, you must follow the procedure below:

1. Delete the partition name from the DATA_PARTITIONS file.
2. Stop all the despooler phantoms.
3. Issue the PROP -COLDSTART command. This step ensures that the SPOOL command uses the revised DATA_PARTITIONS file.
4. Start up the despooler phantoms again.
5. Wait until all the files in the SPOOL_DATA* directory that you want to remove have been printed. Use SPOOL -LIST -FULL to check. The full display includes the partition name of the SPOOL_DATA* directory.
6. Delete the SPOOL_DATA* directory.

Determining Printing Requirements

This section describes information you need to set up printer environments for each printer.

You create environment files that define the printing capabilities of each printer and that the despooler phantoms use to process requests spooled to a particular printer. Before creating printer environment files, you must first determine the printing needs of users and determine how to service various types of requests using the printers available on your system.

If you have only one printer or a limited number of printers, you can create multiple environments for a single printer, each of which performs a different printing task. Although you can have more than one environment for a printer, the printer can service only one environment at a time. Only TCP/IP-connected printers can process requests from more than one environment at any one time.

Using Multiple Printer Environments

You may wish to set up extra environments to

- Change the paper type, forms, or printing formats.

  Anytime you change the paper type, you must first stop the current environment and start a new environment. You might set up the following environment names to handle different printing requests on one line printer, for example:

  > WIDE for printing on standard wide listing paper and a standard print format
  >
  > WHITE for printing on narrow white paper
  >
  > SHIP to print shipping labels on special forms

- Switch fonts.

  You can create separate environments for printing different fonts or to map the characters displayed at your terminal to print a different set of characters. (Certain types of printers, such as PostScript, can support multiple fonts and other format settings simultaneously within one environment file.)

- Limit the size of files printed during peak working hours.

  You may wish to set up environments that allow only small to medium size files to print during working hours, and then switch to another environment for printing large files during the evening hours.

- Direct requests according to the speed of the printer.

If you have two printers that operate at different speeds, you could create an environment that directs all large files to a high-speed printer and one that sends small files to a slower printer.

- Establish default settings.

  Each printer environment should contain default settings for paper orientation (landscape or portrait), paper bin, font type, and other settings that are necessary for most print requests.

Contents of an Environment File

Environment files contain environment directives that implement printing capabilities and control certain hardware features of each printer. Environment directives are divided into these five categories:

- **Interface definition directives** define the data line hardware connection to the system and protocol used to transmit data to the printer.
- **Scheduling requirement directives** match spool requests with an appropriate environment and determine the order in which files are printed. Attributes are one of the types of scheduling requirement directives. Refer to the next section, Using Attributes, for information on how requests are matched with environments.
- **Queue search directives** define the spool queue locations searched by each despooler phantom and how frequently the despooler phantom searches each spool queue.
- **Formatting option directives** determine the format of the printed output. Print handler routines are one type of formatting option directive. Refer to Appendix B for information on print handler routines.
- **Logging despooler directives** define the level of despooler activity reported in each environment log file. You can choose to keep an historic log of despooler entries (processed spool requests) or record only the most recent environment action.

Refer to Chapter 3, Defining Printer Environments, for a full description of the environment file directives and their options.

Using Attributes

Attributes are one of the scheduling requirement directives that you use to define the operation and printing capabilities of a particular type of printer. The names that you choose for attributes generally describe some feature of the printer itself, a printer location, and identifiers for the paper type and other printing options.

The attribute names you choose to describe printers and their features are the same names that users give with the SPOOL –ATTRIBUTE command to select a printer and available printing options.

Example of Using Attributes

The following example shows how attributes are used to match spool requests with the appropriate printer.

Suppose you have two line printers named PR0 and PR1. Printer PR0 has an attribute called WIDE, indicating wide paper, and the printer called PR1 has an attribute called LETTER, identifying letter-size white paper.

To print a file on WIDE paper, issue the following SPOOL command:

SPOOL *filename* –ATTRIBUTE WIDE

The SPOOL command checks the environment files for an attribute called WIDE and notifies the despooler phantom that controls a printer containing the attribute WIDE in its environment file, which in this case is the printer called PR0. (The SPOOL command checks environment file attributes only if you have created special files called attributes files, as explained in the following section.)

The despooler phantom controlling printer PR0 also checks the SPOOL request attributes against those in its associated environment file. If the attributes match, the despooler phantom processes the file. Using the same example, if you want to print a file on letter-size paper, you use the attribute name LETTER to send your spool request to the printer with letter-size paper.

Note Before Rev. 21.0, the SPOOL command options –FORM, –AT, and –TYPE were used to select a printer based on form type and printer location. At Rev. 21.0 and later, you can use the –ATTRIBUTE option to select these features and other available printing options.

Creating Attributes Files

Attributes files are optional files that you can create to verify SPOOL command attributes and supply default attributes for all users, specific users, and projects. The SPOOL command will verify attributes given with spool requests only if you create these files. You can set up attributes files to

- List attributes that you want the SPOOL command to verify.
- List groups of related attribute names, for example, system names, location names, paper types, and so on. Users are allowed to choose only one attribute from each group.

- Create default attributes that the SPOOL command supplies automatically if attributes are not given with the SPOOL command. You must create at least one attributes file called .DEFAULT_ATTRIBUTES. if you want the SPOOL command to provide defaults automatically.
- Create a list of valid attribute names for different groups of users that share one or more printers.

With attributes files, spool requests must contain one or more of the attributes listed in the attributes file (or no attributes if defaults are listed), or the request is not entered into the spool queue, and you cannot give more than one attribute from each group.

Without attributes files, the SPOOL command cannot verify attributes given with spool requests or provide default attributes if none are given on the command line. If users do not provide the attributes needed for a particular printer environment, the spool request is entered into the queue, but may not print if the request does not contain attributes that match an available printer environment.

Naming Attributes Files

To create attributes files, you must first create a subdirectory called ATTRIBUTES in the SPOOL* directory. Within the ATTRIBUTES subdirectory, you can create any number of attributes files. Guidelines for naming attributes files are listed below:

Type of Attributes File	*Filename Convention*	*Example*
DEFAULT	.DEFAULT_ATTRIBUTES.	.DEFAULT_ATTRIBUTES.
User	User ID	LAURA
Project	*project_name	*VOYAGERII

Contents of Attributes Files

An attributes file contains a list of attribute names or groups of related attribute names. Each group of related attribute names must be separated by at least one blank line or a comment line. The first name in each group is the **default** attribute that is supplied. If no default is required, the name .NO_DEFAULT. must be the first name listed in the group. The periods before and after .NO_DEFAULT. are part of the attribute name.

If an attributes file contains groups of related attributes, for example, a group of system names, you can include only one attribute from each group in a spool request. If you include more than one attribute from the same group in your spool request, the message `incompatible attributes` is displayed.

Attributes files can contain comment lines for describing individual attributes or groups of attributes. A comment line begins with a slash, followed by an asterisk (/*) and can occupy a line by itself, which indicates the start of a new group of attributes, or can be placed next to an attribute name to describe that particular attribute.

Defining Mandatory Attributes

You can define mandatory attributes for features or options that are required for a printer environment using the ATTRIBUTE –MANDATORY directive. If you define a mandatory attribute, it means that users *must* give that attribute in their spool request, or the file will not print (unless another available printer environment has the same combination of attributes as those contained in the request).

For example, if you have set up a printer specifically for printing labels, you might define a mandatory attribute called LABELS in the printer environment file, or a name that you choose to indicate label printing only. In this example, the mandatory attribute LABELS would prevent users from spooling other types of files to a printer that has been set up for a special purpose. The entry in the environment file would be as follows:

```
Attribute LABELS -Mandatory  /* LABELS is required
```

Example of Using Attributes Files in a Network

Figure 2-2 shows a sample network of three systems at two different sites that share five printers.

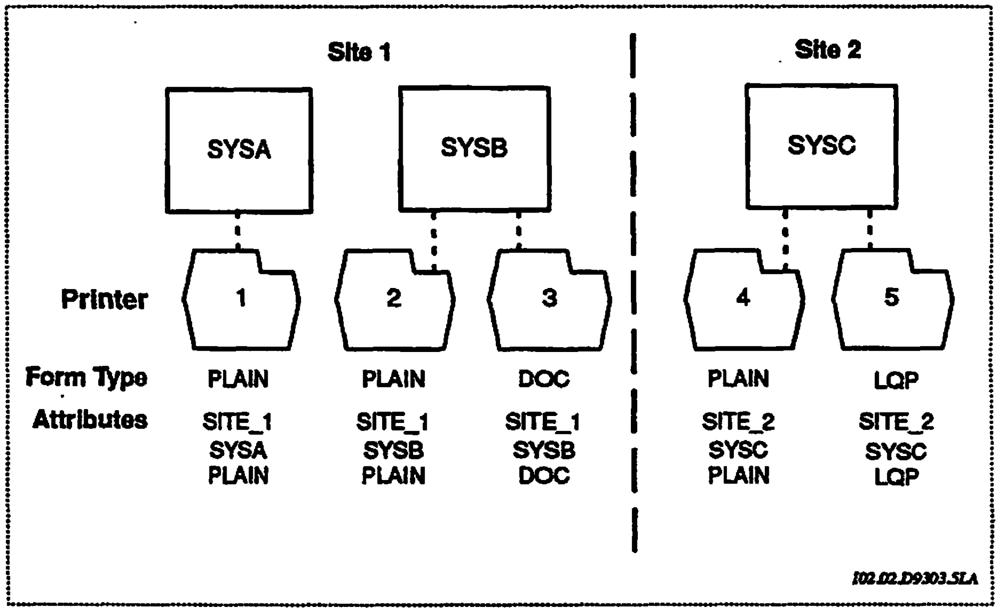

Figure 2-2. Example of Attributes for Printers in a Network

The attributes files for both SYSA and SYSB contain the following entries:

```
/* Site names
SITE_1
SITE_2
/* Form types
PLAIN
DOC
LQP
/* System names
.NO_DEFAULT.
SYSA
SYSB
SYSC
```

Notice that this attributes file contains three groups of attributes: site names, form types, and system names. The default attributes for SYSA and SYSB are SITE_1 and PLAIN.

The attributes file for SYSC contains the following entries:

```
/* Site names
SITE_2
SITE_1
/* Form types
PLAIN
DOC
LQP
/* System names
.NO_DEFAULT.
SYSA
SYSB
SYSC
```

The default attributes for SYSC are SITE_2 and PLAIN.

The following example shows how the attributes in the sample network shown in Figure 2-2 are used to direct spool requests to the appropriate printer. For the purpose of this example, assume that there are five active despooler phantoms, one for each printer, and each phantom searches all of the spool queues.

Example 1: A user on SYSA issues the command

SPOOL *filename*

Since the default attributes are SITE_1 and PLAIN, the document may be printed by either printer 1 or printer 2, depending on which despooler phantom searches the queue first.

Example 2: A user on SYSC issues the command

SPOOL ***filename*** **–ATTRIBUTE DOCS**

Notice that none of the printers at SITE_2 have the form type DOC as an attribute. Although SITE_2 is supplied as a default attribute, the despooler phantoms do not recognize the attribute combination SITE_2 and DOC, so the request remains in the spool queue indefinitely (or until you cancel or modify the request).

If the user in this example entered

SPOOL ***filename*** **–ATTRIBUTE DOC SITE_1**

the file would have been printed by Printer 3.

Example 3: A user on SYSB issues the command

SPOOL ***filename*** **–ATTRIBUTE SITE_2 PLAIN**

The file is printed by Printer 4 on SYSC because it has the right combination of attributes.

Verifying SPOOL Command Attributes

If you create attributes files, the SPOOL command verifies attributes given with spool requests. The SPOOL command verifies attributes by checking for a user-specific attributes file first. If a user-specific attributes file does not exist, it looks for an attributes file with a project login name prefaced with an asterisk.

The names of attributes files for projects must begin with an asterisk so that

- The SPOOL command can recognize attributes files for project groups.
- You can distinguish between the different types of attributes files.

If you do not have user-specific or project-specific attributes files, the SPOOL command looks for the default attributes file that must be named .DEFAULT_ATTRIBUTES..

If no attributes files exist, the SPOOL command enters all spool requests into the spool queue, no matter what attribute names are given with the SPOOL command. If you enter attribute names that are not defined within an environment file, the spool request is entered into the queue but it is not printed.

Using Attributes to Designate a Printing Location

Attribute names can represent something other than a printer feature or printing option.

For example, suppose you have a single printer that services a five-story building, and you need a system for sharing the printer equally among users on each floor. You can define attributes that describe a floor number, a room number, or some other destination that has meaning to a group of users.

The following five steps describe the general procedure for creating environment files and attributes to print files and distribute them to users in the five-story building example.

1. Create five environment files for the printer. The content of each environment file is identical except that each environment file contains a different attribute name for the floor number.
2. Establish default attributes by adding the following lines to the default attributes file on your system:

```
 /* Floor numbers
.NO_DEFAULT.
FLOOR_1
FLOOR_2
FLOOR_3
FLOOR_4
FLOOR_5
```

3. Inform the users on each floor of the attribute name that they must include with their spool requests, and inform them of when the printer will be available for printing their files.
4. Establish a schedule for running each of the five environments alternately throughout the day (or whatever schedule makes sense for your application).
5. While running a printer environment for a particular floor, you can separate the printouts by user name (or whatever identification is given on the header page). Since printing is done for one floor at a time, you eliminate the additional step of sorting printouts by floor number, and then by user name.

The following example shows how users on different floors would spool and print their files. Suppose a user on the second floor issues the following command:

SPOOL *filename* –ATT FLOOR_2

The request is entered into the spool queue but it is not printed until the Operator runs the printer environment containing the attribute FLOOR_2.

If you need to enforce such a system, you could define the floor number as a *mandatory* attribute in the environment files.

If a spool request does not include the mandatory attribute that designates a floor number, the request will not print. You can also set up user-specific or project-specific attributes to add the floor number attribute automatically. Refer to Chapter 3, Defining Printer Environments, for more information on defining mandatory attributes.

Providing Information to Users

The Spooler subsystem consists of many features that you can tailor to meet your particular printing needs. Not all users will know every possible use of the SPOOL command and its options since much depends on how you choose to use the available features. The Spool Administrator should provide users with the following information to enable them to use the Spooler subsystem and the printers efficiently:

- Names of valid device attributes and any mandatory attributes for the printers they use
- Names of alternate printing devices and their attributes if the primary printer is down for service
- Attributes for special printing options or paper for the printers they use
- Schedules of when certain environments are available for spooling special requests
- Knowledge of how to view their entries in the spool queue and restrictions on viewing the spool queues, if necessary
- Knowledge of how to use the particular SPOOL command options for printing their files, canceling requests, and modifying requests
- Physical location of printers and the procedure for either delivering or picking up printouts

3 Defining Printer Environments

Introduction

This chapter describes how to define environments for printers by creating environment files. Despooler phantoms use environment files to control the hardware features and printing capabilities of different types of printers.

Specifically, this chapter discusses the following topics:

- What is an environment file
- Environment directives summary
- Matching requests with printer environments
- Creating environment files
- Environment directives

What Is an Environment File?

An environment file contains directives that define the operation of different types of printers. Environment file **directives** are instructions to the despooler phantom for controlling the printer hardware, available printing capabilities, and instructions for performing the following tasks:

- Processing spool requests for a particular environment
- Searching local, remote, or all spool queues
- Searching one or more spool queue at set intervals
- Controlling page formatting
- Logging despooler actions in the environment log files

The following example shows a sample environment file for a printer called QUME. The entries beginning with a slash and an asterisk (/*) are comment lines that explain the function of each environment directive. Comment lines are also useful for explaining complicated directives.

```
/* QUME (TPUBS/CARBON) PRINTER FOR SYSA
/* Name of environment file is SPOOL*>TP_QUM.ENV
/*
Attribute TP.QUM
Attribute TP_QUM
Attribute TPUBS
Attribute CARBON -Mandatory   /* CARBON must be specified.
Device QUME -Evfu_off
Priority_size 10              /* Print files less than 10 records first.
Node -Local                   /* Scan only SYSA's spool queue.
Format -Width 84 -Length 66 -Bottom_margin 4  /* Sets format of pages.
Header -RBC                   /* Use different characters for header page.
Message TPUBS Qume Printer
Warning -On
Async -Line 108 -Protocol TTY -Speed 1200
Log -Off
```

Attributes are one of the scheduling environment directives that define the type of printing device and its setup. Users must include at least one attribute with their spool requests to select a device. In this example, the device name is TP_QUM, which is also the name of the environment file. Notice also that the sample file includes the mandatory attribute CARBON. Users must include any mandatory attributes defined for a printer or their files will not print.

Environment Directives Summary

This section summarizes the five types of environment directives:

- Interface definition directives
- Scheduling requirement directives
- Queue search directives
- Formatting option directives
- Logging despooler directives

Full descriptions of these directives and their options are given later in this chapter.

Interface Definition Directives

The interface definition directives listed below define the hardware connection and protocol used for a particular type of printer.

Directive	*Description*
ASYNC	Defines a serial printer.
DISK_DBG	Directs print handler debugging messages to a disk file instead of the environment log file.
MPC4000	Defines a high-speed parallel printer using the new MPC4000 interface board that allows up to four printers on each board and a maximum of two boards.
PARALLEL	Defines a parallel printer with up to two connections on each URC board.
PLOT	Defines a Versatec™ or Gould™ plotter.
TCP/IP	Defines remote printers connected to the Prime TCP/IP network.
TTY	Defines a Centronics printer or is used to debug a new print handler.

Scheduling Requirement Directives

The scheduling requirement directives listed below instruct the despooler phantom which jobs to print and in what order to print them.

Directive	*Description*
ATTRIBUTE	Defines the operation of printer features and capabilities supported by a particular printer.
MAX_SIZE	Sets the maximum size (disk records) of files that a particular printer and environment will process.
MIN_SIZE	Sets the minimum size (disk records) of files that a particular printer and environment will process.
PRIORITY_SIZE	Gives priority to spool requests with a record size less than the size set for the despooler phantom that controls a particular printer and its environment.

Queue Search Directives

The queue search directives listed below determine how often the despooler searches both the local queue and remote queues for spool requests, and which queues it searches first.

Directive	*Description*
CONFIG	Defines the location of *common* spool queue commands.
DISK	Defines a pre-Rev. 21.0 partition containing a SPOOLQ* directory.
DISK_SCAN	Sets the interval for despooler configuration changes (affects pre-Rev. 21.0 queues).
NODE	Defines a network node location containing a SPOOL_QUEUE* directory.
POLL_RATE	Sets the default poll rates for spool queue entries.
PRIORITY_TO_LOCAL_ QUEUE	Gives priority to requests in the local queue.
SWO_TIMES_TO_CHECK	Defines the number of times the spooler sleeps when it reaches the end of a SPOOL_WHILE_OPEN print job.
SWO_WAIT_TIME	Defines the time the despooler waits when it reaches the end of a SPOOL_WHILE_OPEN print job.

Formatting Option Directives

The formatting option directives listed below define the device type, format, paper type, and other print options. For PostScript printers, certain attributes can also determine output formatting.

Directive	*Description*
AUXILIARY	Passes environment parameters to user-written print handlers.
DEVICE	Selects a print handler for controlling a particular type of printer, such as Dataproducts, PostScript, or Printronix™.

Directive	*Description*
DPTX	Defines the name of an IBM® printer.
FORMAT	Defines the page format, such as page width and length, top and bottom margins.
HEADER	Sets the header characteristics of print jobs.
MAP	Selects the name of a character mapping file when using the –XLATE option of the SPOOL command.
MESSAGE	Prints the text defined for message on each header page.
PLOT	Tells the Spooler that the output device is a Versatec or Gould plotter.
SET_FONT	Defines the printer fonts and the character sequences used by the printer to select the font named with the SPOOL command.
SET_LANDSCAPE	Enables landscape printing.
SET_PAPER_BIN	Selects the paper bin used for feeding paper to the printer.
SET_PORTRAIT	Enables portrait printing.
TRAILER	Enables or disables printing of trailer pages.

Logging Despooler Directives

The logging despooler directives listed below define the type of despooler activity information recorded in the environment file logs.

Directive	*Description*
LOG	Determines the logging action once the despooler has been started successfully.
PERCENT_SPOOLED_ INCREMENT	Determines the increment used to calculate the percentage spooled.
WARNING	Enables or disables the printing of warning messages at the printer itself for certain printer control actions.

Matching Requests With Printer Environments

Despooler phantoms control printer environments. When you start a despooler phantom to start an environment, either at system cold start or with the PROP command, the environment file information is copied into the environment log file (if LOG is turned ON). The despooler phantom then searches the spool queues for requests containing attributes that can be processed by its associated environment file, and prints only those files.

The despooler phantom determines which spool requests it can process by matching the SPOOL command attributes with the attributes in the environment file that it controls. When spooling a file, you include attributes to select a printer and the capabilities you need to print your file, such as a printer type, paper type, or other capability that is supported by the printer you select. The same capabilities that you request with the –ATTRIBUTE option of the SPOOL command are also defined in the environment file using the ATTRIBUTE directive.

Spool requests are printed by the first available despooler phantom whose environment file attributes match those given with the SPOOL command.

Refer to Chapter 2, Setting Up the Spooler Subsystem, for additional information on using attributes.

Determining Spool Requests to Process

A despooler phantom uses the following criteria to determine which spool requests to process:

- The SPOOL command attributes match the attributes defined in the environment file.
- The spool request includes all mandatory attributes defined in the environment file with the ATTRIBUTE –MANDATORY directive.
- The file size is within the permitted range defined by the MIN_SIZE and MAX_SIZE directives in the environment file.

If users do not include attributes with the SPOOL command, or if they omit mandatory attributes, default attributes can be supplied within special files called attributes files. In order for the SPOOL command to verify attributes, you must first create the attributes files containing default attributes, as explained in Chapter 2, Setting Up the Spooler Subsystem.

Controlling Printers With Despooler Phantoms

Each despooler phantom controls only one output device, and each despooler phantom controls a different output device. You cannot use more than one despooler phantom to control the same printer (unless you are spooling over

TCP/IP which allows more than one despooler and its associated printer environment to share a printer).

While each despooler phantom is running, it uses its own copy of the environment data contained in the environment log file. This means that you can modify an environment file while a despooler phantom is running without affecting the currently running environment. An environment file is activated when you start, reset, or verify the environment with the PROP command.

Creating Environment Files

Environment files are ASCII text files that you create. They must be located in the SPOOL* directory. You can create and edit environment files with any standard text editor such as ED or EMACS. For instructions on using ED and EMACS, see the *New User's Guide to Editor and Runoff* and the *EMACS Reference Guide.*

Naming Environment Files

When creating an environment file, use the following filename convention:

environment-name.ENV

environment-name is a name that you choose to identify a particular printer environment. The name that you choose must be a valid PRIMOS user ID. That is, it must be 1 through 32 characters long; begin with a letter; and contain only letters, digits, periods (.), underscores (_), and dollar signs ($).

Environment filenames must have the .ENV extension. For example, the name of the sample environment file shown previously is TP_QUM.ENV. Environment names can identify a particular printer or printing environment. For example, you might choose LQP to name the system's letter-quality printer. If a printer is set up specifically to print labels, you might use the name LABELS to identify the type of printing environment.

When you start or stop a despooler phantom, the phantom displays a message when the named environment is started or stopped, although the message may not display until the despooler phantom finishes processing to clear the print buffers.

Format of Environment File Entries

Each physical line in the environment file cannot exceed 128 characters. If you type more than 128 characters on a line, the line truncates. If you need more than one physical line of 128 characters for an environment directive, you can continue the entry on the next line and subsequent lines, up to a maximum of eight lines. If you use multiple physical lines for one environment file entry, you must place an ampersand (&) symbol at the end of each subsequent line to indicate that it is a continuation of the first physical line of the entry.

Once you have created and saved an environment file, issue the following command:

PROP ***environment-name*** **–VERIFY**

The PROP –VERIFY command verifies the syntax of the environment file and notifies you of any errors. *environment-name* is the name of the environment without the .ENV extension, such as RB_QUM. If you enter the actual name of the environment file, for example, RB_QUM.ENV, the Spooler displays an error message.

In the following example, PROP –VERIFY is used to verify the environment called RB_QUM:

```
OK, PROP RB QUM -VERIFY
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
/* QUME (GROUP ONE/QUME) PRINTER FOR SYSA
/* Name of environment file is SPOOL*>RB_QUM.ENV
/*
Attribute RB.QUM
Attribute RB_QUM
Attribute WHITE
Attribute GROUP1 -Mandatory   /* GROUP1 must be specified.
Device QUME -Evfu_off
Priority_size 10              /* Print files less than 10 records first.
Node -Local                   /* Scan only SYSA's spool queue.
Format -Width 84 -Height 66 -Bottom_margin 4  /* Sets format of pages.
Option "-Height" not recognized by this command. (PROCESS_FORMAT)
Header -RBC                   /* Use different characters for header page.
Message GROUP ONE'S Qume Printer
Warning -On
Async -Line 108 -Protocol TTY -Speed 1200
Log -Off
Error from Spooler (PROCESS_ASYNC-45):
      Invalid Environment File
ER!
```

The syntax error from the Spooler is highlighted by a double underscore for the purpose of this example. In the sample file RB_QUM, –Height is not recognized as a valid option of the format environment directive. Changing –Height to –Length will correct the syntax of this environment file, as shown below.

```
OK, PROP RB_QUM -VERIFY
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
/* QUME (GROUP ONE/QUME) PRINTER FOR SYSA
/* Name of environment file is SPOOL*>RB_QUM.ENV
/*
Attribute RB.QUM
Attribute RB_QUM
Attribute WHITE
Attribute GROUP1 -Mandatory   /* GROUP1 must be specified.
Device QUME -Evfu_off
Priority_size 10              /* Print files less than 10 records first.
Node -Local                   /* Scan only SYSA's spool queue.
Format -Width 84 -Length 66 -Bottom_margin 4  /* Sets format of pages.
Header -RBC                   /* Use different characters for header page.
Message GROUP ONE'S Qume Printer
Warning -On
Async -Line 108 -Protocol TTY -Speed 1200
Log -Off
OK,
```

Environment Directives

This section describes the five categories of environment directives, their options, and how you use them in environment files to define printer hardware features and capabilities, as summarized earlier in this chapter:

- Interface definition directives
- Scheduling requirement directives
- Queue search directives
- Formatting option directives
- Logging despooler directives

You can type environment directives in uppercase or lowercase letters. Several environment directives refer to special files and directories that are used by the Spooler subsystem. For a summary of the Spooler subsystem directories and their contents, see Chapter 2, Setting Up the Spooler Subsystem. Unless otherwise stated, all numeric arguments and parameters are in decimal.

Interface Definition Directives

The directives listed below define the type of output device, communication protocol, and the interface hardware. See Appendix F for information on the PDN support directive.

The ASYNC Directive: This directive defines the configuration of the asynchronous serial line connected to a printer. *n* is the number of the line to which the printer is connected.

Format:

```
ASYNC -LINE n [options]
```

The ASYNC environment directive is useful if you have only one line available and you move printers frequently, or change the printer connected to that line in order to test different types of printers. If you need this flexibility, configure the asynchronous lines using the ASYNC directive in the environment file instead of using the PRIMOS SET_ASYNC command. The PRIMOS SET_ASYNC command is used to connect printers to dedicated lines that are configured for a particular printer during system cold start.

The ASYNC environment directive is similar to the PRIMOS SET_ASYNC command except that

- Some PRIMOS SET_ASYNC command options are restricted to System Administrators logged in as SYSTEM, which means that equivalent options are not available with the ASYNC directive.
- Many of the SET_ASYNC options are used to set up lines for video display terminals (VDTs) and are not useful for printers.
- The ASYNC environment directive enables carrier detection with the –CARRIER_DETECT option, which is not available with the PRIMOS SET_ASYNC command.

The ASYNC directive options allow you to control the data line features, such as the data transmission speed, number of stop bits, parity type, and device handling settings (whether communication is duplex or half-duplex mode).

Table 3-1 shows the default settings used by the ASYNC directive options. You can override these defaults by giving different values with the options, as described later.

Table 3-1. ASYNC Option Default Settings

Default Option	*Description*
–CHAR_LENGTH 8	Sets the character length to eight bits.
–DATA_SET_CONTROL	Enables modems and port selectors to recognize when information is transmitted.
–ECHO	Sets the line to full duplex.
–MAXIMUM_SHUTDOWN_TIME	Tells the despooler to wait 5 minutes or 300 seconds before shutting down the Spooler when the print buffer is full (MST *nn*).
–NO_DATA_SENSE_ENABLE	Disables the DATA_SET_SENSE (DSS) protocol.
–NO_ERROR_DETECTION	Disables error detection.
–NO_LINE_FEED	Tells the line not to send a LINEFEED character for a RETURN character.
–NO_REVERSE_XOFF	Disables Reverse Flow Control (XON/XOFF) for the line.
–PARITY NONE	Disables parity checking.
–PROTOCOL TRAN	The line uses transparent protocol.
–SPEED 1200	Sets the line speed to 1200 bits per second.
–STOP_BITS 1	Specifies that a single stop bit is used.
–XOFF	Enables Data Flow Control from the system to the device with Ctrl-S and Ctrl-Q.

Options:

–CARRIER_DETECT *n*
–CD

Determines the interval at which the despooler checks the carrier detect line. *n* is the time period between checks, given in milliseconds. The default is 200 milliseconds. The despooler phantom waits if the line is low (or the printer is off). The PRIMOS SET_ASYNC command does not have an equivalent option.

–CHAR_LENGTH *n*
–CL

Sets the number of information and parity bits per character. *n* can be 5, 6, 7, or 8 (the default).

–DATA_SENSE_ENABLE
–DSE

Enables the DATA_SET_SENSE protocol, also known as the Reverse Channel protocol. This option is used for transmitting control information or for controlling the flow of data to devices that do not recognize the XON/XOFF. The default is –NO_DATA_SENSE_ENABLE.

–DATA_SET_CONTROL
–DSC

DATA_SET_CONTROL is required by modems and port selectors to recognize when a block of information is transmitted. The default is –DATA_SET_CONTROL.

–DATA_SET_SENSE $\begin{Bmatrix}\textbf{HIGH}\\ \textbf{LOW}\end{Bmatrix}$
–DSS

Supports devices that toggle an RS-232-C pin (usually pin 8) to indicate when they are busy/ready instead of using XON/XOFF. The DATA_SET_SENSE protocol sets *ready_value* as either HIGH (pin signal raised) or LOW (pin signal lowered). The default is HIGH. The –DSS option must be enabled. When using –DSS, you must also use –NO_XOFF and –DSE and –NO_DSE to enable and disable Data Flow Control. DATA_SET_SENSE is also referred to as Buffered protocol or Reverse Channel protocol.

–ECHO

Sets the line to full duplex. For TTY protocols, data sent to the printer is echoed back to the printer; TRAN protocols do not echo.

–ERROR_DETECTION
–ERRDET

Replaces the incoming character with an ASCII 255, NAK character if a parity error or an input buffer overflow is detected. The default is –NO_ERROR_DETECTION.

–LINE_FEED
–LF

Echoes a LINEFEED character for a RETURN character in addition to echoing the RETURN character. This setting is useful for video display terminals, but not needed for printers. –NO_LINE_FEED is the default setting.

–MAXIMUM_SHUTDOWN_TIME ***nnn***
–MST

Determines the number of seconds that the despooler waits before automatically shutting down the Spooler when the print buffer is full. *nnn* is any time from 10 seconds up to and including 300 seconds. The default is 300 seconds (or 5 minutes).

–NO_DATA_SENSE_ENABLE
–NODSE

Disables the DATA_SET_SENSE protocol (Reverse Channel), and is the default.

–NO_DATA_SET_CONTROL
–NODSC

Disables the –DATA_SET_CONTROL option.

–NO_ECHO

Sets the line to half duplex to prevent echoing for TTY protocols. This must be used for PostScript printers.

–NO_ERROR_DETECTION
–NOERRDET

Prevents the line from sending an ASCII NAK character if an input parity or input buffer overflow is detected. This is the default.

–NO_LINE_FEED
–NOLF

Does not send a line-feed character for a carriage return character. This is the default.

–NO_REVERSE_XOFF
–NOREVXOFF

Disables Reverse Flow Control for the line. This is the default.

–NO_XOFF

Disables Data Flow Control (Ctrl-S and Ctrl-Q). Use this option for devices that transmit these control key sequences for other reasons. –XOFF is the default.

-PARITY {ODD | EVEN | NONE}
-PAR

Sets the line parity to the desired setting or disables parity. The value can be ODD, EVEN, or NONE (the default). Parity checking is a common method of error detection. The transmitting device appends a parity bit to the end of each character based on the number of zeros or ones in the character. The receiving device calculates the parity as it reads the character, and, if the parity calculated agrees with the parity bit setting, accepts the character.

-PROTOCOL [*name*]
-PRO

Defines the format and the relative timing of data, where *name* may be any of the following:

Protocol Name	*Description*
TTY	Standard terminal protocol. Typically this is used for serial printers, such as QUME™ and PostScript.
TRAN	Transparent protocol. This is the default, and is used if you specify -PROTOCOL without an explicit protocol name. TRAN is used by lines connected to peripheral devices or to other computers. Choose TRAN when it is not necessary to echo input, convert carriage returns to linefeeds, or specifically acknowledge carriage returns and linefeeds. All characters pass as data unless XON/XOFF flow control is set.
TT8BIT	Behaves in the same manner as the TTY protocol except that the high-order bit (ASCII parity bit) is not forced on for each character entered at the terminal. All control characters are handled in the same manner as the TTY protocol.
TTY8	Adapts the standard terminal protocol, TTY, for devices that use Prime ECS.
TTY8HS	Equivalent to TTY8 but designed for older model AMLC boards (Model 5052 or 5054).

The obsolete protocols TTYHS, TRANHS, and TTYHUP are supported for older model 5054 AMLC controller boards.

For more information on communications lines and protocols, refer to the *System Administrator's Guide, Volume II: Communication Lines and Controllers.*

–REVERSE_XOFF
–REVXOFF

Enables Reverse Flow Control for a line connected to an ICS controller. This option is not generally used for printers. When the PRIMOS input queue is full, the ICS controller is able to send an XOFF character to the device to prevent more data from being sent to the computer. When the input queue is able to accept more data, the ICS controller sends an XON character to resume transmission. The default is –NO_REVERSE_XOFF.

–SPEED [*value*]

Sets the speed of the line. *value* can be any of the supported SET_ASYNC speeds, and J1, J2, J3, and clock. Numeric values set the baud rate directly. The default is 1200. CLOCK uses the programmable clock, as specified by the PRIMOS AMLCLK configuration directive, which is usually 9600. J1, J2, and J3 use the jumper speeds.

$$\begin{matrix}\textbf{–STOP\_BITS} \\ \textbf{–SB}\end{matrix} \quad \begin{Bmatrix}1 \\ 2\end{Bmatrix}$$

Sets the number of stop bits to 1 or 2.

–XOFF

Enables the use of XOFF (Ctrl-S) and XON (Ctrl-Q) for controlling the data flow from the computer to the device.

The MPC4000 Directive: This directive allows you to configure the line from the MPC4000 high-speed print controller to a Dataproducts, Centronics, or PostScript printer, and supports these printer interfaces:

- Dataproducts standard parallel interface
- Centronics standard parallel interface
- Parallel PostScript interface

The default settings for MPC4000 are

- No serial line support
- The eighth bit stripped from all data
- ODD parity

Format:

MPC4000 GP*n* [*options*]

Options:

GP*n*

Identifies the printer. *n* is the port number on the parallel interface which is a value from 0 to 7, inclusive.

–BAUD *value*

Defines the line speed of the asynchronous communications line. This option is required only if the interface is a parallel/serial connection to a PostScript printer. If –BAUD is omitted, the line is not enabled. The supported speeds (*values*) are

75	110	134.5
150	300	600
1200	1800	2000
2400	3600	4800
7200	9600	19200

If this option is specified without the –SERIAL_PARITY option, the despooler will use ODD parity as a default.

–CENTRONICS

Informs the despooler phantom to expect the different cable wiring pattern used by a Centronics interface.

–EIGHTHBIT

Prevents the eighth bit from being removed. This option should be specified for graphic and certain other specialized applications. If this option is not specified, the eighth bit is is stripped from each data byte sent to the printer.

–ONES_COMPLIMENT

Flips the data bits before sending them to the printer.

–PARALLEL_PARITY

Specifies either –PARALLEL_PARITY ODD or –PARALLEL_PARITY EVEN. This option is used with the –BAUD option to inform the despooler that the interface is a parallel connection to a PostScript printer. If this option is omitted, parity defaults to ODD. If the –BAUD option is omitted, the parallel interface is not started.

–SERIAL_PARITY

Specifies either –SERIAL_PARITY ODD or –SERIAL_PARITY EVEN. This option is used with the –BAUD option to inform the despooler that the interface is a serial connection to a PostScript printer. If this option is omitted, parity defaults to ODD. If the –BAUD option is omitted, the serial interface is not started.

–TIMER ***value***

Allows you to specify a sleep period for the despooler. *value* is specified in milliseconds and allows the despooler to accommodate its transmission rate to the speed of the printer. The following table is a guide to selecting a timer speed to match the speed of the printer:

Printer Speed	*Timer Value*
1200 lpm	2500
24 pg/m	2000
45 pg/m	1000
90 pg/m	500

If you do not specify this option, the default value of 2000 is set. System load can affect printer performance; if a printer is not running at its rated speed, you can decrease –TIMER in units of 100 milliseconds until the printer reaches its rated speed.

The PARALLEL Directive: This directive defines a parallel interface for a Centronics printer connection.

Format:

PARALLEL PR*n* [–CENTRONICS]
PLL

Options:

PR*n*

Identifies the printer type. *n* is the line number on the parallel interface that can be any number from 0 through 3 inclusive.

–CENTRONICS

Informs the despooler phantom to expect the different cable wiring pattern used by a Centronics printer.

The PLOT Directive: This directive defines the interface to a Versatec or Gould plotter. A special PRIMOS routine controls the Versatec and Gould plotters so you do not need to specify the line protocol and configuration. Specifying a print handler is optional.

Do not use the SPOOL –NO_FORMAT option with –PLOT since they both define types of print formats.

Format:

PLOT

The TTY Directive: This directive directs the despooler phantom to send the contents of a file to a terminal buffer or to a Centronics printer.

Format:

$$\text{-TTY}\left[\begin{Bmatrix}1\\2\end{Bmatrix}\right]$$

Options:

1 specifies the first Centronics printer and 2 specifies the second Centronics printer on the interface board. If you omit 1 or 2, the output is sent to a terminal buffer, and written to the log file (similar to the way information is recorded into a COMO file). This option is useful as a debugging tool when you are developing a print handler. You can start a despooler phantom process and view the output on your terminal screen.

The DISK_DBG Directive: This directive directs all printer output to a file using the filename format *mmddhhmm.ss*, which is the current month, day, hour, minute, and seconds, and continues appending to this file as long as the current environment is active.

Format:

DISK_DBG [–SEQ]

Option:

–SEQ

The –SEQ option creates a separate file for each file in the spool queue, using the filename format *mmddhhmm.seq*, using the current month, day, hour, and minutes. *seq* is the spool request number assigned to each file in the queue.

The TCP/IP Directive: TCP/IP is the Transmission Control Protocol/Internet Protocol (TCP/IP) support software. The TCP/IP support software allows you to use the despooler to control printers connected across the Prime TCP/IP network (an optional feature at Rev. 22.1, and included in subsequent revisions of the Spooler subsystem).

Format:

$$\text{TCP/IP}\begin{Bmatrix}\text{-ADDRESS } \mathit{a.b.c.d}\\ \text{-NAME}\begin{bmatrix}\mathit{name}\\ \mathit{alias}\end{bmatrix}\end{Bmatrix}\text{-PORT } \mathit{port\_no}\begin{bmatrix}\text{-DEFER\_TIME } \mathit{nn}\\ \text{-LTS\_SLEEP } \mathit{nn}\\ \text{-PAD\_LF}\\ \text{-RELEASE } \mathit{nnnn}\end{bmatrix}$$

Use this directive to define remote printers connected to the Prime TCP/IP network. Include the TCP/IP environment directive in the printer environment file. The TCP/IP environment directive establishes the address of the Terminal server, and the port number of the connected printer.

The printers must be connected to a Terminal server or similar device. The address of the Terminal server and the port number of the connected printer must be supplied in the despooler environment file.

Options:

–ADDRESS ***a.b.c.d***
–ADD

a.b.c.d is the internet address of the Terminal server (or another device) connected to the printer. This option cannot be used with –NAME. Use either –ADDRESS or –NAME.

–NAME $\begin{bmatrix} \textit{name} \\ \textit{alias} \end{bmatrix}$

name or *alias* is either the name or a recognized alias from the local HOSTS file. The despooler attempts to connect to the address associated with the name or alias in the HOSTS file. This option cannot be used with –ADDRESS. Use either –ADDRESS or –NAME.

–PORT ***port_no***

Indicates which port the printer is connected to on the Terminal server. It is not the outgoing port from the host machine.

–DEFER_TIME ***nn***
–DEFER

Allows you to change the number of minutes that a deferred print job must wait before processing. The default is five minutes. *nn* is a range 0–60 minutes inclusive. A print job is given a deferred status and deferred time of *nn* minutes automatically if processing is attempted while the socket is busy.

–LTS_SLEEP ***nn***

Sets the number of seconds that the socket waits (sleeps) before releasing at the end of a print job. The default is 0–60 seconds. The default is one second.

–PAD_LF

Pads any CR/LF (carriage return/linefeed) pairs into CR/LF/LF. Most Terminal servers adhere to the Telnet protocol, which converts CR/LF pairs to CR only. This option ensures that any transmitted pairs of CR/LF actually arrive as CR/LF, when using a Terminal server that adheres to the Telnet protocol.

–RELEASE ***nnnn***
–RLS

Releases the socket for *nn* seconds after processing each printing job. *nn* is a range 0–3600 seconds inclusive. This option alternates the processing of printing jobs between two environments that share the same printer. When one environment finishes printing a job, it releases control of the printer to the alternate environment.

Printing Over TCP/IP: When the system is correctly set up, the printer environments for remote printers are used in the same way as any other printer environment. For further details on using Prime TCP/IP software, see the *PRIMOS TCP/IP Guide*. The *PRIMOS TCP/IP Guide* contains information on how to configure the port used by the printer. This will be set up as a slave port, typically using XON/XOFF and running at the printer's asynchronous speed (generally 1200 or 9600 baud).

Each printer connected over TCP/IP can service more than one despooler phantom and its associated printer environment. Even on the same node it is possible to have two or more despooler phantoms attempting to transmit data to the same printer address/port. If two printers attempt to connect to an address/port to transmit data, then the second (latest) printer is deferred for five minutes, unless you change the number of minutes using the –DEFER option of the TCP/IP environment directive.

Scheduling Requirement Directives

The environment directives listed below define which print jobs in the spool queue the despooler phantom and its associated environment file will process.

The ATTRIBUTE Directive: This directive defines the type of printer and the printing capabilities supported by that printer.

Format:

```
ATTRIBUTE name  -MANDATORY
ATT             -MAN
```

Options:

name

You can give up to 10 attribute names on the same line with the ATTRIBUTE directive. At least one *name* designates the device type. Additional attribute names may designate other printer features or printing capabilities of a particular printer. *name* can be up to 16 characters long.

-MANDATORY

Defines a required attribute which means that the environment will only process spool requests with that attribute. The request can contain other attributes defined in the environment file, but it must contain any MANDATORY attributes.

You can enter up to 32 ATTRIBUTE directives in the environment file. See Chapter 2, Setting Up the Spooler Subsystem, for a full discussion of attributes.

The MAX_SIZE Directive: This directive sets the maximum disk record size *n* of files printed by the despooler phantom and its associated environment. If you do not include MAX_SIZE in the environment file, the despooler phantom will process all files for that environment regardless of their size.

Format:

MAX_SIZE *n*
MAX

The MIN_SIZE Directive: This directive sets the minimum disk record size *n* of files printed by the despooler phantom and its associated environment. If you do not include MIN_SIZE in the environment file, the despooler phantom will process all files for that environment, including empty files (zero disk records).

Format:

MIN_SIZE *n*
MIN

The PRIORITY_SIZE Directive: This directive gives priority to spool requests with a file size of less than *n* disk records.

Format:

PRIORITY_SIZE *n*
PS

Caution If you use the PRIORITY_SIZE directive, the despooler phantom uses a more complex method to search the spool queue, which reduces overall performance. Use this directive only when it is important to print small files with a minimum delay.

Queue Search Directives

The directives listed below define which spool queues the despooler phantoms search and how frequently they search each queue for spool requests.

The options to the DISK, NODE, and POLL_RATE directives allow you to set specific intervals at which the despooler phantoms check spool queues for requests.

The CONFIG Directive: This directive defines the optional file called QUEUES located in the SPOOL*. You can create the optional QUEUES file to enter one set of spool queue search directives for multiple printer environments designed for the same printer. After you create the QUEUES file, enter the CONFIG QUEUES directive instead of entering individual queue search directives in each environment file.

Format:

CONFIG ***filename***

The DISK Directive: This directive defines the name of a partition containing a pre-Rev. 21.0 spool queue. The options described below determine how frequently the despooler phantom polls the spool queue for spool requests. You can have none, one, or several DISK directives in an environment file, provided that the combined number of DISK and NODE directives does not exceed 100.

Format:

DISK ***name*** [***options***]

Options:

name

Specifies the name of a disk partition.

–FAST_INTERVAL *n*
–FI

Establishes the fast poll rate. *n* is the interval in minutes. The default interval is one minute.

–SLOW_INTERVAL *n*
–SI

Establishes the slow poll rate. *n* is the interval in minutes. The default interval is five minutes.

–QUEUE_SEARCH_INTERVAL *n*
–QSI

Sets the length of time the despooler phantom waits before retrying an unavailable spool queue. *n* is the number of minutes. The default interval is 10 minutes.

–DECAY *n*
–DCY

Determines the number of times the despooler phantom polls a spool queue unsuccessfully at the fast rate, before changing to the slow rate. The default is 10 attempts at the fast rate.

–DISK_SCAN *n*
–DSCN

Sets the interval at which the despooler scans for configuration changes and reconstructs its list of disks on remote nodes. *n* is the number of minutes; the default is 10. DISK_SCAN affects only pre-Rev. 21.0 spool queues that are configured by nodename, and is used only in conjunction with the directive NODE –PRE_REV_21.

The NODE Directive: This directive defines the node containing a spool queue. An environment file can contain none, one, or several NODE directives, provided that the combined total of NODE and DISK directives does not exceed 100.

Format:

NODE [*nodename*] [*options*]

Options:

nodename

Specifies a PRIMENET™ nodename. If you omit *nodename*, the default is the local node. If the environment file does not contain any NODE or DISK directives, the default is NODE –LOCAL.

–LOCAL
–LCL

Defines a local system that is not in a network. It does not require a *nodename*.

–PRE_REV_21
–PRE

Defines a Rev. 20.2 or earlier node. Spool queues at Rev. 21.0 and later are identified by nodename. Spool queues at Rev. 20.2 and earlier are identified either by nodename or by disk partition name.

If you use this option, *all* disks on the remote node that are added to the local node are included in the table of disks that the despooler uses to locate spool queues. The table is updated at regular intervals as defined by the DISK_SCAN directive.

If you do not use the –PRE_REV_21 option, the despooler phantom assumes that *nodename* is a Rev. 21.0 node. When the despooler phantom finds that the node is not a Rev. 21.0 node, it automatically adds all known disks on that node to its table of pre-Rev. 21.0 queue tables.

–RING
–RNG

Directs the despooler phantom to search for a spool queue on all nodes in the PRIMENET ring, except the local queue, when the despooler phantom is started.

Although this method simplifies queue searching, it can degrade performance because the despooler phantom may search nodes that do not contain a spool queue.

The remaining NODE directive options enable you to control how frequently the despooler checks the spool queues for requests.

–FAST_INTERVAL *n*
–FI

Establishes the fast poll rate. *n* is the interval in minutes. The default interval is one minute.

–SLOW_INTERVAL *n*
–SI

Establishes the slow poll rate. *n* is the interval in minutes. The default interval is five minutes.

–QUEUE_SEARCH_INTERVAL *n*
–QSI

Sets the length of time the despooler phantom waits before retrying an unavailable spool queue. *n* is the number of minutes. The default interval is 10 minutes.

–DECAY *n*
–DCY

Sets the number of times the despooler phantom polls a spool queue unsuccessfully at the fast rate, before it changes to the slow rate. The default is 10 attempts.

The POLL_RATE Directive: This directive sets the poll rate parameters that determine how frequently the despooler phantom searches each spool queue for requests. You can give different values in individual NODE and DISK directives.

Format:

POLL_RATE [*options*]
PLRAT

Options:

–FAST_INTERVAL ***n***
–FI

Establishes the fast poll rate. *n* is the interval in minutes. The default interval is one minute.

–SLOW_INTERVAL ***n***
–SI

Establishes the slow poll rate. *n* is the interval in minutes. The default interval is five minutes.

–QUEUE_SEARCH_INTERVAL ***n***
–QSI

Sets the length of time the despooler phantom waits before retrying an unavailable spool queue. *n* is the number of minutes. The default interval is 10 minutes.

–DECAY ***n***
–DCY

Sets the number of times the despooler phantom polls a spool queue unsuccessfully at the fast rate before it changes to the slow rate. The default is 10 attempts.

The PRIORITY_TO_LOCAL_QUEUE Directive: This directive gives priority to requests in the local queue. Normally, when a despooler finishes printing a file, it checks the same queue for more spool requests that it can process. If a remote spool queue contains numerous requests for that despooler phantom to process, it could delay the spooling of requests in the local queue.

The PRIORITY_TO_LOCAL_QUEUE directive forces the despooler to search the local queue after it finishes printing a file, if polling of the local queue is due at that time.

Format:

PRIORITY_TO_LOCAL_QUEUE
PLQ

The SWO_TIMES_TO_CHECK Directive: This directive sets the maximum number of times the despooler sleeps at each end of file (EOF) during a SPOOL_WHILE_OPEN operation. The despooler phantom stops printing when it reaches the end of a SPOOL_WHILE_OPEN file and waits for *n* seconds (defined by SWO_WAIT_TIME) before checking the file for further processing which is set by SWO_TIMES_TO_CHECK.

Format:

SWO_TIMES_TO_CHECK *n*
SWOTTC

Option:

n

Shows the range for *n* is 0 through 32767 inclusive; 1 is the default. If *n* is zero, the despooler phantom stops printing when it reaches the end of file and does not attempt further processing. If *n* is set to 1, the despooler sleeps for *n* seconds (set by SWO_WAIT_TIME) at EOF, reads the file again, and stops printing. If EOF occurs immediately, the file is also read again and printing stops. If *n* is set to 2, the despooler sleeps and retries twice at EOF. If an error occurs at EOF for each retry, the despooler stops printing.

The SWO_WAIT_TIME Directive: This directive defines the time the despooler waits when it reaches the end of file (EOF) during a Spool While Open function.

Format:

SWO_WAIT_TIME *n*
SWOWT

Option:

n

Indicates number of seconds that the despooler sleeps. *n* is a range from 1 through 600 seconds. The default is five seconds.

Configuring Spool Queue Searching: This subsection describes various ways of defining how the despooler phantom searches the spool queues and how frequently the despooler phantom checks each queue for requests.

If performance is important, use the NODE and DISK directives in the environment file to define the exact location of the spool queues you want the despooler phantom to search.

The total combined number of NODE and DISK directives must not exceed 100, including any that are automatically generated. For example, defining a node as pre-Rev. 21.0 generates DISK directives for all added disks on that node. Similarly, the NODE –RING directive generates a NODE directive for each node on the local ring. You can use the –VERIFY option of the PROP command to verify that you have not exceeded the limit.

Defining Common Queue Search Directives for Multiple Environments: If several despooler phantoms search the same spool queues, you can put the NODE and DISK directives in a separate file called QUEUES in the SPOOL* directory, and refer to that file with a single CONFIG directive in each environment file. (Do not confuse the QUEUES file in SPOOL* with SPOOL_QUEUE*>QUEUE, which is the spool queue itself.)

Searching Remote Spool Queues: If you use the NODE –RING directive in each environment file, the despooler phantoms search all nodes for spool queues in the local ring (except for the local machine itself). The NODE –RING directive includes only nodes on systems other than the local system. To search the local machine, you must include NODE –LOCAL.

Although this method of queue searching provides certain flexibility, it is less efficient and may degrade performance because the despooler phantom must search all available disks for spool queues, including disks that do not contain a spool queue.

Polling Spool Queues for Requests: The despooler phantom uses the fast poll rate if the queue contains requests for processing each time it searches the queue. If the despooler phantom searches the queue and does not find requests to process after multiple tries, it changes to the slow rate. The –DECAY option sets the number of polling attempts that the despooler makes at the fast rate before changing to the slow rate. When the despooler phantom finds requests in the queue, it changes back to the fast poll rate. If a spool queue is unavailable when the despooler searches it (for example, when a disk is offline for some reason), it retries after the interval which is set by the –QUEUE_SEARCH_INTERVAL option.

Formatting Option Directives

The directives in this group control the output format of the printed file.

The AUXILIARY Directive: This directive enables you to pass environment parameters to custom print handlers, the header page, and other custom routines. The environment parameters are read from the environment file, and are inserted into an auxiliary structure in external common memory where they can be accessed by the users' own routines.

The auxiliary structure is not used by the standard despooler. It is provided for custom routines. The content and use of these parameters are the user's responsibility.

By using the AUXILIARY directive in a printer environment file, you can include the element in the auxiliary structure that contains a defined string.

Format:

AUXILIARY ***–element n string***
AUX

Option:

n

Is a range from 1 through 5, and is the element in the auxiliary structure containing the string. The string contains a maximum of 160 characters. The auxiliary structure, in SPOOL*>SOURCE>AUXILIARY.INS.PL1, contains an array of up to five elements with their string contents, stored as char(160) var strings. The declaration of the structure is

```
dcl auxiliary(5) char(160) var static external;
```

The DEVICE Directive: This directive defines a print handler that the despooler phantom uses. *handler* is the name of the print handler routine. You must include a DEVICE directive in the environment file except when defining a plotter with the PLOT directive, in which case DEVICE is optional and defaults to Printronix.

You can add print handlers for other devices as explained in Appendix B, Writing a Print Handler. The code for the print handlers is located in the file SPOOL*>DESPOOL_LIBRARY.RUN, which is a library class EPF.

Note For a Printronix plotter, you must give a DEVICE directive, and may need to give an ASYNC directive. You do not need to give either the DEVICE or ASYNC directive for a Versatec or Gould plotter. Give the special PLOT environment directive instead. For details, see the description of PLOT later in this chapter.

You can specify information that the despooler phantom needs to initialize the output device, and other output device information, in the options to the DEVICE directive.

Format:

DEVICE ***handler*** **[*options*]**

Options:

–DELAY [*nn*]

Delays processing *nn* seconds between pages to prevent buffer overflows at the printer. The default number of seconds is 4. This option is for use with the HPJET device only.

–DICT *pathname*

Defines a file that lists the pathnames of PostScript procedure files. (You can have several such files so that the system can support several PostScript printers.) The PostScript procedures are downloaded to the printer when the despooler phantom is started. At the start of each print job, the printer is checked to make sure that the procedures are still loaded. (The printer could have been switched off and back on again, for example.) If the procedures do not exist, they are reloaded. The default *pathname* is SPOOL*>POSTSCRIPT_INIT. Administrators can add the pathnames of additional PostScript procedure files to the .SPOOL*>POSTSCRIPT_INIT file.

–EVFU *pathname*

Defines the pathname of a file that contains an Electronic Vertical Format Unit (EVFU) definition. If –EVFU is given with no *pathname*, a standard EVFU format is used, with channel 1 assigned to line 1 and no other channels defined. Refer to Chapter 4 for more information on using EVFU.

–EVFU_OFF

Indicates that the printer has no available EVFU. In this case, the Spooler subsystem software emulates the EVFU function. This is the default.

–NO_LINEFEED
–NLF

Indicates that the device generates a linefeed when it does a carriage return, and that the despooler therefore does not need to insert line-feed characters. Note that a printer that requires this option cannot perform overstriking.

–POSTSCRIPT_CHECK
–POST_CHECK
–PC

This option is used for defining PostScript printers that print PostScript language files. –POSTSCRIPT_CHECK appends the –NO_FORMAT option to spooled files containing the standard PostScript sequence (%!PS) so that page formatting is performed by the PostScript file itself and not by the Spooler. The Spooler processes any other user-specified options. If the spooled file is a non-PostScript file (a regular text file), the file is formatted by the Spooler. If you spool a non-PostScript file to a PostScript printer with the –PC and –NOF options, a PostScript error displays. See Appendix D for more information on PostScript printing.

–POSTSCRIPT_OVERRIDE
–POST_OVER
–PO

This option is used for defining PostScript printers that print text files (non-PostScript files). –POSTSCRIPT_OVERRIDE ignores a –NO_FORMAT option that a user may supply when spooling a text file to a PostScript printer, thus preventing PostScript errors. See Appendix D for more information on PostScript printing.

–PI

You must specify a Printronix device to use this option. It indicates that jumper W5 on logic board A is made; that is, that the paper instruction is enabled. See the Printronix printer manual for details.

–STATIONERY_FEED_TYPE ***paper feed mechanism***
–SFT

Specifies one of the following types of paper feed mechanisms:

- MANUAL or MAN
- CUTSHEET or CUT
- CONTINUOUS or CON (the default)

–UPCASE
–UC

Indicates that the output device requires the data stream to be converted to uppercase.

The DPTX Directive: The DPTX (Distributed Processing Terminal Executive) directive specifies a suitable IBM *printer_name* when the DEVICE directive has previously been used to select the DPTX print handler. *printer_name* must already be defined in the DPTX configuration.

Format:

DPTX *printer_name*

The FORMAT Directive: This directive establishes the page formatting for files spooled without –NO_FORMAT, –FTN, –COBOL, and –PLOT.

Format:

FORMAT [*options*]

Options:

–BOTTOM_MARGIN *n*
–BM

Specifies number of blank lines to be left at the foot of each page. *n* is the number of lines. The default is two lines.

–LEFT_MARGIN *n*
–LM

Specifies width of left margin. *n* is the number of characters. The default is 0.

–LENGTH *n*
–L

Sets number of lines per page (form length). Your choice depends on the type of stationery used and any special requirements. *n* is the number of lines. The default is 66.

–PAD_CHAR *n*
–PCH

Specifies the character used to fill the last byte of an odd length line that is sent to a device with a parallel interface. *n* is the octal representation of the pad character in Prime ECS. The default is the space character, or 240, in ASCII.

–RIGHT_MARGIN *n*
–RM

Specifies the width of the right margin as number of characters. The value of the default is 0.

–TOP_MARGIN *n*
–TM

Specifies the number of blank lines to be left at the top of each page. *n* is the number of lines. The default is 4.

–WIDTH *n*
–W

Specifies page width. *n* is the number of characters. The default is 132.

The HEADER Directive: This directive enables you to print a file containing 0, 1, or 2 additional header pages. *n* is the number of header pages you specify; 1 is the default. The HEADER directive options enable you to control the header characteristics.

Format:

HEADER [*n*] [*options*]
HDR

Options:

–FACING_BANNER
–FB

Inserts a blank page between print jobs, when needed, so that the banner page of each file faces the same way on fan-folded listings.

–NO_INITIAL_FF
–NIF

Prevents the initial form feed that occurs when the Spooler is started. This option is useful when printing files on preprinted stationery that is loaded on the destination printer.

–SUPRESS_FILE_INFO
–SFI

Prevents file information, such as pathname and date and time modified, from being printed on the header and trailer pages.

The following additional options enable you to choose the characters that the printer uses to form the large outlined letters of the header page text. The default character is an uppercase W. (PostScript printers use a different method to form the large outlined letters of the header page text.)

–CHAR *c*
–C

Uses the character that you enter for *c* to form the large outlined letters of the header page text.

-ROLLING_BANNER_CHAR
-RBC

Uses a different printer character to form each letter of the header text into large outlined letters, which equalizes the use and wear of the printer characters.

-SAME

Uses the same character as each letter of the header text to form the large outlined letters. For example, if the header text is MYFILE, it uses the character M to form the large outlined letter M, the character Y to form the large outlined letter Y, and so on.

The MAP Directive: This directive defines a particular mapping file which is the same mapping file you specify with the SPOOL command –XLATE option. *mapping* is a character string that selects an alternate character mapping for the printer, which cannot exceed 32 characters. The first character must be alphabetic. The remaining characters can be alphanumeric or any one of the following characters: . $ or _. *filename* contains the map definition, and is located in the SPOOL*>MAPPINGS directory.

Format:

$$\textbf{MAP}\ \textit{filename mapping} \begin{Bmatrix} \textbf{-DEFAULT} \\ \textbf{-ALTERNATE} \end{Bmatrix}$$

For example:

```
MAP IBM ECS_TO_EBCDIC_MAP
```

where IBM is the map called with the SPOOL –XLATE command as shown in the following example:

```
SPOOL TEMP.FILE -XLATE IBM
```

ECS_TO_EBCDIC_MAP is the filename in which the IBM map definition is filed in the SPOOL*>MAPPINGS directory.

The –DEFAULT option is used to define a default mapping for the environment. If you do not define a default mapping, the *first* MAP directive in the environment file, excluding any MAPS defined with –ALTERNATE, is used as the default. The default map can be selected explicitly by name.

The –ALTERNATE option is used to define an alternate MAP for files spooled with the –XLATE option without *mapping*. If you do not define an alternate mapping, the first MAP directive that has *not* been defined as the default mapping is used as the alternate mapping.

A single environment file can have up to 64 MAP directives. Refer to Chapter 4 for more information on using character mapping.

If you enter the SPOOL command without the –XLATE option, the Spooler attempts character mapping in the following order:

1. Performs character mapping using the default MAP defined in the environment file.
2. Uses the STANDARD_MAP, if one exists, when a default map is not available.
3. Emulates the Rev. 21.0 version of character mapping if neither a default mapping or a standard mapping exists, and assumes that
   - Space compression has been used.
   - A line of text terminates with a linefeed.
   - A page eject occurs for each form feed in the file.
   - The input files are in ECS and the output is in ASCII.

The MESSAGE Directive: This directive prints a message on each header page. The maximum number of characters for message *text* is 160 characters.

Format:

MESSAGE *text*
MES

The PLOT Directive: This directive tells the Spooler that the output device is a Versatec or a Gould plotter. A special PRIMOS routine controls this type of plotter, so there is no need to include the line protocol and configurations. Using a print handler is optional.

Format:

PLOT

The SET_FONT_NAME Directive: This directive defines the fonts available for a printer environment. You can define up to 32 fonts in an environment file. For information on PostScript font options, see Appendix D, Defining and Using PostScript Printers.

Format:

SET_FONT_NAME *fontname* –SEQ *escape_sequence* $\left\{\begin{array}{l}\text{–STRING}\\ \text{–REP}\end{array}\right\}$ [–DEFAULT]

Options:

fontname

The name of the font used with the SPOOL –SET_FONT command to select the desired font. *fontname* cannot exceed 32 characters.

–SEQ *escape_sequence*

Defines the printer-specific codes that the printer needs to activate the selected font. The *escape_sequence* entry is located in the printer's instruction book.

–STRING

Defines the –SEQ *escape_sequence* as the actual string itself.

–REP

Defines the –SEQ *escape_sequence* as the decimal (base 10) representation of characters.

–DEFAULT

Establishes the default font for the destination printer when spooling a document without –SET_FONT instructions. Only one font can be defined as the default font for the environment. If you do not establish a default font, the printer may select one automatically.

The SET_LANDSCAPE Directive: This directive establishes landscape page orientation that prints text across the longest dimension of the paper.

Format:

SET_LANDSCAPE [–DEFAULT]

Option:

The –DEFAULT option establishes SET_LANDSCAPE as the default page orientation for the printer environment if a page orientation is not selected with the the SPOOL command.

Refer to Appendix D, Defining and Using PostScript Printers, for more information about the SET_LANDSCAPE options for PostScript printers.

The SET_PAPER_BIN Directive: This directive defines the paper bin used by the printer.

Format:

SET_PAPER_BIN $\left\{ \begin{matrix} n \\ \text{MANUAL} \end{matrix} \right\}$ **[–DEFAULT]**

Options:

n is a number from 1 through 9. MANUAL sets the paper bin to manual paper feed. –DEFAULT defines the paper bin that is used if none is specified when spooling a document.

The SET_PORTRAIT Directive: This directive establishes portrait page orientation that prints text across the shortest dimension of the paper. Portrait is the standard page orientation and is the default.

Format:

```
SET_PORTRAIT [–DEFAULT]
```

Options:

The –DEFAULT option of SET_PORTRAIT establishes portrait page orientation as the default if no page orientation is selected with the SPOOL command.

For information on SET_PORTRAIT options for PostScript Printers, refer to Appendix D, Defining and Using PostScript Printers.

The TRAILER Directive: This directive sets the printing of trailer pages ON or OFF. Trailer pages contain the following information:

- Date and time the job was completed
- Request number of the job
- Number of pages printed, excluding header and trailer pages
- User ID
- Project ID
- Nodename
- Pathname of the spooled file

The pathname is not printed if the environment file contains a HEADER –SFI directive or if the –SFI option is used with the SPOOL command.

Format:

```
TRAILER  {–ON }  {–PERF n [–PERF_CHAR n]}
TRLR     {–OFF}  {–NO_PERF              }
```

Options:

–ON
Prints a trailer page at the end of each document.

–OFF

Turns trailer page printing off. This is the default setting.

–PERF *n*

Prints *n* full lines of a character that you define before and after the perforation that occurs between print jobs, and prints one full line on the perforation itself. The default value of *n* is two lines.

–NO_PERF

Prevents the printing of the perforation marker between jobs. This is the default setting.

–PERF_CHAR *n*

Defines the character used to print *n* full lines before and after the perforation between print jobs, and on the perforation itself. If you do not define a perforation character, the rolling banner character is used which selects different characters at random.

Logging Despooler Directives

When you use the PROP command to start a despooler phantom, it reads and validates the environment file, then writes a copy of the environment file into the environment log file using the following pathname:

SPOOL*>LOG>*environment*.LOG

environment is the environment name.

The LOG Directive: This directive determines additional logging actions after the despooler has started successfully.

Format:

LOG [*options*]

Options:

–ERROR_NOTIFY [*user1*] [*user2*] [. . . *user8*]
–ERR_NTFY

Sends printer error messages to a maximum of eight users named in the list. Printer error messages are always sent to the system console. The user names can be on either the local system or remote systems. This directive is used only for printers connected over TCP/IP or ASYNC.

–ON

Records information about the despooler's activities, but does not close the log file. A single line of accounting information is recorded for each job. The next startup of the same environment file overwrites the file.

–OFF

Closes the log file after startup. The next startup of the same environment overwrites the file.

–SAVE

Performs the same function as the –ON option, but appends the date and time to the name of the log file, as shown below:

```
SPOOL*>LOG>environment.LOGdatestamp
```

environment is the environment name and *datestamp* is the full date in the format *yymmddhhmmss*. This file is not overwritten the next time the environment is started because a different date stamp is generated for each successive filename. You need to delete old log files occasionally to prevent the disk from becoming full.

Displaying Spooler Status: The SPOOL –LIST –DETAIL or SPOOL –LIST –FULL command displays the percentage printed of files currently printing. See Chapter 1 for an example of these displays. Each time the despooler phantom finishes processing a page, the number of records processed at that point is compared against two values. If the number of records processed since the last report is less than the reporting increment defined in the printer environment file, the percentage spooled value is not updated. If the number of records processed is greater than the reporting increment, then the number of records processed at that point is compared against the file size, and the percentage spooled is calculated and displayed. The same check is done for multiple copies, as each copy is started.

The PERCENT_SPOOLER_INCREMENT Directive: This directive determines the reporting increment used for calculating the percentage spooled. The default is 10 records. If you do not wish to see the percentage spooled values, you can set a large value for *n*, for example 32767, which represents 10 or more boxes of paper for a single file.

Format:

PERCENT_SPOOLER_INCREMENT *n*
PSI

Option:

n

Indicates the number of records that must be processed before the percentage spooled value is updated.

The WARNING Directive: This directive prints a warning message in large outlined letters at the printer, when starting or stopping a printer environment, or when dropping a file. The default setting is –ON.

Format:

$$\text{WARNING} \begin{Bmatrix} \text{–ON} \\ \text{–OFF} \end{Bmatrix}$$

Modifying Printer Environment Files

If you need to change an environment file, you can use a text editor such as ED or EMACS. Before modifying an environment file, you may want to save a copy of the original environment file in case you make mistakes in the copy you are editing. After you modify the environment file, you must restart the environment to use the modified version.

The Spool Administrator should create an environment file for each different environment needed. If you need an environment that is slightly different from another that you have created, it is best to create a new environment file for the new one, rather than modify one that works successfully. You can stop an active environment and start a new one in its place with the PROP –RESET command.

Controlling Printer Formatting

4

Introduction

This chapter explains how to control printer formatting using the following Spooler features:

- Character mapping
- Electronic Vertical Format Unit (EVFU)

Character Mapping

Character mapping enables experienced programmers to convert characters in spooled files from one character set into an alternate set of characters at the printer.

The Spooler subsystem performs character mapping by default because some printers do not support the Extended Character Set (ECS) used by Prime computers. Printers that do not support ECS may not print the same characters that you see when displaying the file on a terminal screen. Character mapping is also useful if a printer does not support certain characters or symbols.

Character mapping is performed using mapping files. Each mapping file describes a mapping from one character set to an alternate character set at the printer.

Prime supplies two character mapping files in the SPOOL*>MAPPINGS directory (at Rev. 22.0 and later):

- ECS_TO_ASCII_MAP converts ECS characters to ASCII characters and is the default mapping used by the despooler.
- QUME_OAS_MAP is for use with QUME printers that require OAS support.

How the Despooler Manipulates Text for Printing

The despooler must be able to manipulate text it receives for printing. For example, if you want line numbers printed in the file, the despooler must be able to add line numbers to the front of each line. The despooler retrieves any additional text that it needs to process a spooled file from SIT (Software Internationalization Tool).

The despooler adds the text that it retrieves from SIT, PRIMOS, the environment file, and the SPOOL command, to the text of the spooled file. For example, the despooler adds the fixed text printed on header pages, the date and time the file was spooled, the header line for each page of output, the value of the message field that is printed on the header page, and an alternate value for the header field (if one is supplied).

The despooler passes all of the text to a print handler for printing. The print handler performs the output mapping. If a printer does not have an equivalent character value for a particular character in the file, an asterisk (*), for example, a sequence of characters can be used to print a representation. To print an asterisk, you could use the character sequence x **backspace** + which prints an asterisk by printing the x on top of the +.

The Mapping Process

Character mapping is performed in two steps:

1. Input mapping is perfomed when the printer reads the file for printing. Input mapping ensures that the text of the file is in a consistent character set.
2. Output mapping reads the text generated by the input mapping process, merges text that the despooler retrieves from other sources, such as page header information, and then passes that text to the output mapping routines that convert the characters into the character set used by the printer.

When a despooler phantom reads a file for printing, it merges the text in the file with other text that it retrieves from SIT and other sources, and passes the combined text through the input mapping process. The input mapping process must produce the mapped text in a compatible character set.

Figure 4-1 shows the character mapping process.

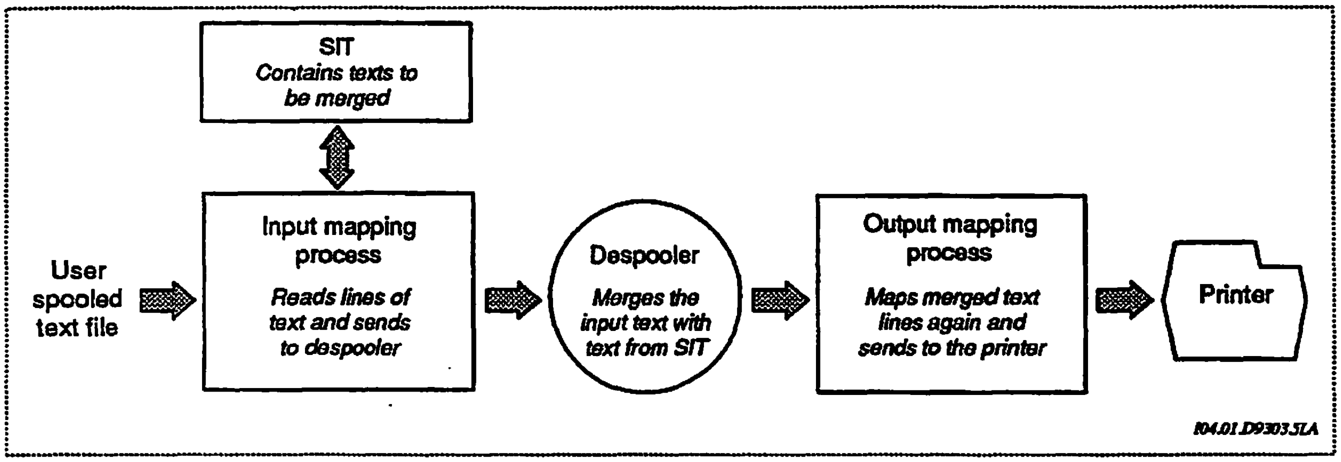
SIT
Contains texts to be merged
User spooled text file
Input mapping process
Reads lines of text and sends to despooler
Despooler
Merges the input text with text from SIT
Output mapping process
Maps merged text lines again and sends to the printer
Printer
104.01.D9303.5LA

Figure 4-1. Character Mapping Process

The input mapping and output mapping operation is controlled by a **map** which is a set of tables used by the mapping routines. Each map contains all of the information required to perform both the input and the output mapping processes. Maps are generated using the CREATE_MAP.RUN utility from source files that you create, as explained in the next section.

Defining Maps

To define a map, you create a **source file**, which is a text file containing comments, keywords, and numbers. Source files are processed through a program called CREATE_MAP.RUN in SPOOL* which converts the source file input into structures referred to as maps. The maps (structures) are loaded into memory for use by the mapping routines which perform the input mapping and output mapping processes.

Contents of Mapping Source Files

Mapping source files contain comments, keywords, and numbers. You enter the keywords at the beginning of a line in the source file. The keywords and their abbreviations are listed in tables that follow. A detailed description of each keyword is provided later in this chapter.

The keywords REXC, RMAP, WALTMAP, WEXC, and WMAP introduce a list of values that must be terminated by END. You can enter the same keyword more than once in a file, but you cannot nest keywords or embed another keyword in a list.

Each comment line begins with a /* and consists of one physical line. Keyword entries can continue on multiple physical lines, forming a single logical line. Each physical line of a single logical line must end with an ampersand (&) to indicate that the logical line continues on the next physical line.

Numbers must be in decimal and range from 0 through 255. These numbers describe various character values to the mapping routines. For example, to map the number zero in EBCDIC to the number zero in ECS, you would need a table of EBCDIC values where zero is 0, and a table of ECS values where zero is 176.

See sections later for more details on how to do this.

Building a Map

After you create the mapping source file containing keywords, comments, and numbers, you run the source file through the program CREATE_MAP.RUN in the SPOOL* directory.

To run this program, you must be attached to the directory containing the mapping source file, then enter the following command line:

R SPOOL*>CREATE_MAP *source_file*

For example:

R SPOOL*>CREATE_MAP SEVEN_BIT_FRENCH_MAP_SRC

where SEVEN_BIT_FRENCH_MAP_SRC is the name of the mapping source file that contains mapping keywords.

The CREATE_MAP program reads the source file line by line, and displays it on the terminal. The program prompts with `--More--` at the end of each screen. If you wish to override the prompt, enter -NO_WAIT on the program command line. If the source file contains errors, the program displays error messages beginning with the word ERROR, followed by an explanation.

If there are no errors in the source file, the program prompts for the name of an output file. The program writes the map out to the filename that you enter at the prompt. If the filename exists, the program prompts you to overwrite the file. Entering Quit stops the program. After the program writes the map file, it is ready to be loaded directly into memory. The map must be placed in the SPOOL*>MAPPINGS directory so that the despooler input processing routine can find it.

Input Mapping Process

Input mapping constructs lines of text for processing by the despooler input routine which reads the characters in the file submitted for printing and converts them to the character set used by the despooler for printing.

The despooler input routine reads each individual character of text until it reaches an end of line or a form-feed character. The despooler then processes that line without the end of line or form-feed character, but with a flag that indicates which character was used to end the line. If the line ended with a form-feed character, a new page is output after printing the line. If the despooler routine encounters a space compression character while reading the line, it uses the character `ignoring NUL 0` as the count for the number of spaces inserted. All other characters are read and returned.

The primary function of mapping is to map each character to an equivalent character in the alternate character set. For example, all ECS characters are mapped to an equivalent character in ASCII. Mapping can also handle exceptions, such as reading two characters in a file that represent one character at the printer. For example, mapping uses the characters \ a to represent the letter *a* with an accent grave.

Summary of Keywords for Input Mapping

Keyword	*Abbreviation*	*Description*
end_of_line	**EOL**	Defines the character that terminates a line of input.
form_feed	**FF**	Terminates a line of input and starts a new page.
form_feed_off	**FFOFF**	Disables testing for the form-feed character.
read_escape	**RESC**	Defines the value for testing whether special processing is required during read mapping.
read_exception	**REXC**	Defines a list of character sequences with instructions for their special processing.
read_map	**RMAP**	Defines a table of one-to-one substitutions.
space_compress_off	**SCO**	Disables the expansion of a space compression character.
END	**END**	REXC and RMAP introduce a list of values that must be terminated by END.

Input Mapping Keywords

This section describes the keywords used for input mapping, with examples.

End_of_line

Defines the character that ends a line of input. All files read for mapping must contain a character that ends a line of text. An EOL 141 resets the end of line character to CR (carriage return). The default EOL is 138.

When the input mapping routine reads an EOL character (not a form_feed character), it always reads the next character as the begining of a word (16 bits). If the EOL occurs at an odd-numbered byte (counting from 1), it reads the next character as an odd-numbered byte.

Form_feed

Defines the character that causes the line to be terminated and a new page to be started at the end of the line.

If the input character set uses a character other than FF to signal page throws, then form_feed can be used to reset it, e.g., FF 137 would reset the page throw character to HT (horizontal tab). Changing FF has no affect on the sequence which is output to affect a page throw. The default is form_feed 140.

Form_feed_off

Disables testing for a page eject character. Detecting for a form_feed character is not always required, particularly if the input character set does not have a character that produces a page eject. Form_feed_off ensures that a new page is not generated by mistake.

Read_escape

Defines the character for special processing. The read_escape character is set using RESC. For example, RESC 255 resets the read_escape character to DEL. The default is RESC 155.

Certain character sequences require special processing (see Read_exception). The input routine detects these character sequences by testing the value returned after it finishes the one-to-one character mapping. If the value returned is the read_escape character, then it performs the special processing. If the character mapped to read_escape does not form a read_exception sequence, the input routine reads the original value of character mapped to the read_escape character.

Read_exception

Defines sequences of characters for special processing and the instructions for processing. Two types of character sequences require special processing. Character sequences that are passed directly to the despooler without the normal input processing, and pairs of characters that represent a single character.

When input processing detects a read_escape character, it retrieves and uses the original value of that character plus the following character to search a list of exceptions. If the character mapped to read_escape does not indicate special processing, the original value of the first character is passed through, and the second character is processed as normal.

Read_exception defines the list of exceptions. Each line following read_exception defines one entry in the list. Each line consists of a pair of character values, a type value that indicates the type of processing required, and other information.

The following types are available:

- Fixed length transfer with no substitution (type = 0), for example:

  ```
  155 193 0 2  /* transfer ECS A and next 2 as read
  ```

  The initial pair and the next *n* (*n* is 0 through 255) characters are transferred without performing the one-to-one substitution. While transferring a sequence, the usual end of line test, space decompression, and form-feed processing are not performed.

- Variable length transfer with no substitution (type = 1), for example:

  ```
  155 194 1 187  /* transfer ESC B until (and including);
                 /* as read
  ```

  The initial pair and all characters following are transferred until it encounters an exact match. No substitution is performed. The transfer ends when the character 187 (;) is detected. The usual end of line tests are not peformed while transferring a sequence.

- Variable length transfer with no substitution (type = 2), for example:

  ```
  155 195 2 59  /* transfer ESC C until (and including);
                /* as read
  ```

  Type 1 performs an eight bit test, while type 2 tests only the lowest order 7 bits. In this example, the transfer ends when 59 or 187 is detected. The usual end of line tests are not performed while transferring a sequence.

- For two input characters, substitute one character (type = 3), for example:

  Assuming an RMAP statement exists,

  ```
  RMAP
  220 155  /*Map \ to the read_escape character
  END
  ```

then

```
REXC
220 193 3 45  /* FOR \ A return single character
END
```

The two characters \ A are substituted for the single specified character which is returned.

All of the following sequences can be defined at the same time:

```
REXC
155 193 0 2   /* transfer ESC A and next 2 as read
155 194 1 187 /* transfer ESC B until (and including);
              /* as read
155 195 2 59  /* transfer ESC C until (and including);
              /* as read
220 193 3 45  /* for \ A return single character
```

Read_map

Defines a table of one-to-one character substitutions. By default, input processing passes characters through as read. Read_map is used to define a one-to-one substitution. The read_map keyword is followed by pairs of numbers. The first number is the value of a character read and the second number is the character value to return for further processing.

Any characters that should be passed straight through do not need to be in the list. For example:

```
RMAP
193  194      /* read A return B
194  193      /* read B return A
220  155      /* map \ to the read escape character
```

This RMAP passes all characters straight through, except A and B are swapped, and \ attempts special processing.

Space_compress_off

Disables the expansion of a space compression character. The input routine expands the character DC1 (142) into a number of spaces (character 160 SPAC). The number of spaces inserted is determined by the character following DC1 (ignoring any NUL 0). Space_compress_off prevents this processing, and causes DC1 to be treated as other characters.

Summary of Keywords for Output Mapping

Keyword	*Abbreviation*	*Description*
spacing_char	SPAC	Moves the print position one character to the right.
write_alternate_map	WALTMAP	Defines the alternate write map.
write_escape	WESC	Defines the value to be tested to see if special processing is required during write mapping.
write_exception	WEXC	Defines a table of characters that require special processing.
write_map	WMAP	Defines the basic output map of one-to-one substitution.
END	END	WALTMAP, WEXC, and WMAP introduce a list of values that must be terminated by END.

Output Mapping Keywords

This section describes the keywords used for output mapping, with examples.

Spacing_char

Defines the character that advances the print position one character.

When overlaying multiple lines, unmapped characters are replaced with the spacing_char. For example:

```
SPAC 9  /* Move print position 1 space to the right.
```

In this example, 9 is defined as the spacing character. When the printer encounters a 9 (or an I in ASCII), it advances the print position one space to the right. The default spacing character is 32 (the space character). This applies to lines printed as part of type 1 mappings. It does not affect other processing such as overprinting in FORTRAN mode (using + at the start of a line).

Write_alternate_map

Defines the alternate write map. Characters that are not given in the write map are not substituted. An example follows.

```
WALTMAP
 "   "
193 67  /* When in an escape sequence map A to C
194 66  /* When in an escape sequence map B to B
 "   "
END
```

All other characters pass through unchanged.

Write_escape

Defines the value that tests for special processing. If a single character or the first character of a pair of characters requires special processing, it must be mapped to the write_escape character. The default is WESC 27 (in ASCII this is ESC escape). For example:

```
WESC 164  /* Make & the write_escape character
```

Therefore, to treat * as an exception:

```
WMAP
170 164  /* To construct *, first map it to escape
END
```

Write_exception

Defines a table of characters that require special processing. Once the output text is mapped using the write_map keyword, it is checked for characters that require special processing. Characters for special processing are detected by checking for the write_escape character. When the write_escape character is found, the character in the corresponding position of the text to be mapped is used (and sometimes the following character) to search a list that defines the special processing required.

All lines that define a write_exception have the format:

```
orig type <other values>
```

- **orig** is the character from the despooler that was mapped to the write_escape.
- **type** indicates the type of processing to perform. The type of write_exception is defined after **orig** because some types of write exception are defined for a single character (see types 0 and 1).

If the original character requiring special processing is not found, then the write_escape character is output. This is the opposite of the read_exception processing.

Possible types are

- Output *n* characters in place of the one character (type 0), for example:

  ```
  170 0 120 8 43  /* Instead of * output x <backspace> +
  ```

  Instead of outputting the original character, output the list of characters. You can have 0 through 32 character values after the 0. This type is used primarily to construct a single character by overprinting several characters, but you can also use it to construct any output sequence, as long as the printer has the capability. For example, you could map one control character to turn underscore on, and map another control character to turn underscore off.

- Overprint *n* characters in place of one character (type 1). For example:

  ```
  170 1 120 43  /* Instead of * output x and +
  ```

  Instead of the original character, output the following characters, one at a time, and one on top of the other. You can have 0 through 32 character values after the 1. The first character is printed on the same line as other mapped characters. The second and subsequent characters are printed on new lines and each new line is printed top of the previous lines. All characters on the second and subsequent lines that are not mapped with this type are replaced with the spacing_char (described later in this section). This type may not work correctly if the replaced characters do not have the same print width as the spacing_char. For any single character, you can define either a type 0 or a type 1.

- Fixed length sequence which is not to be mapped (type 8). For example:

  ```
  152 8 220 2  /* CAN \ and next 2 are not to be mapped
  ```

  If the output text contains sequences that should be sent to the printer unchanged, do not perform the one to one substitution. If the sequence to be printed unchanged has a fixed length, then use this type. The original character values and the following *n* character values (0 through 255) are printed, not the mapped values.

  In this example and the examples that follow, CAN is the double underline control character for the QUME printer.

- Fixed length sequence which is to be mapped differently (type 9). For example:

  ```
  152 9 220 2  /* CAN \ and next 2 are to use the
               /* alternate map
  ```

Some character sequences may produce the wrong result if they are passed through the usual substitution. Type 8 allows these sequences to pass through unchanged. However, some of these sequences may still require alteration. Type 9 maps these sequences using the alternate write map (see the description for write_alternate_map earlier in this section). For example, the character [(219) may need mapping to occur in the escape sequence CAN\[B.

- Variable length sequence which is not to be mapped (type 16). For example:

```
152 16 220 187   /* CAN \ and following until ; are not
                 /* to be mapped
```

  The only difference between type 16 and type 8 is that the sequence in type 16 is not a fixed length. The sequence is terminated by detecting a particular character value.

- Variable length sequence which is to be mapped differently (type 17). For example:

```
152 17 220 187   /* CAN \ and following until ; are
                 /* to use the alternate map
```

  The only difference between type 17 and type 9 is that the sequence is not a fixed length. The sequence is terminated by detecting a particular character value.

- Variable length sequence which is not to be mapped (type 20). For example:

```
152 20 220 59   /* CAN \ and the following until ; are
                /* not to be mapped
```

  The only difference between type 20 and type 16 is that type 20 ignores the high order bit when testing for the terminating character, which means that the sequence terminates with either 59 (00111011 in binary) or 187 (10111011 in binary).

Write_map

Defines the output map. The write_map keyword is followed by pairs of values, with each pair on a separate line. Each pair of values consists of the character value sent to the printer and the character value that is printed.

Assuming that ECS is the input character set and ASCII is the output character set, all characters *not* included in the write_map are mapped. Characters in the range 0 through 127 are mapped to 128 through 255, and characters in the range 128 through 255 are mapped to 0 through 127. For example:

```
WMAP
195 68  /* Instead of C output D
196 67  /* Instead of D output C
```

This example swaps the characters C and D.

Mapping Examples

Example 1:

Suppose a printer has an escape sequence that sets the line spacing in 216ths of an inch. The format of this command is ESC 3 *n*. In ASCII this is 27 51 *n*, where *n* is 1 through 255.

When the line spacing feature is used, this sequence must not be altered by the Spooler subsystem. This sequence must be passed through both the read mapping and the write mapping routines unchanged. To achieve this, the following map may be used:

```
REXC
27 51 0 1          /* Pass ESC 3 and next character
                   /* straight through the read routine.
END

WEXC
27 8 51 1          /* Pass ESC 3 and next character
                   /* straight through the output mapping
                   /* routine.
END
```

Example 2:

Suppose a printer has an eight bit character set. The printer prints the o circumflex character when it receives the character 148. This corresponds to 86 in ECS. Therefore to print the character o circumflex (a lowercase o with a ^ above it), a write map must be defined that maps 86 to 148. For example:

```
WMAP
86 148   /* Make ECS o circumflex output the correct
         /* symbol.
END
```

Another printer may only have a limited number of characters that do not include the symbol o circumflex (which is a lowercase o with a ^ directly above it). To simulate the o circumflex, print the o and overprint it with a double quotation mark (").

This can be done using the following sequences:

```
WESC 255 /* Define the write escape character as
         /* something we never want to output to
         /* the printer.


WMAP
86 255   /* Map the ECS character for o circumflex to the
         /* write escape.
END


WEXC
86 0 111 8 34      /* Instead of one character for
                   /* o circumflex output o backspace ".
END
```

Default Mapping

A default mapping is used for each printer unless you request a different one with the SPOOL command –XLATE option. If you need a different default mapping, you can create a source file, run CREATE_MAP, and save the map with the filename STANDARD_MAP in the MAPPINGS subdirectory for this purpose.

The STANDARD_MAP file is not created as part of the Spooler subsystem installation for improved performance. Do not create a STANDARD_MAP file unless you need to replace the default mapping.

The online file DEFINING_MAPS.RUNO in the directory SPOOL*>INFO also contains an explanation of character mapping.

Note When plotting a file, either using the SPOOL –PLOT option or by embedding the character sequence ^001^005 in the file, no mapping or formatting will take place. Mapping can degrade performance. At Rev. 21.0, mapping is performed by reading characters using an ECS file with space compression and ASCII output. If you use QUME printers that require OAS support, replace the ECS_TO_ASCII map with the QUME_OAS_MAP. Performance also depends upon the complexity of the map itself.

Electronic Vertical Format Unit

Some printers use **Electronic Vertical Format Unit (EVFU)** to print invoices, payroll checks, and other forms on which a few items of information must be placed accurately on particular lines. The files used to print information on forms must include special control codes to operate the EVFU.

When you create an environment file for a printer that uses an EVFU, you must perform the following three tasks:

- Create an EVFU file.
- Use the –EVFU option of the DEVICE directive in the printer environment files to give the pathname of the EVFU file.
- Ensure that the FORMAT directive parameters in the environment file are consistent with the EVFU file.

Note If a printer using an EVFU either loses power or is switched off, the paper must be realigned. To realign the paper, first stop the despooler phantom that controls the printer, and then restart it to reload the EVFU memory (unless you are spooling over TCP/IP). If you are spooling over TCP/IP, it is not necessary to stop and start the printer.

EVFU Files

An EVFU file uses **channel numbers** to designate physical lines on a form. All EVFU files must conform to the following rules:

- Each line of the form is associated with only one channel.
- Channel 1 must be line 1 of the form.
- A maximum of 12 channels can be designated.
- Only channel 12 can be assigned to more than one line.
- The maximum form length for a 300-lpm printer/plotter is 132 lines.
- The maximum form length for a band printer is 143 lines.
- The EVFU file must reside in the SPOOL* directory.

EVFU files are ASCII files that you create with an editor such as ED or EMACS.

You can create an EVFU file in either the new format or the old format. Both formats are described below.

New Format EVFU Files: Using the new format, you define channel numbers and their associated line numbers explicitly. The example below includes four channels. Channel 1 is associated with line 1 of the form (and must be line 1 or the form will not work), channel 2 with line 5, channel 4 with line 14, and channel 10 with line 40.

```
Channel 1 Line 1

Channel 2 Line 5

Channel 4 Line 14

Channel 10 Line 40
```

Because it is easier to construct and correct EVFU files using the new format, it is recommended that you create any new EVFU files using the new format (not the old format).

Old Format EVFU Files: Old format EVFU files are graphic representations of the forms or page layouts that they produce. Be sure that the number of lines in an old format EVFU file is equal to the number of lines in the form.

Using the old format EVFU files, you enter channel numbers on the physical lines of the form that they control. This format is difficult to read and maintain because files typically contain a large number of blank lines. For example, if you want channel 5 to represent line 20, you must enter channel 5 on line 20 of the form.

- The channel number must be the first nonblank character on the line.
- The first line of the form must be channel 1, which means that you must enter channel 1 on the first line of the EVFU file.
- Lines without channels assigned to them can contain a 0, or can be left blank.

You can enter comments on any line that contains a channel number if you wish.

The following example shows how to use ED to create and file an old format EVFU file. This example contains the same information in the example of the new format EVFU file shown previously.

```
OK, ED
INPUT
1                  /* This is line 1 of the form
.
.
2                  /* This is line 5 of the form








4                  /* This is line 14 of the form
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
.
10                 /* This is line 40 of the form
EDIT
FILE PAYROLL-FORMAT
OK,
```

How to Use an EVFU File

When you use EVFU files, you embed control characters in the text file that will be formatted by the EVFU file. The control characters are called **skip-to-channel** instructions. When the printer receives a skip-to-channel instruction, it skips to the line associated with that channel number in the EVFU file.

This example shows a file that can be formatted by either of the EVFU files shown previously:

```
^001^000                          /*Enter NO_FORMAT print mode
^003^001 Top of page is here      /*Channel 1
^003^002 This is Line 5.........  /*Channel 2
 Line 6 information is here......./*More output file lines
 Line 7.............              /* are put in here
^003^004 This is Line 14.......   /*Channel 4
 Line 15............              /*Another output file line
^003^010 This is Line 40......... /*Channel 10 in octal
 Line 41............              /*Another output file line
```

Comment lines begin with the /* marker. Comments would not ordinarily be present but have been included in the above example to clarify what is in the file.

The control character ^003, which must be in column 1, tells the despooler phantom to look at the next character, which gives the number (in octal) of the channel to skip to. In this case, the phantom is instructed to skip to channel 1 (top of page), channel 2 (line 5), channel 4 (line 14), and channel 10 (line 40). When the file is spooled, the normal despooler pagination format control must be disabled. Formatting is disabled by one of the following two methods:

- Using the –NO_FORMAT option of the SPOOL command
- Placing the control characters ^001^000 on the first line

Embedded Control Codes in Printable Files

The following list contains the control codes that you can embed in a file, and describes how the Spooler interprets their function. Control codes must start in column 1 unless otherwise stated. Each control code occupies a line in the file. Control codes do not print anything after the code sequence unless otherwise specified. Channel (both vertical and horizontal) commands apply to the logical page.

Mode Change Commands

Control Characters	*Translation*
^000^001 *header*	Sets page header to *header*, enters Paginate mode, and reinitializes page layout parameters.
^001^000 ^201^000	Sets No Format mode (–NOF), which passes the text directly to the printer without interpretation.
^001^001 ^201^001	Sets FORTRAN Forms mode (–FTN). Spooler now interprets: 1 + 0 – Standard FORTRAN codes that appear in column 1.
^001^002 ^201^002	Enters COBOL Forms mode. Identical to FORTRAN Forms mode except the first two characters (format characters) are used. The first character is the same as in FORTRAN format, the second character is always a space.
^001^003 ^201^003	Enters Paginate mode (driver default). When in this mode, the device driver interprets all occurrences of ^214 as a form feed. A page eject is performed automatically when the bottom margin is reached. A page heading is printed at the top of each page provided that the top margin is greater than 1.
^001^004 ^201^004	Enters No Header mode. Same as Paginate mode except that no page headings are printed.
^001^005 *nnn* ^201^005	Sets Raster Plot mode. *nnn* is the number of words to pass directly to the printer without intepretation. If *plot* is unspecified, then plots to end of file.
^001^006 ^201^006	Sets Vector format.
^001^036 *header* ^201^036 *header*	Sets page header to *header*, starts a new page, and does not reset page layour parameters.
^003^*nnn*	Skips to EVFU channel *nnn* which must be in the range 1 through 12. If *nnn* is not given, then the channel number defaults to 1, which is a Top-Of-Form and advance top margin.

Control Characters	*Translation*
^001^037^*nnn*^*nnn* ^201^037^*nnn*^*nnn*	Sets new page size specified by the 16-bit number contained in the two *nnn* bytes. Obsolete option, maps into set bottom margin.
^001 *header* ^201 *header*	Sets page header, enters Paginate mode, and reinitializes page layout parameters.

Logical Page Formatting Commands

Control Characters	*Translation*
^002^000SET_PORTRAIT	Causes a new page to be started in portrait orientation.
^002^000SET_LANDSCAPE	Causes a new page to be started in landscape orientation.
^002^000SET_PAPER_BIN *n*	Causes the paper bin to be changed on the next feed. *n* can be from 1–9 or MANUAL.
^002^000SET_FONT *fontname*	Sets the font to *fontname* starting with the next line. *fontname* must be from the list supported in the printer's .ENV file, so it works like the –SET_FONT SPOOL option.
^002^001 *nnn*	Sets left margin (for example, ^002^00110 sets LM to 10). *nnn* is the column number of the left margin in ASCII.
^002^002 *nnn*	Sets right margin. *nnn* is the column number of the right margin in ASCII.
^002^003 *nnn*	Sets top margin. *nnn* is the line number of the top margin in ASCII.
^002^006	Sets Truncate Long Lines mode (clears Wrap Long Lines mode). Lines are truncated – data is lost.

Control Characters	*Translation*
^002^004 *nnn*	Sets bottom margin. *nnn* is the line number of the bottom margin in ASCII. Attempting to print beyond this causes a page eject.
^002^005	Sets Wrap Long Lines mode. Lines exceeding the number of columns between left and right margins are wrapped to the next line.
^002^006	Sets Truncate Long Lines mode (clears Wrap Long Lines mode). Lines are truncated – data is lost.

Form Formatting Commands (Reserved)

Control Characters	*Translation*
^002^010 *nnn*	Sets characters per inch. *nnn* is the number of lines per inch in ASCII. The physical page is also changed to accommodate lines per inch. This is *not* used by the PostScript driver.
^002^011 *nnn*	Sets lines per inch. *nnn* is the number of characters per inch in ASCII. This is *not* used by the PostScript driver.
^002^012 *nnn*	Sets the number of lines per page. *nnn* is the number of lines in ASCII.
^002^013 *nnn*	Sets characters per page. *nnn* is the number of characters across in columns in ASCII.

Horizontal Movement Commands

Control Characters	*Translation*
^002^020 *nnn*	Sets the tab interval, where *nnn* is the tab interval in ASCII.
^002^021 *str*	Sets tab stops to nonspace characters specified by *str*. For example, if *str* is " ! !", tab stops are set in columns 4 and 8.

Control Characters	*Translation*
^002^022	Clears all tab stops.
^002^023^*nnn*^*mmm*	Sets channel skips, where *nnn* is the channel number and *mmm* is the line to skip to, both in ASCII.
^002^024^	Clears channel. Sets channel 1 to line 1 and clears all other channel settings.

Other Controls

Control Characters	*Translation*
^002^030	Looks for boldface commands on this line.
^201	Bold overstrike on, this may appear anywhere on the line.
^202 or <CR>	Bold overstrike off, this may appear anywhere on the line.
^002^031	Looks for underscore commands on this line.
^201	Underscore on. This may appear anywhere on the line.
^202 or <CR>	Underscore off. This may appear anywhere on the line.
^002^032	Converts lowercase to uppercase before printing.
^002^033	Prints lowercase (clears uppercase).
^002^034	Prints line numbers in the left margin.
^002^035	Does not print line numbers.
^002^036^*nnn*<*text*>	Skips *nnn* lines and prints *text* which is an ASCII text string.
^211 <HT>	When embedded in Paginate mode, tabs to next tab stop.
^215 <CR>	End of line terminator. Sends a <CR> to the device when in Paginate mode.

Control Characters	*Translation*
^212 <LF>	End of line terminator. Sends a <CR><LF> (carriage return and linefeed). The device is in Paginate mode.
^213 <VT>	Skips to next EVFU channel, or 4 lines if not set.
^214 <FF>	A form feed and skip top margin is done when embedded in Paginate mode.
^215 <CR>	End of line terminator. Sends a <CR> to the device when in Paginate mode.

Control code information can also be found in the *Subroutines Reference IV: Libraries and I/O*.

Tailoring Spooler Information

5

Introduction

This chapter describes changes that you can make to Spooler subsystem features to suit your informational needs, as follows:

- Creating a custom spool list display
- Expanding log file information
- Creating a file of user address information
- Changing header page information
- Using multilanguage despoolers

Creating a Custom Spool List Display

You can create your own format for displaying spool queue entries with the SPOOL –LIST command. You can create a custom spool queue display format using one of the following two methods:

- Using the Programmable Interface, which allows you to replace the existing routine that displays spool queue entries with a new SP$LST routine that you create.

  If you create a new routine with the Programmable Interface, the spool list display changes for all of the SPOOL –LIST options: –BRIEF, –DETAIL, and –FULL.

- Using the Configurable Listing Interface, which allows you to create an ASCII file containing a custom spool list display format that you process through the CREATE_CONFIG_LIST program located in SPOOL*.

  If you create a new spool list display using the Configurable Listing Interface, the spool list display changes only when using the –BRIEF option of the SPOOL –LIST command (the default).

Instructions for creating your own spool list displays are described in the following sections.

Using the Programmable Interface

The existing routine that displays each spool queue entry is now an entrypoint called SP$LST in the SPOOL command library SP$LIB.

The existing routine generates one of the three types of displays, –BRIEF, –DETAIL, or –FULL, for each spool queue entry, and verifies user names, attribute names, and so on. If your search rules contain an earlier version of SP$LST, you can now replace it with your own version of SP$LST. Replacing the entrypoint SP$LST does not compromise Spooler security.

The entrypoint takes arguments of the queue entry to be displayed, and the REPORT_OPTIONS structure. Templates of the queue entry and REPORT_OPTIONS structure are provided in insert files named REP_OPTIONS in SPOOL*>SOURCE. The insert files are in languages PL1, FTN, and C.

The calling sequence is shown below:

```
SP$LST(QE, REPORT_OPTIONS)
```

QE is the entry structure holding the details of an individual request.

REPORT_OPTIONS is a structure holding the options used for scanning and displaying the queue information.

The routine is called once for each request displayed. If there are 10 requests returned from the spool queue, then the SP$LST routine is called 10 times. The SP$LST routine checks for valid user names, matching attributes, style of reports (–BRIEF, –FULL, –DETAIL), and so on.

The arguments to SP$LST are both structures, which are defined in the following insert files located in the SPOOL*>SOURCE directory:

- REP_OPTIONS.INS.PLP for PLP
- REP_OPTIONS.H.INS.CC for C
- REP_OPTIONS.INS.FTN for FORTRAN

The REP_OPTIONS.ENTRY_FOUND flag is the only argument that you should set as true. If you do not set this flag when the Spooler calling routine finishes scanning the queue, the message `no entries found` is displayed. The SP$LST routine changes all three types of spool displays, –BRIEF, the default, –FULL, and –DETAIL.

Using the Configurable Listing Interface

The Configurable Listing Interface provides an easy method for creating custom spool list displays for individual users, project users, and for all users on a system.

To create a custom spool display, you create an ASCII file containing a header section and a selection section. You use the header section to enter column headings for the display, and the selection section to select the spool queue information displayed for each heading. Creating a custom spool list display using this method is similar to creating a report from a database. You choose the wording you want for the column headings, and select specific fields of information from the spool queue that you want displayed for each heading.

After creating the ASCII file, run a special program, CREATE_CONFIG_LIST, located in SPOOL*, that tests your ASCII file for errors and creates a binary file that generates the actual display when you issue the SPOOL –LIST command. The binary file must reside in the SPOOL*>CONFIG_LIST subdirectory.

The ASCII input file contains components called markers and keywords. Markers begin and end the header and selection sections of the ASCII file. Keywords are field names that you enter to select the spool queue information you want displayed. For example, file_name_32 selects the filename of the spool request and displays up to 32 characters of the filename. For each keyword, you can choose to display a set number of characters, or display a range of characters of information for that field.

The markers and keywords are listed and described in the following list.

Marker	*Description*
start_header	Begins the header section containing column headings that you enter for the spool list display.
end_header	Ends the header section.
start_selection	Begins the selection section that you use to enter keywords that select the spool queue information displayed.
end_selection	Ends the selection section.

Keyword	*No. of Characters Displayed*	*Spool Queue Information Displayed*
attributes_16	16	Attributes of the spool request.
attributes_32	32	
attributes_64	64	
attributes_128	128	
attributes_var	Range from 2 to 128	
blank_var *n*	Range from 1 to 32	Add space between display fields to align information under column headings.
copies	2	Number of copies requested.
defer_time	5	Spool request deferred printing time.
env_name_8	8	Name of the printer environment processing the request.
env_name_16	16	
env_name_var	Range from 2 to 16	
env_node_6	6	Nodename on which the despooler phantom processing the request resides.
env_node_8	8	
env_node_16	16	
env_node_32	32	
env_node_var	Range from 2 to 32	
file_name_16	16	Name of the file being processed.
file_name_32	32	
file_name_var	Range from 2 to 32	
file_size	5	Record size of the file being processed.
map_name_6	6	Character map name given with the spool request.
map_name_8	8	
map_name_16	16	
map_name_32	32	
map_name_var	Range from 2 to 32	
origin_node_6	6	Node from which the spool request was issued.
origin_node_8	8	
origin_node_16	16	
origin_node_32	32	
origin_node_var	Range from 2 to 32	

Keyword	*No. of Characters Displayed*	*Spool Queue Information Displayed*
options_16 **options_32** **options_64** **options_128** **options_var**	16 32 64 128 Range from 2 to 128	SPOOL command options given with the request.
page_header_40 **page_header_80** **page_header_160** **page_header_var**	40 80 160 Range from 2 to 160	Page header text (unless the –NPH option is given).
partition_name_6 **partition_name_8** **partition_name_16** **partition_name_32** **partition_name_var**	6 8 16 32 Range from 2 to 32	Partition name of the SPOOL_DATA* directory containing the spooled file.
path_name_40 **path_name_64** **path_name_128** **path_name_160** **path_name_var**	40 64 128 160 Range from 2 to 160	Pathname of the spooled file.
percent_printed	2	Percentage of the total file that has finished printing.
ps_proc_6 **ps_proc_8** **ps_proc_16** **ps_proc_32** **ps_proc_var**	6 8 16 32 Range from 2 to 32	PostScript procedure name given with the spool request.
raster_size	5	Buffer size set for plotting a particular job.
request_number	5	Spool request number.
request_time	5	Time at which the spool request was entered into the queue.
state	5	Status of the spool request, such as whether it is printing.

Keyword	*No. of Characters Displayed*	*Spool Queue Information Displayed*
user_address_16 user_address_32 user_address_var	16 32 Range from 2 to 32	Distribution address of the user who spooled the file, as listed in the USER_ADDRESSES file.
user_name_16 user_name_32 user_name_var	16 32 Range from 2 to 32	Login name of the user who spooled the request.
user_no	3	User number from which the request was spooled.

The following example shows a sample ASCII input file for generating a custom spool display for an individual user whose user ID is OPS. In this example, the user OPS wants only to see how many jobs are in the spool queue, the request number and size of the job, and the name of the printer processing each request.

```
/* ascii input template file for print room operators
/* input template filename is ops.template
/* First you enter the header section.
start_header
/* Now enter your column heading names (use the spacebar to
/*   enter any needed space between headings).
Reqno    Size    Copies    Attributes
/* End the header section when you have finished entering
/* the column headings you want.
===========================================================
end_header
/* Now begin the "required" selection section.
start_selection
/* Enter only the keywords that select the spool queue information
/*    you want displayed underneath each column heading.
Request_number blank_var 2 File_size blank_var 2 Copies blank_var
7 Attributes_var 78
/* Now end the selection section.
end_selection
```

In the sample ASCII file shown here and shown later in the procedure for creating an ASCII file, the keyword line is shown on two or more physical lines in order to fit the example on the page. In order for the spool queue information to display underneath the column headings and across the screen in one row (similar to report columns), the keyword line must not wrap to the next physical line in the ASCII input file. If the keyword line wraps to the next line in the file, the information selected by those keywords displays on a second line, underneath the first line of information.

The following example shows the custom display produced by the sample ASCII input file OPS.TEMPLATE when user OPS issues SPOOL –LIST:

```
OK, SPOOL LIST
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
System SYSA
Reqno     Size     Copies     Attributes
===================================================================
10 April 92
1911      2        1           WIDE
22 April 92
2454      3        3          TP_QUM
2455      10       10         TP_QUM
OK,
```

Naming the ASCII Input File

Before you create the ASCII file, you must first decide whether you are creating a custom display for an individual user or a project ID, or a custom default spool list display for all users on the system. Although you can give the ASCII file any name you want, the binary output filename must be the user ID or project ID, prefaced with an asterisk (*).

If you are creating a new default spool list display for all users on the system, you must name the file .DEFAULT_CONFIG_LIST. The periods at the beginning and the end of the filename are part of the name.

Creating an ASCII File for a Custom Spool Display

This section explains how to create an ASCII input file for a custom spool list display, and explains how to run the CREATE_CONFIG_LIST program to test your input file and generate the actual spool list display.

Before you begin, it may be helpful to first decide the keywords and display width of the spool queue information you want displayed. You need to determine the display width of the keyword fields to figure the space between each column heading.

After you run the ASCII file through the CREATE_CONFIG_LIST program, you may need to adjust the spacing in the template file to get the headings and queue information to align properly.

1. Create an ASCII file using a text editor such as ED or EMACS. You can refer to the sample template called SAMPLE.TEMPLATE shown at the end of these instructions.

2. Enter a comment line indicating the name of your template file, and any other information you may want to include to describe the purpose for the custom spool display.
3. Begin the header section by entering the marker start_header on a separate line.
4. Starting on a new line, enter your column heading names. Use the Spacebar to add space between headings.

   You can arrange the column headings in one line across the screen or in a column. However, if you have long spool list displays, arranging the headings across in a row would allow more display information per page than if you arranged the headings in a column which would display each field of information to the right of the heading. You can arrange the headings in any order that you want. For example, you can have the filename display before the request number, and so on.

   In the sample SAMPLE.TEMPLATE, the column headings are arranged in a row. The sample template also contains a row of dashes underscoring each column heading, which is optional. You can use any character you want for an underscore, or choose not to underscore the headings at all.

   Be sure the column heading names are meaningful to the user, the project group, or all of the users who will use this display.

   You may want to abbreviate a column heading if the display width of the spool queue information is only five characters to allow space for other columns, if you need it.
5. Enter the end_header marker on a separate line.

   You can enter comment lines anywhere in the file, if you need them.
6. Begin the selection section by entering the start_selection marker on a separate line.
7. Starting on a new line, enter the keywords for the information you want displayed under each column heading. You cannot enter any other text on the keyword line. Only the keywords themselves. Each keyword automatically adds one space to the display width.

   If the keyword displays five characters, the display width of that field is actually six characters including the space.

   Use the blank_var *n* keyword between keyword fields to add whatever space you need to align the spool queue information under the correct heading.

   For example, suppose the column heading is Req.No., which is seven characters long, and there is one space between Req.No. and the next heading (a total of eight characters from the start of the Req.No. heading and the next column heading). The keyword request_number displays only five characters plus one space. Since the total display width is shorter than the column heading spacing of eight characters, you need to add two spaces so that the next column of information aligns properly under the column heading.

8. Enter the marker end_selection on a separate line. Save the ASCII file.

   The sample ASCII template file SAMPLE.TEMPLATE is shown below:

```
OK, SLIST SAMPLE.TEMPLATE
/* Sample.template
/* Sample ascii file for creating a custom SPOOL -LIST display
/* Copyright (c) 1992, Prime Computer, Inc., Framingham, MA 01701
/*
/* The following line is an optional Header Section
start_header
User Name  Filename         Req.No. Size   Copies    System Status
---------  --------         ------  ----   ------    ------ ------
end_header
/* The selection section is required to select spool queue
information.
start_selection
user_name_var 10 file_name_16 Request_number blank_var 2 file_size
blank_var 2
Copies blank_var 6 origin_node_6 state blank_var 5 defer_time
blank_var 5 attributes_128
end_selection
OK,
```

9. Run the CREATE_CONFIG_LIST program using the following format:

   RUN SPOOL*>CREATE_CONFIG_LIST -INPUT *pathname* -OUTPUT *pathname*

   The output pathname is the binary filename that must be the individual user ID, project ID (prefaced with an asterisk *), or named .DEFAULT_CONFIG_LIST., and it must reside in the subdirectory CONFIG_LIST below SPOOL*. The CREATE_CONFIG_LIST program is used in the following example to generate the binary file SAMPLE from the ASCII input file called SAMPLE.TEMPLATE.

   RUN SPOOL*>CREATE_CONFIG_LIST -I SAMPLE.TEMPLATE -O SPOOL*>CONFIG_LIST>SAMPLE

   The CREATE_CONFIG_LIST program displays the template contents on your screen, and displays any errors found in processing the template file. For example, an error message displays if the template is missing the selection section, or missing a beginning or ending marker.

10. Assuming there are no errors in the ASCII file, and the binary file was written successfully to the CONFIG_LIST directory, you can now test your

custom spool list display by entering SPOOL -LIST. The next example shows the custom spool list display for SAMPLE.TEMPLATE.

Notice that the spool request number does not line up perfectly under the Req.No. heading. You can easily adjust the alignment by editing the template file and adding more space using the blank_var *n* keyword.

```
OK, SPOOL -LIST
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
System ENB
User Name  Filename         Req.No. Size   Copies    System Status
---------  --------         ------  ----   ------    ------ ------
13 February 92
RHONDA     SAMPLE.TEMPLATE  20       2       2        ENB    Defer
     15:00
     DEFER, CARBON
14 February 92
BSB        MAIL.SPOOL.TEMP  107      2       1        ENB
     QMST, 2
OK,
```

Each time you edit the template file, run the CREATE_CONFIG_LIST program again to incorporate the changes.

Expanding Log File Information

Each despooler phantom records a line of basic accounting information in its log file after completing each print job, provided that the log file is active. If the basic despooler accounting information is not adequate for your needs, an experienced programmer can use the Accounting Interface to create an accounting routine to your exact requirements.

The routine that you create must be a library EPF routine with the entrypoint SP$ACT. You must also modify the search rules in SPOOL*>ENTRY$.SR so that the new EPF can be found.

Data Structure Information

The call to SP$ACT takes two arguments. The first argument is a data structure that matches the PL1 declaration shown in Figure 5-1. All character fields are a fixed length to make the structure compatible with FTN. The following Insert files are supplied in SPOOL*>SOURCE for PL1 and FTN:

- ACCOUNTING.INS.PL1
- ACCOUNTING.INS.FTN

The second argument to SP$ACT is a group of text fields that define information for printing on the trailer page, if a trailer page is used. Each field is a fixed-length string of 80 characters. If the open_option_used bit is set, then the filename and dtm fields are not significant.

```
dcl 1 accounting_data,
2 version bin,
  2 environment_name char(16),            /* Spooler environment name */
  2 origin_node char(32),                 /* Requesting user's node name */
  2 user_name char(32)                    /* Requesting user's login name */
  2 user_no bin,                          /* Requesting user's user number */
  2 project_id char(32),                  /* Requesting user's login project */
  2 request_time char(12),                /* Time of request, yymmddhhmmss */
  2 file_name char(160),                  /* Full path name of file to print */
  2 file_size bin,                        /* Size in records * no of copies */
  2 no_of_copies bin,                     /* No of copies requested */
  2 dtm bin(31),                          /* File system format */
  2 no_of_attributes bin,                 /* No of names following */
  2 attribute(32) char(16),               /* Attribute names from request */
  2 options,
    3 no_copy bit(1),                     /* -NOCOPY option used */
    3 numbers bit(1),                     /* Generate line numbers */
    3 no_header bit(1),                   /* No header page */
    3 no_final_eject bit(1),              /* Suppress final page eject */
    3 no_format bit(1),                   /* No format (-NOFMT) */
    3 fortran_format bit(1),              /* Fortran format codes */
    3 plot_file bit(1),                   /* Plot file */
    3 notify_user bit(1),                 /* Notify user on completion */
    3 truncate bit(1),                    /* Truncate long lines */
    3 cobol_format bit(1),                /* -COBOL */
    3 no_overprint bit(1),                /* -NOP */
    3 open_option_used bit(1),            /* Set if -OPEN or SPOOL$ equiv */
    3 no_header_format bit(1),            /* -NPH */
    3 suppress_file_info bit(1),          /* -SFI */
    3 from_page bit(1),                   /* -FROM */
    3 to_page bit(1),                     /* -TO */
  2 options2,
    3 swo bit(1),                         /* -SPOOL_WHILE_OPEN */
    3 xlate bit(1),                       /* -XLATE */
    3 rfu bit(14),                        /* RFU as spare options */
  2 raster_size bin,                      /* Words per raster in plot file */
  2 start_print_time char(12),            /* Print started at yymmddhhmmss */
  2 end_print_time char(12),              /* Print completed at yymmddhhmmss */
  2 terminating_status bin,               /* See below */
  2 no_of_backups bin,                    /* No of times print backed up */
  2 no_of_restarts bin,                   /* No of print restarts */
  2 total_pages bin(31),                  /* Total pages including restarts */
  2 total_lines bin(31),                  /* Total lines including restarts */
  2 total_characters bin(31),             /* Total chars including restarts */
  2 from_page_num bin(31),                /* from page number */
  2 to_page_num bin(31),                  /* to page number */
  2 map_name char(32);                    /* name of map */
%replace accounting_version_ by 2         /* Value in VERSION above */
%replace SUCCESS_ by 0,                   /* Terminating status values */
    ABORTED_ by 1,
    DROPPED_ by 2;
```

Figure 5-1. PL1 Data Declaration for SP$ACT

Coding Rules

The SP$ACT accounting routine can perform any actions that do not destroy the despooler program from which it is called. The rules for writing the routine are as follows:

- The accounting routine can attach to a new directory and can safely leave the despooler phantom attached there on return.
- Files opened by the accounting routine must be opened using the K$GETU key, and must be closed before returning to the despooler.
- The accounting routine must not attempt to drive the printer as this can cause undesirable effects when printing continues after returning to the despooler.
- The accounting package can invoke EPF or static mode programs so long as they do not destroy the despooler, which itself is an EPF.
- Any dynamically allocated memory must be released before returning control to the despooler.

Caution Any conditions signaled while in the accounting routine, and not handled by it, cause the despooler to shut down.

Calling the Accounting Routine

When you have coded and tested your accounting routine, use BIND to create a program class library EPF and add it to the search rules in SPOOL*.

When a despooler phantom completes its first print job, it checks for the existence of SP$ACT. If an SP$ACT is found, it is run after each print job. If it is not found at the first attempt, no further attempts are made to call it.

Creating a File of User Address Information

The Spooler subsystem provides a facility that enables you to set up user address information for distributing printed output. Deciding how to distribute printed output to users depends on the setup of your facility, the number and location of printers, and the number of users serviced by each printer.

Setting Up User Address Information

To set up address information for distributing output to users, you create a file called USER_ADDRESSES in the SPOOL* directory. You use this file to enter the user ID and a description of the physical location of each user. The address information that you enter for each user ID is printed on the top-right section of the header page of each printout.

A typical address designating a user's physical location might be a room number, an office name, a mailstop number, or any name that enables your distribution personnel to distribute printouts to the correct location.

Each line in the USER_ADDRESSES file contains a user ID and the address information for that user. The user ID and address information must be separated by at least one space. The user ID is required. You can enter up to 32 characters for the address information.

If you create USER_ADDRESSES files for a network of systems, you need to determine a way to keep each file up to date. Maintaining identical USER_ADDRESSES files on each system might be the simplest way of keeping address information up to date on each system in the network. This method means that the file for one system may contain address information for users who cannot necessarily log in to or spool to that system.

Searching the USER_ADDRESSES File

The default routine supplied with the Spooler subsystem uses a simple sequential method to search the USER_ADDRESSES file for address information. If your USER_ADDRESSES files are exceptionally large, the time it takes to search the file may be unacceptable for your needs.

You can implement another method of searching the file by creating your own search routine as described in the next section.

Using Your Own Address Distribution Routine

Prime supplies the SP$ADDRESS library EPF routine in the SPOOL command library (SP$LIB). The SPOOL command calls SP$ADDRESS when a request is submitted to a Rev. 21.0 or later spool queue, or when a request in a Rev. 21.0 or later spool queue is modified.

The SPOOL command software passes the user ID to the SP$ADDRESS routine, or, if the –ALIAS option was used in the SPOOL command, it passes the alias instead. The SP$ADDRESS routine then checks the file SPOOL*>USER_ADDRESSES, and returns the address information to the SPOOL command software.

You can supply a routine of your own to replace the existing SP$ADDRESS routine. The call to SP$ADDRESS in the SPOOL command is as follows:

```
call sp$address(user_name, user_address)
```

Both *user_name* and *user_address* are 32-character fixed strings. The *user_address* parameter is preset to a string of 32-space characters, so that if SP$ADDRESS has no information to return, it should leave user_address unchanged.

When you have coded and tested your new routine, use BIND to create a program class library EPF. You must then modify the search rules in SEARCH_RULES*>ENTRY$.SR and enter the pathname of your SP$ADDRESS library on the line before the SP$LIB.RUN entry. If you do not modify the system search rules, the SPOOL –LIST command will not display the user addresses.

Changing Header Page Information

The following information is printed on the standard header page:

- User ID
- Request number and node from which the request was made
- User name and distribution address
- User ID or alias in large letters (see –ALIAS option of SPOOL command)
- Filename or replacement name in large letters (see –AS option of SPOOL command)
- Copyright message
- Despooler environment used
- Print request attributes
- SPOOL command options given
- File size in records
- Pathname of the file
- Time the file was last modified
- Time the request was made and time that printing started
- Message specified in the environment file

Using Your Own Header Page Routine

You can replace the routine that produces the standard header page with one of your own. The existing routine is a program class EPF with entrypoint SP$HDG in SPOOL*>DESPOOL_LIBRARY.RUN. The routine that you create must also be a program class EPF, named SP$HDG. You can create your routine at any location in the file system, but you must modify the search rules in SPOOL*>ENTRY$.SR so that your replacement for SP$HDG is found before the supplied version. For convenience, create your replacement routine in SPOOL* and modify the search rules list as follows:

```
spool*>sp$hdg.run
spool*>despool_library.run
-SYSTEM
```

This example assumes that you have created a SP$HDG.RUN in SPOOL*.

Data Structure Information

The header page routine is called by the PRINT_FILE routine of the despooler phantom. PRINT_FILE passes the arguments listed below, which must be declared in your header page routine.

```
call SP$HDG (start_date_time, copyright_line, vcom,
envcom, pagcom, RBI, envname)

start_date_time char(15)
copyright_line char(78) var
qcom structure
envcom structure
pagcom structure
rolling_banner_index bin
environment_name char(16) var
```

You can use the information passed in these data structures but you should not alter any of them. The qcom, envcom, and pagcom structures are explained in more detail in Appendix B, Writing a Print Handler.

Coding Recommendations

The source code of the header page routine that Prime supplies is available in SPOOL*>SOURCE>SP$HDG.PLP. You should use this as a template for your own header page routine. If the existing header page does not meet your needs exactly, you can produce the header that you want with a modified version of the existing routine.

Using Multilanguage Despoolers

Sites that use several languages, through the Software Internationalization Tools (SITs), can have despoolers that run in several languages. These languages are handled through the –ENV_LANGUAGE (–ELANG) option to PROP. This option defines the language used by the environment file. SIT support for the language is needed for –ELANG to work correctly.

The –ELANG option can be used with the –VERIFY and –START options to PROP. For example, to start a French despooler on an English (default language) site, the command format would be

PROP ***environment-name*** **–START –ELANG FRENCH**

The command to validate the despooler environment would be

PROP ***environment-name*** **–VERFIY –ELANG FRENCH**

6 Managing Spooler Operations

Introduction

This chapter explains how to use the SPOOL and PROP commands to carry out many of the tasks that are often required of an Operator. For a summary of the PROP and SPOOL commands and all the available options, see Appendix A, SPOOL and PROP Commands. The *PRIMOS Commands Reference Guide* contains information on the PROP and SPOOL commands for users, and the *Operator's Guide to System Commands* contains a reference section of the PROP and SPOOL commands for Operators and System Administrators.

The operations described in this chapter are as follows:

- Checking the status of printer environments
- Controlling printer environments
- Changing to another printer environment
- Modifying a printer environment
- Viewing spool queue entries
- Canceling requests in the queue
- Modifying spool requests
- Changing the printing order of requests in the spool queue

Checking the Status of Printer Environments

Printer environments are controlled by despooler phantoms. Although you can have more than one printer environment for a printer, only one environment can be active for a printer at any one time, except for TCP/IP printers. The PROP command enables you to check the status of available printer environments, such as whether an environment is started, stopped, or in some other state. To check the status of each available printer environment, issue the PROP –STATUS –ALL command as shown below:

```
OK, PROP -STATUS -ALL
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
REQS             Not Started
SPWT             Idle
WIDE             Idle
OPS              Idle
QMST2            Idle
TP_QUM           Printing (FORUM.SPACE: Page 1, Copy 1 of 1
                           Request: 622)
POM2             Idle
SPTNG            Idle
LABEL            Not Started
TRAIN            Idle
OK,
```

In this example environment, TP_QUM is printing a file. Notice that the `Printing` status includes information about the printing file, such as the filename, request number, the page number that is currently printing, and the number of copies requested. The status `Not Started` indicates that the environment is available, but has not been started by a despooler phantom. The status `Idle` indicates that the environment is started and available, but is not currently printing any files.

The status of a printer environment may be any one of the following:

Status	*Description*
Aborting	Shows that the printing file is aborted (the file remains in the spool queue).
Backing Up	Indicates that the printing file is backing up to the top of the current page and restarting.
Dropping	Indicates that the printing file is being canceled (which deletes the request from the queue).
Hanging	Shows that the environment is hung and waiting for the command to continue.
Idle	Shows that the environment is idle and waiting for spool requests.
Lineup	Indicates that the environment is suspended following a paper alignment check.

Status	*Description*
Not Started	Shows an available environment for a particular printer that is not started.
Printing	Indicates the despooler phantom controlling the named environment is printing a file.
Reset	Indicates that one environment is stopping and a new environment is starting.
Restarting	Shows that a previously suspended environment is restarting.
Sleeping	Indicates a despooler phantom is sleeping while waiting for more information to print a file that was spooled with the –SPOOL_WHILE_OPEN option.
Starting Up	Shows that the environment is starting.
Stopping	Indicates that the environment is stopping.
SWO	Indicates the despooler phantom is printing a file spooled with the –SPOOL_WHILE_OPEN option. It also displays the status message `Spooling while open` *`filename`*`, Page` *`p`*`, Copy` *`a`* `of` *`b`*.

Viewing Printer Environments

You can view the contents of an environment file by issuing the following command:

PROP *environment-name* –DISPLAY

Environment files are located in the SPOOL* directory and contain information that the despooler phantoms use to control a printer and its setup.

PROP displays the status of *environment-name* on the first line (after the copyright line), followed by the environment file contents. *environment-name* is the name of the environment, such as WIDE or LABELS.

If you view environment information using SLIST or a text editor, such as ED or EMACS, you must use the full pathname of the file unless you are attached to the SPOOL* directory. If you view the environment file contents with SLIST, ED, or EMACS while attached to the SPOOL* directory, you must supply the filename of the environment, not the environment name itself. For example, LABELS.ENV instead of LABELS.

The following example shows the use of PROP –DISPLAY to view the contents of the environment name QMST2, which is a printer environment for a QMST laser printer.

```
OK, PROP -DISPLAY QMST2
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
QMST2            Idle
/* QMST2 laser printer for SYSA
/* Name of environment file is SPOOL*>QMST2.ENV
/*
ASYNC -LINE 107 -PROTOCOL TTY -SPEED 9600
ATTRIBUTE QMST2 -MANDATORY   /* QMST2 attribute must be specified.
ATTRIBUTE WHITE
DEVICE PRINTRONIX -EVFU_OFF
FORMAT -WIDTH 80 -BOTTOM_MARGIN 1 -PAD_CHAR 0 /* Added -pad_char re: Rev 23 info file
MESSAGE ReSeTrEsEtReSeT^PY^-^IFX1X1^IJ00010^IT00230^IB10950^IS00404^IL0610^-^PN^-
LOG -ON
PRIORITY_SIZE 20             /* Print files less than 20 records first.
NODE -LOCAL                  /* Scan SYSA's spool queue.
NODE ENR
POLL_RATE -DECAY 30          /* Poll at the fast rate 30 x before slowing.
PLQ                          /* Give priority to jobs in local queue.
WARNING -ON
OK,
```

Controlling Printer Environments

This section explains how to use the PROP command to control despooler phantoms and their associated printer environments. Despooler phantoms enable you to

- Start a printer environment.
- Suspend a printer environment.
- Restart a previously suspended printer environment.
- Stop a printer environment.

You can have up to 128 despooler phantoms running on your system, although the number of phantoms available for the despooler is dependent upon the total number of phantoms available for a particular system.

At Rev. 23.3, the PRIMOS commands STAT USERS and LIST_USERS identify Spooler phantoms associated with printer environments, as described and shown in the next two examples.

STAT USERS displays the environment name in the User No column (column 1), and lists the word *spool* in the Line No column (column 3), as shown below. This display was shortened for the purpose of this example.

```
OK, STAT USERS
                              User No   Line No
User                           (In Decimal)     Devices (AL in Decimal)
SYSTEM                            1       asr     <SYSENB>
PI_SERVER                       306      phant    <USRNET> <SYSENB>
RELAY                           315      phant    <SYSENB>
NFS_PORTMAP                     319      phant    <SYSENB>
NFS_MOUNT                       320      phant    <SYSENB>
NFS_SERVER                      321      phant    <SYSENB>
NFS_PCNFSD                      322      phant    <SYSENB>
NTS_SERVER                      329       ncm     <SYSENB>
TIMER_PROCESS                   330      kernel   <SYSENB>
BUFFER_SERVER                   331      kernel   <SYSENB>
LOGOUT_SERVER                   332      kernel   <SYSENB> (SYSTEM IDLE)
LOGIN_SERVER                    333       LSr     <SYSENB> (3)
DSMSR                           335       DSM     <SYSENB>
DSM_LOGGER                      336       DSM     <SYSENB>
SYSTEM_MANAGER                  337      SMSr     <SYSENB>
ISC_NETWORK_SERVER              338      ISCNsr   <SYSENB>
NAME_SERVER                     339      NameSr   <SYSENB>
QMST2                           347      spool    <SYSENB> AL107
POM2                            349      spool    <SYSENB> AL21
HROA                            351      spool    <SYSENB> AL92
SPTNG                           352      spool    <SYSENB> AL134
TRAIN                           353      spool    <SYSENB> AL135
TNG                             354      spool    <SYSENB> AL146
PF1                             355      spool    <SYSENB> AL151
PLANS                           356      spool    <SYSENB> AL153
UBI_SERVER                      357      kernel   <SYSENB>
DSMASR                          358      phant    <SYSENB>
TP_QUM                          360      spool    <TPUBS8> <SYSENB> AL108
```

LIST_USERS displays the process running on the system in groups by type, and displays the number of processes for each type, as shown in the following example.

```
OK, LIST_USERS

39 users:

  4  LAURA              11  DICK               14  GARFIELD
 17  BOBBY              18  SPOT               20  SNOOPY
184  LISA              185  JANE              186  RHONDA(me)

24 phantoms:

272  DST_SERVER        273  MAIL_SERVER        274  YTSMAN
275  FTP               276  BATCH_SERVICE      277  SYSTEM
278  SYSTEM            279  SYSTEM             281  NOTES
282  RFT_AGENT         283  FTX_AGENT          284  TALK_SERVER
286  TCPIP_MANAGER     287  NM_SERVER          288  TCPFTP_SERVER56
289  MAILER_DAEMON     290  HYPERSESSION       300  PI_SERVER
304  TONYPL            315  RELAY              319  NFS_PORTMAP
320  NFS_MOUNT         321  NFS_SERVER         322  NFS_PCNFSD

One slave:

256  ACE

16 spoolers:

340  ENBPR1            341  WIDE               342  OPS
343  SUDS              344  INFO               345  SPWT
346  BUD               347  QMST2              349  POM2
351  HROA              352  SPTNG              354  TNG
355  PF1               356  PLANS              360  TP_QUM
362  TRAIN

15 servers:

  1  SYSTEM            270  NETMAN             271  RT_SERVER
329  NTS_SERVER        330  TIMER_PROCESS      331  BUFFER_SERVER
332  LOGOUT_SERVER     333  LOGIN_SERVER       335  DSMSR
336  DSM_LOGGER        337  SYSTEM_MANAGER     338  ISC_NETWORK_SERVER
339  NAME_SERVER       357  UBI_SERVER         358  DSMASR
OK,
```

Starting a Printer Environment

Use the following command to start a new environment and to restart an environment that has been stopped:

PROP *environment-name* –START

The example below shows the use of PROP to start the TP_QUM environment:

```
OK, PROP TP_QUM -START
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
OK,
*** TP_QUM (user 359 on SYSA) at 12:08
Despooler for TP_QUM ready.
```

When starting a printer environment, the despooler phantom prints the word *STARTING* in large, outlined letters at the printer (if the WARNING –ON directive is set in the environment file).

Hanging a Printer Environment

Use the following command to hang a printer environment:

PROP *environment-name* –HANG [*options*]

Hanging a printer environment enables you to stop printing temporarily to service the printer, for example, to change paper, align a form, or other such task that you perform while the printer is online.

The –FINISH option suspends the phantom after it finishes printing a file already in progress. The –NOW option suspends the phantom immediately. The print request remains in the spool queue and the file is continued when you restart the phantom. The default is –FINISH.

The –IDLE option suspends the phantom when there is nothing left in the queue to print.

In the following example, the –FINISH option is used with the PROP command to suspend printing activity on LQP after printing the current file (if a file is printing at the time the command is issued):

```
OK, PROP LQP -HANG -FINISH
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
OK,
*** LQP (User 102 on SYSA) at 09:29
Hanging
```

Restarting a Hung Printer Environment

Use the following command to release a hung printer environment and continue printing:

PROP *environment-name* –CONTINUE

The despooler phantom that controls *environment-name* continues its activity from the point at which it was hung. For example:

```
OK, PROP LQP -CONTINUE
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
OK,
```

The –CONTINUE option does not print a warning message at the printer indicating that the environment is continuing, unlike the –START, –HANG, –DROP, and –STOP options.

Stopping a Printer Environment

Use the following command to stop a printer environment:

PROP *environment-name* –STOP [*options*]

When stopping a printer environment, the despooler phantom prints the word *STOPPING*, in large, outlined letters at the printer.

The –FINISH option stops the phantom controlling the environment after it has finished printing the current file (if one is printing at that point). –FINISH is the default.

The –NOW option stops the phantom immediately. The print request remains in the spool queue, and is restarted from the beginning of the file when you start the phantom again.

The –IDLE option stops the phantom when there is nothing left for it to print.

If you want to use an alternate printer environment, you must first stop the printer environment that is currently running. Or, if you have modified a printer environment, you must stop the environment and then restart it in order to use the modified version.

The following example shows the PROP command to stop the LQP printer environment. Since no option was used, by default the PROP command stops the environment after it finishes printing a file already in progress.

```
OK, PROP LQP -STOP
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
OK,
*** LQP (user 102 on SYSA) at 10:33
Environment for LQP stopping

Phantom 102: Normal logout at 10:33
Time used: 01h 03m connect, 01m 23s CPU, 00m 48s I/O
```

Changing to Another Printer Environment

To change the environment that is currently running on a particular printer, you must first stop the current environment, and then start the alternate environment. If you are starting a new environment for a printer the first time, you need only to start the environment.

If you need to change paper or make physical adjustments to the printer, you can perform these tasks during the time between stopping the current environment and starting the alternate environment.

In the example that follows, the LQP environment is stopped and the alternate environment DOC is started.

```
OK, PROP LQP -STOP -FINISH
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
OK,
*** LQP (user 102 on SYSA) at 09:40
Environment for LQP  stopping

Phantom 102: Normal logout at 09:40
Time used: 00h 10m connect, 00m 21s CPU, 00m 07s I/O

OK, PROP DOC -START
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
OK,
*** DOC (user 103 on SYSA) at 10:03
Despooler for DOC ready
```

There may be some delay between the time you enter a command and the time the message displays at the printer.

You can use PROP –STATUS to make sure that the phantom has started.

If you do not need to make any changes to the printer itself, you can stop one environment and start another environment with the following single command:

PROP *env1* –RESET *env2*

In the following example, the LQP environment is stopped and the DOC environment is started on the same command line, using PROP –RESET.

```
OK, PROP LQP -RESET DOC
[PROP Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]
OK,
*** LQP (user 102 on SYSA) at 09:40
Environment for LQP  stopping

Phantom 102: Normal logout at 09:40
Time used: 00h 10m connect, 00m 21s CPU, 00m 07s I/O

*** DOC (user 103 on SYSA) at 09:40
Despooler for DOC ready
```

Modifying a Printer Environment

For efficiency, you should have one environment file for each type of printing job that users request regularly. Even if the paper type is the only element that is different between one printing environment and an alternate printing environment, it avoids the extra step of having to modify the environment file first before changing paper, and then stopping and restarting the environment in order to use the modified version.

Once you have created and tested the environment files you need for most print requests, you may want to avoid making mistakes in the environment files that users depend on to perform their day-to-day work.

However, there are times when it may make sense to modify an environment file to process an occasional special request. When modifying an environment file, be sure to save a copy of the original file under another name so that you can easily recover it later.

To modify an environment, you edit the environment file using an editor such as ED or EMACS. All environment files must reside in the SPOOL* directory. Chapter 3, Defining Printer Environments, contains information on creating environment files and using environment directives to control printers and their printing capabilities.

Checking a Modified Environment File

Once you have modified an environment file, save the modified version of the file under the same name, then check the syntax of the file contents using the PROP –VERIFY command. For example, if you modify an environment file called WHITE.ENV, enter the following command to verify the syntax:

PROP WHITE –VERIFY

If the file is correct, you can replace the active version of the WHITE environment with the new version by issuing the following command:

```
PROP WHITE -RESET
```

The modified version of the environment is sent to the despooler.

If the phantom is printing a file, the modified environment file does not take effect until it finishes printing the file. If the phantom is not printing a file, the modified environment takes effect immediately. When the modified environment takes effect, a message similar to the following is displayed on your terminal:

```
*** WHITE (User 97 on SYSA) at 9:43
Despooler for WHITE ready
```

If you modify the .ENV file for an active environment, and then issue PROP –DISPLAY, the display shows the current status of that environment and displays the contents of the current environment file, which is not the same data that the despooler phantom is using to operate the current environment. When you start an environment, the despooler phantom copies the environment data to its own data area into the environment LOG file.

Viewing Spool Queue Entries

The SPOOL –LIST command enables you to view the spool queue to monitor the progress of print requests. The SPOOL –LIST command displays information that may alert you to possible problems with the printers or with requests not printing for various reasons. For example, if a request appears to be sitting in the queue for an extended amount of time, it could indicate that a printer is either out of paper or jammed, or the request itself has the wrong attributes for the designated printer.

The following example shows the SPOOL –LIST –DISPLAY output, which shows more information than the abbreviated version that is displayed with SPOOL –LIST alone.

```
OK, SPOOL -LIST -DETAIL
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]

System SYSA
Request  Time       User                           Copies Size  State
-------- ---------- -------------------------- ------ ----- ------------
23 March 92
483      15:13:24   AB1                        1      1
   File name    <PMANGR>ACCTG>REPORT1
   Attributes   RASLW

23 March 92
731      16:31:32   RHONDA                     1      2
   File name    <BOOKS>RHONDA>FOOTBALL>FBNEWS.28
   Attributes   TP_QUM
   Deferred until 17:00:00 on 25 March 92
   Options       -SET_PORTRAIT

732      16:32:48   RHONDA                     1      1
   File name    <BOOKS>RHONDA>FOOTBALL>TEAMS
   Attributes   TP_QUM
   Options       -NO_COPY
OK,
```

Only privileged users can see all the requests in the spool queue. Nonprivileged users can see only their own requests unless the Spool Administrator has given specific users rights to view all spool queues. Refer to Chapter 2, Setting Up the Spooler Subsystem, for more information.

For information on creating custom displays for the SPOOL –LIST command, see Chapter 5, Tailoring Spooler Information. For more information on SPOOL command options, see Appendix A.

Canceling Requests in the Queue

Privileged users, SYSTEM (the supervisor terminal), and members of the .SPOOL_ADMINISTRATOR$ group can cancel any pending print request and stop any file that is currently printing on the local system. Nonprivileged users can cancel only their own print requests, including their own requests that are currently printing on the local system.

You may wish to stop a file that is currently printing because a

- File that is intended to drive a dot matrix printer has been sent to a line printer. The result is a large amount of meaningless output.
- Large file has started printing, and you want to stop it in order to allow several small files to print first, and then allow the large file to restart printing.

Canceling a File Not Yet Printing

To remove a request from the spool queue, use the following command:

```
      [–ON nodename  ]         [request–number [request–number . . . ]]
SPOOL [–DISK diskname] –CANCEL [–ALL                                  ]
```

–ON *nodename* option specifies the node containing the spool queue.

–DISK *diskname* option specifies a disk containing a pre-Rev. 21 spool queue.

request-number is the number assigned to the request in the spool queue that you can display using the SPOOL –LIST command as explained earlier in this chapter. The –CANCEL –ALL option removes all your own entries from the spool queue.

Stopping a File Currently Printing

There are several ways to stop a file from printing after it has already started. The method you choose to stop the file depends on your reason for stopping the file and the task you are attempting to perform, such as:

- You want to stop the file from printing and remove the request from the spool queue. You can do this in one of two ways:

  Use the SPOOL –CANCEL command as described in the previous section.

  Use PROP *environment-name* –DROP, which prints the word *DROPPING* in large, outlined letters at the printer. One advantage of using PROP –DROP is that you do not need to know the request number of the printing file.

- You want to stop the file from printing, but leave the request in the spool queue for the following reasons:

  You may want to stop a large file from printing to allow several smaller, but more urgent files to print first. In this situation, use the PROP *environment-name* –ABORT command to stop the current file from printing, then use one of the following methods to allow the smaller files to print first, before restarting the aborted file:

  Add the –RUSH option to the smaller print requests that you want printed first.

  Add a defer time to the request that you have just aborted.

  If a PROP –HANG or PROP –STOP is pending at the time you use the –ABORT option to stop a file from printing, the phantom suspends or stops the environment immediately after the file aborts (stops printing).

- Use the PROP–SUSPEND command to suspend the currently printing job and allow other files to print. You can then release the suspended job with –RELEASE, and restart it from the top of the page where it was suspended. Only one job can be suspended at any one time on the system.

You must be a privileged user to issue the –DROP, –ABORT, –SUSPEND, and –RUSH options of the PROP command, unless the Spool Administrator has given access rights to specific users to perform these operations for a particular environment.

Note You cannot cancel a file on a remote node that has already started printing.

Modifying Spool Requests

Users may need to modify their spool requests after they have issued the SPOOL command to

- Change the paper type or the number of copies required.
- Add or remove a SPOOL option or printer attribute.
- Defer printing to a later time or change it.
- Send the file to a different printer.

For example, a user may have spelled an attribute incorrectly on the command line and needs to modify the request before it is printed.

Use the –MODIFY option of the SPOOL command whenever you need to modify a spool request. You can issue the –MODIFY option to change any characteristics of your own print requests. However, only privileged users, that is, SYSTEM and members of the .SPOOL_ADMINISTRATOR$ ACL group, can modify requests of other users.

Use the following command format to change the printing characteristics of a file in the spool queue:

```
SPOOL [–ON nodename    ] –MODIFY request–number [options]
      [–DISK partition ]
```

request-number is the number given to the request by the Spooler subsystem. You can find the request number given to your request by looking at the SPOOL –LIST display.

options are any SPOOL options that define job characteristics.

You must use the –ON option or the –DISK option whenever the file you wish to modify is not in the local spool queue.

The –ON or the –DISK option identifies the queue holding the request to be modified. –DISK is used for a pre-Rev. 21.0 queue. You cannot move a print request from one spool queue to another by using –MODIFY.

Changing the Printing Order of Requests in the Spool Queue

Privileged users can use the –RUSH option of the SPOOL command to move one or more print requests ahead of the other print requests in the queue before the despooler phantom begins processing the next request in the queue.

The –RUSH option is useful in order to allow smaller files to print ahead of very large files in the spool queue, or to allow an important document to print ahead of other requests in the queue.

You need to consider the following points when using the –RUSH option:

- It does not stop a file that is currently printing. The file with rush priority prints after the current file finishes.
- If there is more than one file with rush priority, they are printed in the order that the requests were put in the spool queue, but before other requests.
- Use of the –RUSH option is restricted to the supervisor terminal and members of the .SPOOL_ADMINISTRATOR$ ACL group.
- –RUSH does not take priority over queue search directive settings. This means that files in other queues may be printed before the file with rush priority.
- The –RUSH option overrides any –DEFER option.

7 Resolving Operational Problems

Introduction

This chapter explains how to identify and solve some common software and hardware problems that may occur in using the Spooler subsystem and printers:

- Files not printing
- Mechanical problems

The essential first step in solving a problem is to determine whether the problem is with the hardware or with the software. If you are unfamiliar with a particular aspect of the Spooler subsystem or a particular printer, leave the error condition and contact your PrimeService^{SM} Representative. Be sure to write down any information, such as error messages, that will aid in resolving the problem.

Files Not Printing

Users may report problems with their files not printing. Assuming that the printer is turned on, plugged in, and working correctly, there may be several reasons why a file will not print. The most typical problems that prevent files from printing are listed below, with a solution for correcting them.

Problem	*Solution*
Environments appropriate to the print request are active, but the despooler phantom is printing other users' files.	The file will print when a printer with the appropriate environment becomes available.
Appropriate environments exist for printing the file, but they are not active.	Start a despooler phantom for the environment that will print the file. You may need to stop one of the active despooler phantoms to release a printer for the new environment that you wish to start.
An appropriate environment for printing the file does not exist.	Create a new environment for printing the file, or modify the request so that it can be printed by another available environment. Refer to Chapter 3, Defining Printer Environments.

Mechanical Problems

If there are problems with the operation of a printer, take the following corrective action:

1. Check that the printer is not jammed or out of paper.
2. Check that the printer is ready to print.
3. Check that a suitable despooler phantom is running by using the PROP –STATUS command. If no despooler is controlling the printer, start one by issuing the command PROP *environment-name* –START.
4. Make sure that the paper mounted on the printer matches the type called for in the printer's environment. Check the specified paper by typing PROP *environment-name* –DISPLAY.

Caution

If more than one parallel printer is connected to a URC controller, and one of them shuts down (because of a blown fuse or power supply problem), then it is possible that the controller will hang. In order to recover, you must empty the data buffer in the controller. Usually a warm start recovers the situation, but if that fails, you will need to perform a system cold start. You should repair or replace the faulty printer before the cold start. If that is not possible, then you must not start a despooler phantom that controls the faulty printer. The cold-start procedure is described in Chapter 2, Setting Up the Spooler Subsystem.

Aligning Forms

A file must be printing to align the paper correctly. You should create a standard line up test file and print it whenever you want to realign the printer. To align the paper to the top of form, use the –LINEUP option of the PROP command as follows:

1. Type the command PROP *environment-name* –LINEUP.
2. Wait for the despooler phantom to display this message:

   ```
   Check paper alignment
   ```
3. Check the alignment of the paper on the printer. The printer should have printed four lines at the top of the page (top of form).
4. If the paper is aligned correctly, skip steps 5 and 6 and proceed to step 7. If the paper is not aligned correctly, go to step 5.
5. Align the paper so that four lines are at the top of the form.
6. Check the alignment by repeating steps 1 through 4.
7. Issue the command PROP *environment-name* –CONTINUE.

Recovering the Printer After a Paper Jam

If the printer runs out of paper or ribbon, you may be able to load a new box of paper or mount a new ribbon without losing or destroying part of the printed output. If so, you will be able to start printing again without using the PROP command. The phantom continues printing where it stopped.

If you encounter a paper jam or other situation where output is lost, follow these steps:

1. Take the printer offline and attend to the problem, that is, load a new box of paper on the printer or remove a torn page and reset top of form. Prepare the printer to start printing.
2. At this point you can take one of three possible actions:
   - Abort printing the current file, but leave the request in the print queue, by issuing the command PROP *environment-name* –ABORT. Take this course if there are other requests in the queue that need to be dealt with more urgently than the file that was printing when the problem arose.
   - Reprint the current file from the start, by issuing the command PROP *environment-name* –RESTART. Use this option if little of the file has printed.
   - Reprint the current file from a suitable point earlier in the document, by issuing the command PROP *environment-name* –BACK *n*, where *n* is the number of pages that you want to back up. This is a maximum of 100 pages. If *n* is greater than 100 pages, the despooler phantom

reprints the file from the beginning. You may need to use PROP –STATUS to find out how many pages had been printed before the paper jam, and so decide how many pages to back up.

You can use PROP –BACK with PostScript printers, and the command works correctly if the laser printer is emulating a line printer, or is producing small facsimile pages on which the layout emulates line printer output.

The –BACK option does not necessarily have a useful effect if the PostScript procedure does complex text formatting. Also, if you must switch off a laser printer, the PostScript procedures must be reloaded before it can continue. In either case, you should use the –RESTART or –ABORT option.

Appendices

SPOOL and PROP Commands

A

Introduction

This appendix describes the SPOOL and PROP commands and their options. It includes the command format, option descriptions, and examples of use.

Spool Command

The SPOOL command is used for

- Adding a request to the spool queue
- Modifying a spool request
- Canceling a spool request
- Viewing the spool queue

Format

```
        ┌pathname [print–options]                ┐
        │–MODIFY request–number [options]        │
SPOOL  <│–LIST [request–number] [options]        │>
        │–CANCEL [{options}]                     │
        └–HELP                                   ┘
```

Adding a Request to the Spool Queue

To print a file, you add a request to the spool queue by entering the following command:

SPOOL *pathname [print-options]*

pathname is the pathname of the file to be printed. If the file is in your current directory, you can give the filename only. *print-options* are listed below.

Options

–ALIAS *name*
–ALI

Allows you to replace the user ID that is printed in large letters on the file header page with another *name* of 32 or fewer characters. If *name* contains a space, you must enclose the whole name in single quotations. The user ID from which the file was spooled is still used for other purposes, such as in accounting data.

You cannot use –ALIAS on a pre-Rev. 21.0 system, nor can you use it with the –DISK option because it places the spool request in a pre-Rev. 21.0 system queue which does not support aliases.

–AS *name*

Allows you to replace *pathname*, which is printed in large letters on the file header page and in SPOOL –LIST displays, with another *name* of 32 or fewer characters. If *name* contains a space, you must enclose it in single quotations. You may want to replace *pathname* with a more meaningful name, such as the name of your document, for example.

–ATTRIBUTE *name* [*name* . . .]
–ATT

Selects one or more attributes used for spooling the file. **Attributes** are named characteristics of a printer and its environment, such as the paper type, printer destination, and so on. You can use the –ATTRIBUTE option as many times as needed on the command line.

In a pre-Rev. 21.0 spool queue, the –AT option selects a printer destination, and –FORM selects the paper type. If you do not supply a name with the –AT option, the despooler checks for an L.DFLT file in the SPOOLQ directory on the specified partition, and, if it exists, uses the default destination name contained within that file.

–CANCEL *n* [*n* . . .] [*options*]
–CAN

Cancels one or more print requests. For more information on canceling a print request, see Canceling a Print Request later in this appendix.

–COBOL
–COB

Prints COBOL format files. This option may also be used when submitting requests to pre-Rev. 21.0 systems.

–COPIES *n*
–COP

Prints *n* number of collated copies of the spooled file, up to 99 copies. The default is one copy. When more than one copy is printed, only one header is produced before the first copy.

–DEFER [*hh:mm*]
–DEF

Defers printing of a file to a time that you specify. Specify *hh:mm* in either 24-hour format (00:00 = midnight) or in 12-hour format (12:00AM = midnight). The colon is optional. You must not leave a space between the digits and the AM or PM. If you do not include a time with the –DEFER option, midnight is assumed. In a network that crosses time zones, the deferred time is determined by the machine on which the queue is located. Formats matching the PRIMOS DATE command –FULL, –UFULL, and –VFULL options are also supported.

–DISK *diskname*
–DIS

Enters a spool request on a disk containing pre-Rev. 21.0 spool queue. In order for existing programs to run unchanged on a Rev. 21.0 or later system, the SPOOL command ignores the –DISK option if it refers to a disk on the local system. Furthermore, if an attempt to add a request to a remote spool queue fails, and the remote system is running Rev. 21.0 or later, the SPOOL command changes the –DISK option into an appropriate –ON option and tries again. *diskname* is the name of a disk partition.

–ERROR_BRIEF	**–ERRB**
–ERROR_MEDIUM	**–ERRM**
–ERROR_DETAIL	**–ERRD**

Each ERROR option displays Spooler error messages in a slightly different format. The default is –ERROR_MEDIUM.

–FROM *m* **–TO** *n*
–FRO

Identifies a range of pages to print. *m* and *n* are the page numbers of the first and last pages you want to print. In normal page formats, this option prints the physical pages *m* through *n* inclusive, and prints logical pages when in FORTRAN or COBOL format.

You can omit –FROM or –TO. The default values are from page one to infinity. The –FROM value must not exceed the –TO value. You can use both the –FROM and –TO options with the –COPIES *n* option to print multiple copies. Note that embedded specific escape sequences from pages prior to *m*, for example, to establish a different format or to select a different font, may not take effect because the sequences are not sent to the printer.

–FTN

Specifies that printing should commence in FORTRAN format.

–HEADER $\left\{\begin{array}{l}\textit{text}\\ \textbf{–NO\_INITIAL\_FF}\end{array}\right\}$
–HDR

Allows you to replace the default page header with a header of your own choosing. The default page header is the first line of the file. (Do not confuse this with the header page of the spooled document.)

text is the page header that you want printed. Any format commands embedded in the file that change the page header still function. If *text* contains spaces, you must enclose the entire text in single quotations. You can use the keyword FILE in place of *text* to use the pathname of the file as the page header. This option applies only to files that are paginated. It does not work with FORTRAN or COBOL files, for example.

–HELP
–HEL

Displays the SPOOL command syntax and options.

–LIST [*options*]
–LIS

Lists one or more spool queues. For more information on listing files in the spool queue, refer to Viewing the Spool Queue later in this appendix.

–LNUMBERS
–LNU

Prints a line number, enclosed in parentheses and followed by a space, at the beginning of each line in the data file. If a line occupies more than one line of printed output, the number is printed only on the first line. Lines that overprint the previous line are not numbered.

–MODIFY [*options*]
–MOD

Changes the options of a print request. For more information on modifying a spool request, refer to Modifying a Spool Request later in this appendix.

–NO_COPY
–NOC

Prints a file from its current file system location. The file is not copied to the SPOOL_DATA* directory. Any changes made to the file between the time it is spooled and the time it is printed appear in the printed output. The file must reside on the same node as the spool queue to which the request is added. The despooler phantom that processes the request must be able to attach to the directory that contains the file (R access), and must have read access to the directory containing the file (LUR). You cannot use the –NO_COPY option to send requests to pre-Rev. 21.0 queues.

–NO_DEFER
–NOD

Cancels a previous –DEFER option when used the –MODIFY option.

–NO_EJECT
–NOE

Disables the form feed that normally occurs after the file has finished printing. Because a form feed is never performed before a header page, use this option only when using the –NO_HEADER option that does not print a header page.

–NO FORMAT [–AT TEXT]
–NOF

Disables the Spooler format control (pagination and header generation). It is also used for files containing EVFU skip-to-channel commands. It cannot be used with the –PLOT option.

This option is also used when printing text files to a PostScript printer. For PostScript printers, –NO_FORMAT, when used with –AT TEXT, treats a spooled file as a text file, not as a PostScript file. This allows you to disable the Spooler's page formatter when not using –PROC to specify a PostScript procedure.

If –POSTSCRIPT_OVERRIDE is included with the DEVICE _POSTSCRIPT directive in the environment file, the –AT TEXT option no longer works. For more information on PostScript printing, see Appendix D.

–NO_HEADER
–NOH

Prints the file without header pages.

–NOP

Inhibits overprinting, whether required by the FORTRAN or COBOL format + control character, or by trailing carriage return characters. The Prime convention is to end lines with a line-feed character only. The –NOP option is useful with output that uses a carriage return (with both CR and LF, –NOP would likely double space the output) as the line terminator.

–NO_RUSH
–NOR

Cancels a previous –RUSH option when used the –MODIFY option.

–NO_SWO
–NOS

Cancels the –SPOOL_WHILE_OPEN process when used with the –MODIFY option. The despooler phantom prints the file from its current file system location (the same operation performed with the –NO_COPY option) and drops the special end of file (EOF) handling.

–NOTIFY
–NFY

Notifies the user who spooled the request when printing is completed for local nodes only. You cannot use this option with the –ON option. An error message displays if you attempt to use –ON with –NOTIFY.

–NO_XLATE
–NOX

Cancels the –XLATE (alternative mapping) option when used with the –MODIFY option.

–NPH

Disables the page header and page numbers. Note that –NPH is not the same as –HEADER or –NO_HEADER. The page header is also disabled if the environment file has the FORMAT –TM directive set to 0 or 1.

–ON *nodename*

Allows you to add a request to a Rev. 21.0 or later spool queue on a specific *nodename* on a network.

–OPEN
–OPE

Opens a data file in the SPOOL_DATA* directory. Data is generated by a program and directed to the data file. The file is opened on unit 2 unless you specify otherwise. (See –TUNIT.)

–PLOT *n*
–PLO

Specifies the plot raster size for a plotter output file. This option does not send a file to a plotter. To select a plotter, use –ATTRIBUTE with the valid attributes for your system. The print mode will be PLOT format. This option cannot be used with the –NO_FORMAT option. *n* is the raster size in words per scan. The default is 128 for a 200-raster/inch plotter.

–PROC *name*
–PRO

Selects a PostScript procedure for printing a document on a PostScript laser printer. You must load the PostScript procedure when you start up the printer.

If the procedure is not currently loaded, which could happen if power to the printer is switched off and then on again, then it is reloaded. Refer to Appendix D, Defining and Using PostScript Printers, for more information on PostScript procedures.

–RUSH
–RUS

Selects a spool request for printing ahead of other requests in the same spool queue, ignoring any size or defer time restrictions. Only privileged users and members of the ACL group .SPOOL_ADMINISTRATOR$ can change the priority of a request in the queue.

–SET_FONT *fontname*

Selects a particular typeface for printing a file. *fontname* is a character string with a maximum length of 32 characters. Refer to Appendix D, Defining and Using PostScript Printers, for more information on fonts, options, and printer environments.

–SET_LANDSCAPE

Selects landscape paper orientation that prints text across the longest width of a given size of paper. –AT_DRAFT applies to any PostScript job (portrait or landscape) that does not use –PROC.

–SET_LANDSCAPE has special attributes for use with PostScript printers for printing multiple formatted pages on one sheet of paper (referred to as *n*-up printing, or thumbnail printing). Refer to Appendix D, Defining and Using PostScript Printers, for more information on fonts, options, and printer environments.

–SET_PAPER_BIN $\begin{Bmatrix}\text{MANUAL}\\ n\end{Bmatrix}$

Selects a printer paper bin from which to feed paper for printing. Some laser printers have more than one paper bin. MANUAL sets the paper bin for feeding paper manually. *n* is a number from 1 through 9, inclusive.

–SET_PORTRAIT

Selects portrait paper orientation that prints text across the shortest width of a given size of paper. –AT_DRAFT applies to any PostScript job (portrait or landscape) that does not use –PROC.

Special PostScript printer attributes can be used with –SET_PORTRAIT that enable you to print multiple formatted pages on a single sheet of paper (also referred to as *n*-up printing, or thumbnail printing). Refer to Appendix D, Defining and Using PostScript Printers, for more information on fonts, formatting options, and printer environments.

–SFI

Prevents the printing of the pathname and the date/time modified notifications on the header page and the trailer page of the file. This option is useful when the file infomation is considered confidential. However, SPOOL –LIST –DETAIL displays the pathname even if you use the –SFI option.

–SPOOL_WHILE_OPEN
–SWO

Prints a file while it is open for writing to disk. If you use the –SWO option, the file is not copied into SPOOL_DATA*. Without –SWO selected, the despooler terminates when it reaches the end of file (EOF). With the –SWO option, the despooler waits for a defined time, and checks a number of times before it terminates: the Wait Time and Times to Check parameters are defined in the printer environment file. Using the –SWO option does not guarantee immediate printing. Printing starts when a printer with the correct environment is available.

–TRUNCATE
–TRU

Truncates all lines that are longer than the printer width defined in the environment file. (Use the PROP –DISPLAY command to list the value of the –width parameter of the FORMAT environment directive.) If you do not use this option, lines longer than the printer width wrap around two or more lines so that no printed text is lost. If overprinting is specified for a line within a FORTRAN-formatted or COBOL-formatted file and that line is split, overprinting is ignored. If overprinting is ignored, the overprint text is printed on a separate line.

–TUNIT *n*
–TUN

Specifies the file unit number associated with the –OPEN option. *n* is the file unit number. If you do not use –TUNIT, the default file unit number is 2.

–XLATE *mapping*
–XLA

Selects an alternate character set for the printer. *mapping* is a 1–32 character string that you use to select the mapping. The first character must be alphabetic. The remaining characters may be alphanumeric, numeric, or any of these three characters: . $ _.

Modifying a Spool Request

Privileged users can modify any request in the spool queue. General users can modify only their own requests. Use the following command format to modify a request in the spool queue:

```
      [-ON nodename    ]
SPOOL [-DISK partition ] -MODIFY request-number [options]
```

request-number is the request number assigned to the spooled file.

Use SPOOL –LIST to display the request number of an entry in the spool queue. Refer to Viewing the Spool Queue later in this appendix.

When modifying a spool request, you can use all of the options available for adding a request, with the exception of –NO_COPY, –OPEN, –TUNIT, and –SPOOL_WHILE_OPEN. Modifying a request usually adds an option. If the new option conflicts with a previous option, the previous option is removed. For example, specifying –FTN would cancel an existing –NO_FORMAT in the original request.

If you change the attributes of a request with the –ATT option, you must reissue all of the attributes for that request.

To cancel a –DEFER or a –RUSH option, use –NO_DEFER and –NO_RUSH, respectively.

Canceling a Print Request

To cancel a request you must know its request number. You can find the request number of an entry in the spool queue by viewing the queue with SPOOL –LIST, as described later in this appendix. If you are a privileged user, you can cancel requests of other users by request number, otherwise you can cancel only your own requests.

Use the following command format to cancel print requests:

```
      [-ON nodename   ]         [request-number [request-number . . . ]]
SPOOL [-DISK diskname ] -CANCEL [-ALL                                  ] [- LIST]
```

nodename is a nodename.

diskname is a pre-Rev. 21.0 disk name.

request-number is the request number of the spooled file. You can use the –ALL option in place of a request number to cancel all of your requests. If you are a privileged user, you cannot use –ALL to cancel other users' print requests. To cancel another user's print request, you must supply the request number.

You can use the –LIST option of SPOOL on the same command line to verify that the request has been removed from the spool queue.

Viewing the Spool Queue

Privileged users can view the entire queue on Rev. 21.0 nodes and later. Other users can view only their own requests. The Spool Administrator can change this restriction. Refer to Chapter 2, Setting Up the Spooler Subsystem, for more information.

Use the following command format to view entries in the spool queue:

SPOOL –LIST [*request-number*] [*options*]

Option	*Description*
–ALL	Lists all queues as defined by the Spool Administrator in the file SPOOL*>QUEUES.
–ATTRIBUTE ***name*** **[*name* . . .]** **–ATT**	Reports only entries with given attributes. *name* is an attribute name.
–BRIEF **–BR**	Produces a short report. This is the default option and is the same as SPOOL –LIST
–DETAIL **–DET**	Produces a more detailed report.
–DISK ***diskname*** **–DIS**	Reports requests in a pre-Rev. 21.0 queue on the named partition.
–FULL **–FUL**	Gives the same report as –DETAIL, and includes the partition name of the SPOOL_DATA* directory containing the spooled file.
–NO_WAIT **–NW**	Disables the `--More--` prompt and enables continuous scrolling when the display continues on more than one screen.
–ON ***nodename***	Reports requests in the queue on the named node.
–USER ***name*** **–USE**	Reports only requests for the named user ID.

Example 1: The –DETAIL option displays detailed information on requests in the spool queue.

```
OK, SPOOL -LIST -DETAIL
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]

System SYSA
Request  Time       User                       Copies Size  State
-------- ---------- -------------------------- ------ ----- ------------

23 March 92
483      15:13:24   AB1                        1      1
   File name   <BOOKS>AB1>COUGAR
   Attributes  RASLW

785      10:31:08   MELIS                      1      3
   File name   <BOOKS>MELIS>COMMS>MINUTES>$CELIA
   Attributes  QMST
   Options      -NO_FORMAT -NO_HEADER

792      11:22:56   RHONDA                     1      2
   File name   <BOOKS>RHONDA>FOOTBALL>TEAMS
   Attributes  TP_QUM, WHITE
   Deferred until 17:00:00 on 26 March 92
   Options      -SET_PORTRAIT
OK,
```

Example 2: The –FULL option of SPOOL –LIST displays the same information as the –DETAIL display, and includes the partition name of the SPOOL_DATA* directory that contains a copy of the spooled file.

```
OK, SPOOL -LIST -FULL
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]

System SYSA
Request  Time       User                       Copies Size  State
-------- ---------- -------------------------- ------ ----- ------------
23 March 92
483      15:13:24   AB1                        1      1
   File name   <BOOKS>AB1>COUGAR
   Attributes  RASLW
   Data partition PUBS

785      10:31:08   MELIS                      1      3
   File name   <BOOKS>MELIS>COMMS>$CELIA
   Attributes  QMST
   Options      -NO_FORMAT -NO_HEADER
   Data partition PUBS

794      11:26:00   RHONDA                     1      2
```

```
   File name    <BOOKS>RHONDA>FOOTBALL>FBNEWS.28
   Attributes   CARBON, WHITE
   Deferred until 17:00:00 on 26 March 92
   Data partition PUBS

795      11:27:00   RHONDA                     1      2      (Printing)
   File name    <BOOKS>RHONDA>FOOTBALL>FBNEWS.37
   Attributes   CARBON
   Options       -NO_HEADER
   Despooler    TP_QUM on node: SYSA
   Data partition PUBS
OK,
```

The –DETAIL and –FULL displays provide information about the user who spooled the request, the time it was spooled, number of copies, total size in records, and file pathname. The displays include the pathname of the spooled file even for requests that include the –SFI option that prevents file information from printing on header and trailer pages.

The attributes entry includes the attributes given explicitly with the request, and any attributes supplied by default.

The options entry lists the options given with the SPOOL command, and displays only if at least one option is given with the spool request.

The defer time entry shows the earliest time at which the file will print.

If the display includes requests from an earlier date, the date is included before the first reported request and each time the date changes.

Example 3: The –BRIEF option displays a single line of information for each spool request. –BRIEF is the default. Entering SPOOL –LIST –BRIEF or SPOOL –LIST produces identical listings.

```
OK, SPOOL -LIST -BRIEF
[SPOOL Rev. 23.3.0 Copyright (c) 1992, Prime Computer, Inc.]

System SYSA
Request  Time   User             File                          No Size  State
-------- ----- ---------------- ---------------------------- -- ----- -----
23 March 92
483      15:13 AB1              COUGAR                        1  1
785      10:31 MELIS            $CELIA                        1  3
794      14:15 RHONDA           FBNEWS.28                     1  2
795      14:15 RHONDA           FBNEWS.37                     1  2
OK
```

You can also create your own spool queue display format that is produced by the SPOOL –LIST command. Refer to Chapter 5, Tailoring Spooler Information, for more information.

PROP Command

PROP allows you to control the operation of your system's printers and printer environments, and monitor their status. Operators and the Spool Administrator use PROP to perform the following printer operations:

- Monitor the status of printer environments.
- Control despooler phantoms and their environments.
- Change to another printer environment.
- Verify a printer environment.
- Suspend a currently printing job and restart it later.

Format

$$\textbf{PROP}\left\{\begin{array}{l}\textit{environment–name}\ [\textit{options}]\\ \{\textit{system–options}\}\\ \textbf{–HELP}\end{array}\right\}$$

environment-name is the name of a printer environment.

All users can give the PROP –STATUS, –HELP, –REPORT_SIZE, and the PROP *environment-name* –DISPLAY command. Only the Spool Administrator and privileged users can use all other PROP command options.

At Rev. 23.3, the Spool Administrator can give specific users RX access rights to abort, back up, continue, drop, hang, line up, release, reset, restart, start, stop, and suspend the printer by changing the ACL on the environment file.

Environment Options

This section lists the PROP command options used to control printer environments.

–ABORT
–AB

Stops printing the current file without removing the request from the spool queue. This option also forces a previously requested paper change, hang, or shutdown to occur immediately, if the request was made using the –FINISH suboption of the PROP command option (PROP *environment-name* –HANG –FINISH, for example). The –ABORT option also implies a –CONTINUE.

–BACK *n*

Backs up *n* pages and reprints the file from that point forward. This option is useful after clearing a paper jam or after fixing other problems with the printer. *n* must be a number greater than 0. The default value of *n* is 1; the maximum value of *n* is 100 (allowing you to back up to 100 pages). The –BACK option is useful when very large files have been partially printed because of a printer problem, and allows an Operator to restart printing of the file beginning from the point at which the printer problems began.

–CONTINUE
–CONT

Restarts printing (or queue scanning) after using –HANG or –LINEUP.

–DISPLAY [–NO_WAIT]
–DISP [–NW]
–DP

Displays the status of the named environment and lists the environment file contents. The –NO_WAIT option enables continuous scrolling if the display spans multiple pages. If you do not use –NO_WAIT, the `--More--` prompt displays between pages, in which case you press the Return key between pages.

–DROP

Stops the job that is currently printing and deletes the request from the spool queue.

$$\textbf{–HANG} \left[\begin{Bmatrix} \textbf{–FINISH} \\ \textbf{–NOW} \\ \textbf{–IDLE} \end{Bmatrix} \right]$$

Temporarily suspends printing and queue scanning by stopping the despooler phantom that controls the named environment. This option allows an Operator to attend the printer to change the ribbon, or perform some other adjustment to the printer. Use the –CONTINUE option to restart the phantom. The –NOW, –IDLE, and –FINISH options determine when to suspend the despooler. –NOW suspends the despooler immediately, even if a file is currently printing. –FINISH suspends the despooler after the file has finished printing (the default). –IDLE suspends the despooler when it becomes idle (when there are no more files to print).

When the despooler phantom hangs, it sends the message `Hanging` to the terminal; printing may continue for a short time while the buffers empty.

–LINEUP [*linum*]
–LU

Backs up the despooler to the start of the current page, prints *linum* lines of the document, then pauses and displays the message `paper alignment` `check`. Restart the despooler phantom using PROP with either the –CONTINUE option (which does not reprint the first part of the data) or the –LINEUP option to repeat the lineup sequence.

Note You may include –LINEUP as part of the –START command, as in

```
PROP PR1 -START -LINEUP 5
```

In this case, the lineup is done on the first file eligible for printing.

–NO_WAIT
–NW

Disables the `--More--` prompt between screen displays and enables continuous scrolling.

–RELEASE
–REL

This option releases a suspended print job, and restarts it from the top of the page at which it stopped, after printing the current file. Otherwise, the suspended job is released immediately.

–RESET [*new_env*] $\left[\begin{Bmatrix}\textbf{–FINISH}\\ \textbf{–NOW}\\ \textbf{–IDLE}\end{Bmatrix}\right]$

Stops and restarts a despooler phantom. This is useful if the environment file has been altered. The optional new environment name, ***new_env***, allows you, with a single command, to replace one despooler environment with another. The –NOW, –FINISH, and –IDLE options determine when to stop the current environment and start the new environment. –NOW resets the environment immediately. –FINISH resets the environment after the current file has finished printing, and –IDLE resets the environment when there are no more requests to print.

–RESTART
–RES
–RST

Restarts the current print job.

–START

Starts the named environment, provided that you have access to the environment file.

–STOP $\left[\begin{Bmatrix}\textbf{-FINISH}\\ \textbf{-NOW}\\ \textbf{-IDLE}\end{Bmatrix}\right]$

Shuts down the named environment. The options –NOW, –FINISH, and –IDLE determine when to shut down the environment. –NOW shuts down the environment immediately, even if a file is currently printing. –FINISH shuts down the despooler after the file has finished printing. –IDLE shuts down the despooler when it becomes idle (no more files to print).

–SUSPEND
–SUS

Suspends the current printing job. A suspend operation occurs immediately.

–VERIFY
–VFY

Checks the syntax of an environment file and notifies the user of any errors.

System Options

This section lists the PROP command options used to control the Spooler operations and certain spool queue settings.

–COLDSTART [–NO_QUERY]
–COLD [–NQ]

Used during machine cold start to initialize the Spooler subsystem data structures and to reset the queue control information. This option may be issued manually during cold start or placed in the PRIMOS.COMI file (which is the recommended procedure). (See Chapter 2 for cold-start procedures.)

At Rev. 23.3, the –NO_QUERY option enables you to cold start the Spooler subsystem without having to answer YES to the query message `Okay to shut down env`. Only members of the .SPOOL_ADMINISTRATOR$ group are allowed to use this option.

Caution If you use the –NO_QUERY option, it may still take time for the active environments to stop. Do not start any new environments without first verifying that the phantoms have logged out.

–ENV_LANGUAGE ***lang-name***
–ELANG

Defines the environment file language using the Standard Internationalization Tool (SIT). This option is useful only at multilingual sites. You can use this option with the –START and –VERIFY options.

–HELP
–H

Displays the PROP command syntax and options.

–MAX_SIZE [*nnnn*]
–MAX

Sets the maximum disk record size of files entered into the local spool queue. Spooled files with record sizes greater than the maximum set by –MAX_SIZE are rejected by the spool queue. *nnnn* is the number of records. If you omit –MAX_SIZE, there is no limit on the file size. Only members of the .SPOOL_ADMINISTRATOR$ group are allowed to use this option. The default maximum size is zero. You can reset the maximum size to zero by entering 0 or nothing on the command line, if you do not wish to place any limits on the size of files entered into the spool queue.

–MIN_SIZE [*nnnn*]
–MIN

Sets the minimum disk record size of files entered into the local spool queue. Spooled files with record sizes less than the minimum set by –MIN_SIZE are rejected by the spool queue. *nnnn* is the number of records. If you omit –MIN_SIZE, the default value of one disk record is used. Only members of the .SPOOL_ADMINISTRATOR$ group are allowed to use this option.

–REPORT_SIZE
–RPT_SIZE

Reports the current system limit established for the size of spool requests allowed into the spool queue.

–STATUS $\begin{bmatrix} \textbf{–ALL} \\ \textbf{–NO\_WAIT} \end{bmatrix}$

Displays a list of currently active environments and indicates the status of each environment, such as whether it is printing, idle, hanging, and so on. If an environment is currently printing, the display includes the name of the file that is printing, current copy, request number, node name (if printing from another system), and number of the page that is currently printing. If you use –ALL, the status is displayed for all environments for which environment files exist. The –NO_WAIT option disables the `--More--` prompt between multiple screen displays which allows continuous scrolling.

Writing a Print Handler

B

■ ■ ■ ■ ■ ■ ■

Introduction

This appendix contains the following topics:

- What is a print handler
- What you need
- How the despooler users print handlers
- Using print handler functions
- Using low-level device drivers
- Calling interfaces
- Building a print handler
- Using data structures

What Is a Print Handler?

A **print handler** is software that a despooler phantom uses to perform standard output operations and perform other device handling for a particular output device. Print handlers are not user-visible.

Advantages of Print Handlers

The print handler is entirely separate from the despooler. Separated in this way, all output devices appear the same from the viewpoint of a despooler phantom, and the despoolers can use similar output control commands for all the print handlers. Consequently, you can support any new output device by writing a suitable print handler. There is no need to make any change to the despooler code.

Print Handlers Supplied by Prime

Prime supplies the following print handlers:

DPTX
FUJITSU
HPJET
POSTSCRIPT
NEC
QMS
QUME
TTY
DATAPRODUCTS
DIABLO
PRINTRONIX

The following table lists the print handler name, the printer type, and the line protocol for each.

Print Handler Name	*Printer Type*	*Line Protocol*
FUJITSU	Band printer	Async, XON/XOFF, parallel, MPC4000
HPJET	HP LaserJet	Async, parallel
DPTX.PLP	DPTX	DPTX
POSTSCRIPT.PLP	PostScript	Serial line using XON/XOFF, no echo (does not support parallel)
NEC_OAS.PLP QUME_OAS.PLP	OAS_PRINTING using ETX/ACK protocol	Serial line but not using XON/XOFF
TTY.PLP	Transparent, but recognizes EVFU skips	Parallel or serial

Code for the print handlers is located in SPOOL*>DESPOOL_LIBRARY.RUN, which is a program class EPF.

Use the following guidelines to determine the print handler and communication protocol for the type of printer you are using:

- The PostScript and DPTX printers each have their own special print handler which should be used.
- If your printer is supplied with a print handler, use that print handler.
- The communication medium used (serial, parallel, PDN, TCP/IP) provides information on which print handler to use.

- If a parallel communication medium is used, try one of the following print handlers: TTY, Dataproducts, or Printronix.
- If a serial communication medium with XON/XOFF is used, try one of the following print handlers: NEC, QUME, or DIABLO™.
- If a serial communication medium with ETX/ACK is used, try one of the following print handlers: NEC_OAS or QUME_OAS.

Selecting a Print Handler

You select a print handler using the DEVICE directive in the environment file, as explained in Chapter 3, Defining Printer Environments.

This appendix contains information on how to create additional print handlers for devices that are not directly supported by Prime.

What You Need

The despooler and the print handler software supplied by Prime is coded in PL1, therefore it is recommended that you also work in this language. In addition to the standard insert files in SYSCOM, PL1 insert files are provided in the SPOOL*>SOURCE directory. Some support is given for FTN so this could be a second choice. You can use other languages, but this entails more work.

When you have coded and compiled a new print handler, you should then test it. To test a new print handler, create a separate library EPF, and modify the search rules in SPOOL*>ENTRY$.SR so that a despooler phantom can find it. When the print handler is fully tested, you should add it to the despooler search rules.

See the *Programmer's Guide to BIND and EPFs* to find out how to create EPFs with the BIND command.

The source code should follow the pattern below:

Procedure label and declaration
Standard insert files from SPOOL*>SOURCE
Standard insert files from SYSCOM
Required data declarations
Replacement strings
Further data declarations
Body of program

You should construct the body of the program as a select clause that deals with each possible print handler function in turn.

Refer to Example of a Print Handler later in this appendix for an example that you can use as a model when you construct your own print handlers.

How the Despooler Uses Print Handlers

The structure of the despooler software is shown in Figure B-1. SPOOL*>DESPOOL.RUN is a program class EPF that is invoked for each despooler phantom that is started. The PRINT_FILE routine within DESPOOL.RUN issues print directives to the appropriate print handler in SPOOL*>DESPOOL_LIBRARY.RUN.

The print handler in turn makes use of low-level device drivers that are part of DESPOOL.RUN. These low-level device drivers provide a common interface to the PRIMOS device/line driver subroutines. Figure B-1 also shows SP$FLP, which is the entrypoint for a routine that print handlers use to convert ECS data to ASCII-8.

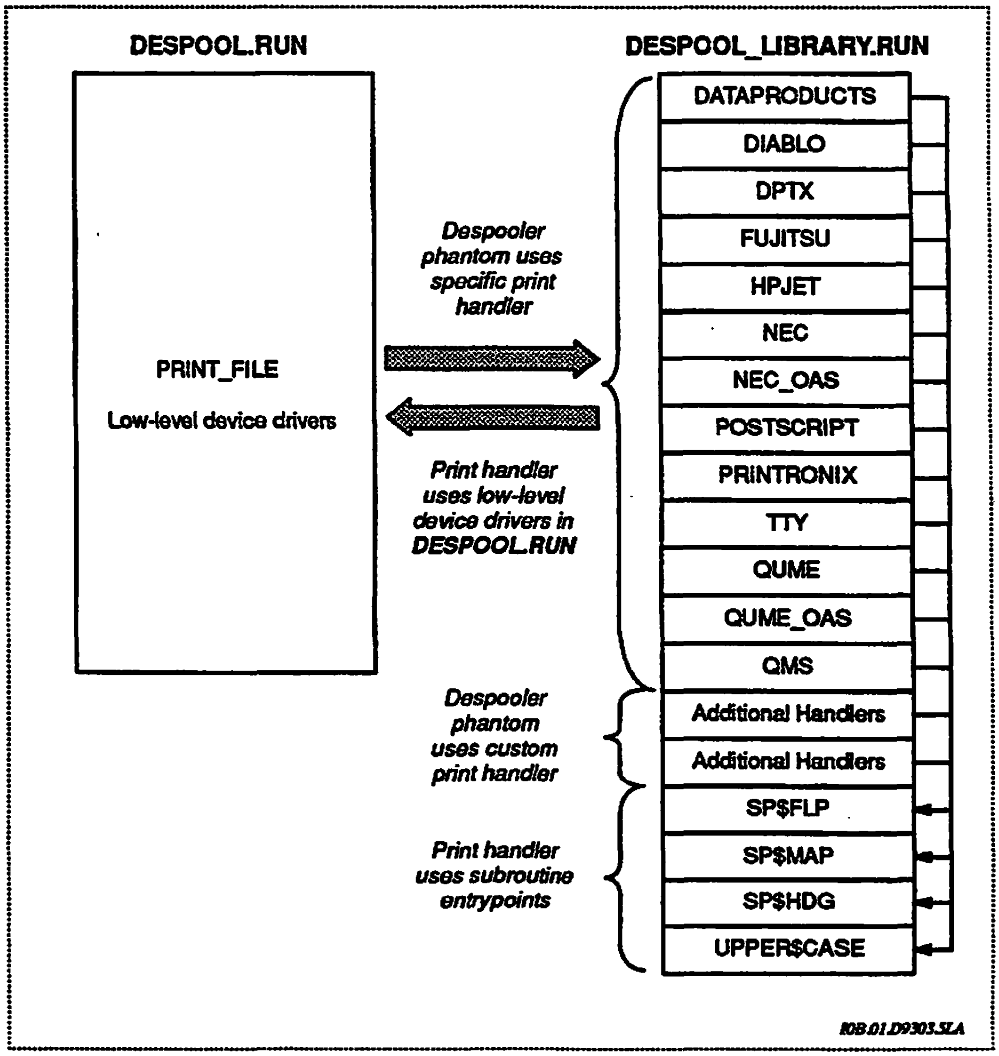

Figure B-1. Using Print Handlers

Using Print Handler Functions

Each print handler must provide code for the functions listed below. The despooler phantom expects all the functions to be present. You must therefore provide an entry for each function, even if the code only performs an immediate return to the calling routine.

The keynames are the names that the despooler phantom uses when it requests an action from a print handler. The keynames are listed below. Note that each keyname ends with an underscore character (_).

S$BEEP_

Sends code or escape sequence to the output device to generate an audible tone.

S$BIN_

Tells the device handler to change to the paper bin that is passed as the argument.

S$END_DOC_

Processes the end of the user document; for example, reverts to standard font to print trailing pages.

S$END_JOB_

Performs end-of-job processing; for example, resets output device characteristics.

S$FONT_

Tells the device handler to change to the font that is passed as the argument.

S$INIT_

Initializes the output device. Usually this means setting characteristics of the output device and the line to which it is connected. The characteristics initialized here are assumed to last until the despooler phantom that controls the device is logged out. Some examples of tasks that may need to be performed by S$INIT are the following: assign a line to the output device, download a special file, and clear font memory. The particular tasks and precise details of the initialization depend on the device and the way that your system is organized.

S$INPUT_

Reads input from a device and returns the input string to the calling program.

S$LARGE_PRINT_

Formats and outputs a message in large characters.

S$MAKE_READY_

Brings the printer to a state of readiness to print, that is, to a state in which the S$START_JOB_ command works.

S$PAGE_

Sends the appropriate code or escape sequence to the device to make it form feed.

S$PORT_

Tells the device handler to change to either portrait or landscape.

S$PRINT_

Accepts a line of print passed in a buffer, translates it to the correct characters for output, passes the output string to the correct line driver, and updates the line count in the PAGCOM data structure.

S$START_COPY_

Sends at the start of each copy of a document. For a single copy it is sent once, for multiple (*n*) copy documents it is sent once per copy (*n* times).

S$START_DOC_

Initializes the start of a user document; for example, selects the correct font, if different from the header page.

S$START_JOB_

Initializes each job; for example, checks that procedures are loaded in a PostScript laser printer.

S$START_PRINT_

Sends a code or escape sequence to make the device continue operation. This key is used after S$STOP_PRINT_ has been called.

S$STATUS_

Reads status from the device, and returns it to calling program.

S$STOP_PRINT_

Sends a code or escape sequence to stop the output device or make it pause.

S$UNREADY_

Returns the device to the state before S$MAKE_READY was executed.

S$VALIDATE_

Not implemented. You must provide a dummy entry and an immediate return for this function.

In addition to providing the above functions, the print handler must convert the input character set to the printer character set. See SP$FLP and SP$MAP subroutines that are described in Calling Interfaces later in this appendix.

Print Handler Keyname Sequences

A print handler requires the keynames in the following sequence:

```
S$INIT_
when a despooler is started
  S$MAKE_READY_
  when work is available
    S$START_JOB_
    print header pages
    S$START_DOC_
      S$START_COPY_
      print all copies, s$start_copy_ per copy
    S$END_DOC_
    print trailer pages
    S$END_JOB_
S$UNREADY_
when no more work available
```

Using Low-level Device Drivers

Print handlers should invoke low-level device/line drivers through the common interface provided by DESPOOL.RUN, irrespective of the type of line or device that is used. They should not contain device specific code. The functions provided by DESPOOL.RUN deal with connecting to and disconnecting from the device, and handle all communication with the device. DESPOOL.RUN provides the following low-level device/line drivers:

- Asynchronous lines
- Parallel lines
- Public Data Networks (PDNs)
- Terminal lines
- Disk files
- IBM devices connected by Distributed Processing Terminal Executive (DPTX)
- Printers connected to a TCP/IP network

The following section describes interfaces in more detail.

Calling Interfaces

This section describes the following calling interfaces from

- DESPOOL.RUN to the print handler
- A print handler to low-level device/line drivers in DESPOOL.RUN
- A print handler to the character conversion routines

Calling Interface From DESPOOL.RUN to the Print Handler

The call (in PL1) is as follows:

```
call print_handler (key, buffer, qcom, envcom, pagcom,
code);
```

print_handler is the name of the print handler; for example QUME or DATAPRODUCTS. The parameters are shown below:

Parameter	*Type*	*Description*
key	fixed bin(15)	Name of print handler function to use.
buffer	char(1024) var	Text buffer.
qcom	1 (structure)	See Using Data Structures section.
envcom	1 (structure)	See Using Data Structures section.
pagcom	1 (structure)	See Using Data Structures section.
code	fixed bin(15)	Returned error code.

Calling Interface From Print Handler to Low-level Device/Line Drivers in DESPOOL.RUN

The call (in PL1) is as follows:

```
call addr(envcom) -> envcom_based.sp$dtr (key, bp, nc,
instr, code);
```

The parameters are shown below:

Parameter	*Type*	*Description*
key	fixed bin(15)	Name of device driver function to use.
bp	ptr	Pointer to user's data buffer.
nc	fixed bin(15)	Number of characters to input or output.
instr	fixed bin(15)	Parallel instruction word.
code	fixed bin(15)	Returned error code.

The keynames and their functions are listed below:

S$ASGN
Assigns device, and initializes device driver.

S$UNAS
Unassigns device, and undoes initialization of device driver.

S$MRDY
Brings device to ready to print state.

S$URDY
Returns device to condition before S$MRDY was executed.

S$STRT
Starts transmission of file to be printed.

S$ENDT
Ends transmission of file to be printed.

S$GETB
Reads buffer from device.

S$PUTB
Writes buffer to device.

S$LSTA
Tests status of line/physical connection.

S$BSTA
Tests status of device buffer.

S$CLIB

Clears input buffer.

S$CLOB

Clears output buffer.

Calling Interface From Print Handler to Character Conversion Routines

Before each call to a low-level device driver, you need to ensure that the output is consistent with the character set of the printer. There are two subroutines provided to achieve this. You can call either SP$FLP or SP$MAP.

SP$FLP converts ECS data to ASCII. This ensures that ASCII is consistently sent from your print handler to the output device. The subroutine can be called from either PL1 or FTN.

In PL1 you should declare the entry point as follows:

```
dcl sp$flp entry (char(*) var) options (shortcall);
```

The PL1 call is as follows:

```
call sp$flp (ECS_data)
```

In FTN you must declare the entry point in a SHORTCALL statement as follows:

```
SHORTCALL SP$FLP
```

You must then use the compiler option –SPECIAL_OPTION (–SPO), otherwise the SHORTCALL statement is not recognized.

The parameter that is passed to SP$FLP is of type INTEGER*2, and must be declared as such. The calling sequence is therefore as follows:

```
INTEGER*2 P8DATA(1025)
     .          .
     .          .
     .          .
CALL SP$FLP (P8DATA)
```

SP$MAP is a general purpose routine capable of converting characters from one character set to another.

In PL1 you should declare the entry point as follows:

```
dcl sp$map entry (ptr, char(1024) var, bin, char(1024) var,
bin, bin);
```

The PL1 call is as follows:

```
call sp$map(envcom.map_table_ptr, source_string,
current_position, target_string,
             current_status, status);
```

The calls and their descriptions are listed below:

Call	*Function*
envcom.map_table_ptr	The pointer to the map as provided in the environment data area.
source_string	The varying string to be converted.
current_position	The point (within source_string) at which to start, or continue, the conversion.
target_string	The converted string.
current_status	Either zero or the last value of status if the source string requires more than one call to complete the conversion.
status	The result of the conversion. If 0, then the conversion is complete. If greater than 0, then an unrecoverable error occurred during the conversion, the other returned variables should be ignored. If less than 0, then the source string could not be completely converted in this call.

It may be necessary to call SP$MAP more than once for a particular line of output. When this happens, status is returned as a negative value. After you output the portion of the converted line, returned in target_string, you should call SP$MAP again. When you call again, the current_position should be the value returned by the last call to SP$MAP, and current_status should be the value of status from the last call.

If status is –1000, then the return buffer, target_string, is not large enough. The handler should output the target_string that has been returned and then hold the print position at the end of it. Thus, the next piece is printed at the end of the first. If status is in the range –1 through –999, then the line contains characters that must be built up by overprinting. The returned target_string should be output and the print position returned to the beginning of the current line.

Auxiliary Configuration Parameters

You can use the AUXILIARY environment directive in the printer environment files to pass environment parameters to your custom print handlers and routines. The environment parameters are read from the environment file, and are inserted into an auxiliary structure in external common memory, from where they are accessed by the custom user routines. Note that the auxiliary structure is not used by the standard despooler, but exists solely for custom routines. The content and use of the environment parameters are the users' responsibility.

For further details of the AUXILIARY environment directive, see Chapter 3, Defining Printer Environments.

Building a Print Handler

The following example illustrates the procedure to build your print handler:

```
OK, BIND
[BIND Rev. T3.1-23.0 Copyright (c) 1991, Prime Computer, Inc.]
: LIBMODE -PROGRAM
: LOAD Routine
: LIBRARY
: DYNT SP$FLP
: DYNT -ALL
BIND COMPLETE
: FILE
OK,
```

Using Data Structures

Three data structures used by DESPOOL.RUN are accessible to the print handler: QCOM, PAGCOM, and ENVCOM. Each structure provides information needed to perform some of the print functions correctly. You must keep some of the data items up to date, but you are free to alter others if necessary.

QCOM

This data structure holds information about the entry in the spool queue. The declaration is as follows:

```
/* START-DESCRIPTION :   This insert file is the interface between the despooler and the
   queue handling routines.
END-DESCRIPTION */
/* START-CODE : */
/* Queue interface common block */
dcl 1 qcom_based based,
       2 flags,
          3 printing_from_pre_rev_21_queue bit(1),
          3 rfu bit(15),
       2 qno bin(31),             /* Queue entry number */
       2 pp_no bin,               /* Poll point number of node or disk */
       2 queue_name char(32) var, /* Node or disk name */
       2 qinf,                    /* Information from the queue entry */
          3 user char(32),        /* Primos user name */
          3 node char(32),        /* Node of origin */
          3 file char(160),       /* File name */
          3 header char(32),      /* File name for header */
          3 alias char(32),       /* User name for header */
          3 copies bin,           /* No of copies */
          3 rqtime char(15),      /* Request time YYMMDDHHMMSSsss */
          3 dtm bin(31),          /* Date/time modified */
          3 flags1,
             4 fortran_format bit(1),  /* Fortran format control */
             4 page_header_supplied bit(1),
             4 numbers bit(1),    /* Generate line numbers */
             4 no_header bit(1),  /* Suppress header page */
             4 no_final_eject bit(1),  /* No page throw at end */
             4 no_format bit(1),  /* No format control */
             4 plot_file bit(1),  /* Plot file */
             4 open_option_used bit(1),
             4 open_option_used bit(1),
                                  /* Set if -OPEN was used */
             4 no_header_format bit(1),
                                  /* -NOPAGEHEADER */
             4 suppress_file_info bit(1),
                                  /* -SFI */
/* SPAR 4024926 */
             4 rfu_2 bit(1),
             4 file_is_att bit(1),
             4 cobol_format bit(1),/* Cobol format control */
             4 no_overprint bit(1),
             4 notify_user bit(1),
             4 no_copy bit(1),
          3 flags2,
             4 from_page bit(1),  /* -FROM option used */
```

```
            4 to_page bit(1),      /* -TO option used */
            4 swo bit(1),          /* -SWO option used */
            4 rfu_1 bit(3),
            4 truncate bit(1),     /* Truncate long lines (-TRUNC) */
            4 non_default_xlate bit(1),
                                   /* Use a non default map */
            4 portrait_flag bit(1),    /* -SET_PORTRAIT */
            4 landscape_flag bit(1),   /* -SET_LANDSCAPE */
            4 paper_bin_flag bit(1),   /* -SET_PAPER_BIN */
            4 font_flag bit(1),    /* -SET_FONT */
            4 rfu_2 bit(4),
         3 ps_proc char(32) var,  /* PostScript procedure name */
         3 raster_size bin,       /* Words per raster in plot file */
         3 user_no bin,           /* User no */
         3 project_id char(32) var,   /* Project name */
         3 file_size bin,         /* Size in records * copies */
         3 no_of_attributes bin,  /* No of attributes following */
         3 attribute(32) char(16),/* Device attributes */
         3 page_header char(160) var, /* Alternate page_header */
         3 user_address char(32), /* From SP$ADDRESS */
         3 from_page_num bin(31), /* From page number */
         3 to_page_num bin(31),   /* To page number */
         3 map_name char(32) var, /* Name of map to be used */
         3 words_printed bin(31), /* Words printed so far */
         3 file_size_copy bin,    /* Copy of file_size */
         3 paper_bin bin,         /* number of bin to use */
         3 font char(32);         /* name of font to use */

/* END-CODE */

%list ;
```

PAGCOM

This data structure holds information that the print handler needs to format each page correctly; it also records the current state of the page that is being printed. Your print handler must keep the line counts up to date. The declaration is as follows:

```
/* START-DESCRIPTION :
    description to be inserted
   END-DESCRIPTION */
/* --- Physical definition for format modes; ie the equivalent values that */
/*    would appear in standard print file, after the ^001/^201 control code. */
%Replace no_fmt_ by 0,
         fortran_fmt_ by 1,
         cobol_fmt_ by 2,
         paginate_fmt_ by 3,
         no_header_fmt_ by 4,
         raster_fmt_ by 5,             /* Aka "binary" mode. */
         vector_fmt_ by 6;             /* Aka "high-bit-off" mode. */
%Replace max_header_ln_ by 160,
         max_tab_stops_ by 128;
%Replace pagcom_size_ by 237;          /* Size of PAGCOM in words. */
dcl 1 pagcom_based based,
       2 print_mode bin,               /* Print format mode. (physical). */
       2 chars_pline bin,              /* Characters per line. (physical). */
       2 char_count bin,               /* Character position on current line.
                                          */
       2 total_char_count bin(31),     /* Total number of characters printed.
                                          */
       2 lines_ppage bin,              /* Lines per page. (physical). */
       2 line_count bin,               /* Current line number on page. */
       2 virt_line_count bin,          /* Virtual line number on page. */
       2 total_line_count bin(31),     /* Total number of lines printed. */
       2 page_count bin,               /* Current page number of this copy. */
       2 virt_page_count bin,          /* Virtual page number of this copy. */
       2 total_page_count bin(31),     /* Total number of pages printed. */
       2 header char(max_header_ln_) var,
                                       /* Current header message. */
       2 left_margin bin,              /* Left margin. (logical). */
       2 right_margin bin,             /* Right margin. (logical). */
       2 top_margin bin,               /* Top margin. (logical). */
       2 bottom_margin bin,            /* Bottom margin. (logical). */
       2 cpi bin,                      /* Characters Per Inch. */
       2 lpi bin,                      /* Lines Per Inch. */
       2 tab_interval bin,             /* Tab Interval. */
       2 num_tab_stops bin,            /* Number of tab stop positions. */
       2 tab_stops(max_tab_stops_) bin,
                                       /* Top Stop positions. */
       2 pad_char char,                /* Padding char, space/null/0 etc. */
       2 fmt_flags,                    /* Extra Format flags. */
          3 append_lf bit(1),          /* Append LF to print line. */
          3 line_wrap bit(1),          /* Wrap line after right margin
                                          exceeded. */
                                       /* '1'b = wrap, '0'b = truncate. */
          3 pad_char_flag bit(1),      /* '1'b = pad char specified in option
                                          */
                                       /* '0'b = use pad_char in driver */
          3 mbz bit(5),                /* Reserved for future use */
                                       /* Pre-rev21 sets mbz to zeros */
       2 fs_lin_count bin(31),         /* -LNUM line number */
       2 file_position bin(31),        /* Saved file position for Backup. */
       2 rev_22_bits,
          3 print_it bit(1),           /* true if printing this part of file */
          3 to_point_reached bit(1),   /* true if -TO page number completed */
       3 override_from_to bit(1),   /* temporarily ignoring -FROM/-TO */
          3 rfu bit(13);               /* reserved for future use */
%list;
/* --- End insert file: page_com.ins.plp */
```

ENVCOM

This data structure provides the print handler with information about the despooler environment. The declaration is as follows:

```
/* --- Print handler classes. */
%Replace BUILTIN_ by 1,
         DYNAMIC_ by 2,
         UNKNOWN_ by 3;
/* ********************************************************************** */
/* --- Table of builtin handler names. */
%Replace PRINTRONIX_ by 1,
         QUME_ by 2,
         DIABLO_ by 3,
         MAX_TBL_SZ_ by 1;
/* dcl int_hdlr_tbl(MAX_TBL_SZ_) char(32) var static init('PRINTRONIX', 'QUME',
              'DIABLO'); */
dcl int_hdlr_tbl(MAX_TBL_SZ_) char(32) var static init('');
/* --- Builtin print handler entry dcls (belong here). */
/*
   dcl printronix entry(bin, char(*) var, 1, 1, 1, bin),
    qume entry(bin, char(*) var, 1, 1, 1, bin),
    diablo entry(bin, char(*) var, 1, 1, 1, bin);
*/
/* ********************************************************************** */
%Replace MAX_EVFU_ by 143;             /* Max size of evfu storage. */
%Replace MANUAL_ by 1,                 /* Stationery feeding types. */
         CUTSHEET_ by 2,
         CONTINUOUS_ by 3;
%Replace LOG_OFF_ by 0,                /* ENVCOM.LOG_TYPE values */
         LOG_ON_ by 1,
         LOG_SAVE_ by 2;
/* ****************************** Envcom ****************************** */
dcl 1 envcom_based based,
       2 handler_name char(32) var,    /* Printer handler name. */
       2 print_handler entry(bin, char(*) var, 1, 1, 1, bin) variable,
                                       /* The printer handler to use. */
       2 big_letters entry(char(*) var, (7) char(*) var, char(*)) variable,
       2 read_evfu entry(1) variable,
       2 page_throw entry() variable,
       2 sp$dtr entry(bin, ptr, bin, bin, bin) variable,
       2 device_type bin,              /* Low level device driver index. */
       2 comms_ptr ptr,                /* Address device driver data area. */
       2 gen_flags,                    /* No generic class for these bits. */
          3 evfu_sw bit(1),            /* Please dll evfu file (below). */
          3 font_sw bit(1),            /* Ditto for font. */
          3 dict_sw bit(1),            /* Ditto for dict. */
          3 crlf_sw bit(1),            /* Auto line feed switch. */
          3 uc_sw bit(1),              /* Uppercase switch. */
          3 ksr_sw bit(1),             /* Defines printer to be KSR type. */
          3 warning_sw bit(1),         /* En/Dis-able warning banner. */
          3 evfu_disabled_sw bit(1),   /* Evfu not enabled in hardware. */
          3 rbc_sw bit(1),             /* Use rolling banner char for big
                                          letters. */
          3 facing_banner_sw bit(1),   /* Select upwards facing banner. */
          3 banner_eject bit(1),       /* Eject before printing banner. */
          3 suppress_file_info bit(1), /* Suppress file info on banner page. */
          3 no_eject_sw bit(1),        /* Unconditionally honor -NOJ. */
          3 mbz bit(3),                /* Must be zero. */
       2 evfu_file char(128) var,      /* EVFU pathname. */
       2 font_file char(128) var,      /* Font pathname. */
       2 dict_file char(128) var,      /* Dictionary pathname (PS printers). */
       2 message_text char(160) var,   /* Msg to display on header banner. */
       2 printer_info(2) char(160) var,
                                       /* Optional printer info for hdr page.*/
       2 header_pages bin,             /* Number of header pages to print. */
       2 trailer_pages bin,            /* Number of trailer pages to print. */
       2 evfu(MAX_EVFU_) bin,          /* Storage for evfu after evfu_read. */
       2 stationery_feed_type bin,     /* Manual, Cut-Sheet or Continuous. */
       2 despool_fatal_error label,    /* Fatal errors trap to here. */
       2 big_char char,                /* Character used for big letters. */
       2 old_pspass_now_RFU char(6),   /* PostScript password */
       2 log_type bin,                 /* Mode of log file */
       2 trailer_perf_lines bin,       /* overprint lines on trl perfs */
       2 trailer_perf_char_flag bit(1) aligned,
                                       /* if -PERF_CHAR option used */
       2 trailer_perf_char char,       /* character for perf overprints */
       2 printronix_pi bit(1) aligned, /* Printronix paper instruction */
       2 swo_wait_time bin,            /* Time to wait on EOF with -SWO */
       2 swo_times_to_check bin,       /* No times to loop on -SWO & EOF */
       2 psi_increment bin(31),        /* Word increment for %age printed */
       2 Map_table_ptr ptr,            /* Pointer to current mapping table */
       2 sdb ptr,                      /* Pointer to despooler SIT context */
       2 aux_addr ptr,                 /* Pointer to auxiliary structure */
       2 env_name char(16) var,        /* environment name */
       2 evfu_ptr ptr,                 /* pointer to EVFU code */
       2 no_lf_after_ff bit(1) aligned,
                                       /* true if LF immed after FF ignored */
       2 pspass char(32),              /* PostScript password */
       2 nif_flag bit(1) aligned,      /* no initial form-feed flag */
       2 wait_time bin,                /* max shutdown time for async */
       2 data_file_unit_number bin,    /* unit number for data file */
       2 sequence bit(1) aligned,      /* sequence output filename */
       2 plot_page_def char(80) var,   /* define a page break in plot mode */
       2 hp_delay bin,                 /* define delay time (for HP LaserJet) */
       2 gen_flags_2,
         3 post_check bit(1),          /* check print job for PostScript */
         3 post_override bit(1),       /* override print no_format selection/*
         3 reserve_2 bit(14),          /* reserved for future growth */
                                       /* based on line one of print job. */
       2 prop_user_no bin;             /* User no of process that used PROP.*/
%list ;
/* --- End insert file: envcom.ins.plp. */
```

Example of a Print Handler

An example of a print handler is shown below. It can serve as a template when you write a print handler of your own. The language used is PLP. Please note that the Printronix handler that is actually supplied as part of your Spooler subsystem may differ in some respects from the version listed here.

```
/* PRINTRONIX.PLP,SPOOLSRC>DESPOOL>HANDLERS,SPOOLER PROJECT, 11/22/91
   PRINTRONIX printer handler
   Copyright (c) 1986, Prime Computer, Inc., Natick, Ma 01760 *//* TITLE :
PRINTRONIX - PRINTRONIX printer handler */

PRINTRONIX:
   procedure (key, text, qcom, envcom, pagcom, code);
$Insert spoolsrc>insert>envcom.ins.plp

$Insert spoolsrc>insert>qcom.ins.plp

$Insert spoolsrc>insert>page_com.ins.plp

$Insert spoolsrc>insert>constants.ins.plp

$Insert spoolsrc>insert>device_info.ins.plp

$Insert spoolsrc>insert>cc_defns.ins.plp

$Insert spoolsrc>insert>s$keys.ins.plp

$Insert syscom>keys.ins.pl1

$Insert syscom>errd.ins.pl1

$Insert intcom*>si$keys.ins.pl1

$Insert intcom*>si$errs.ins.pl1

$Insert spoolsrc>insert>spooler_msgs.ins.pl1

   dcl key bin;                        /* action key */
   dcl text char(1024) var;            /* text to process */
   dcl 1 qcom like qcom_based;         /* queue common */
   dcl 1 envcom like envcom_based;     /* environment common */
   dcl 1 pagcom like pagcom_based;     /* page common */
   dcl code bin;                       /* standard error code */

   dcl sp$map entry(ptr, ptr, char(*) var, bin, char(1024) var, bin, bin),
       ioa$ entry options(variable);
/* No longer required
   %Replace bel_ by '07'b4,               /* Chars with Top bit off. ASCII 8. */
         ff_ by '0C'b4,
         lf_ by '0A'b4,
         cr_ by '0D'b4,
         esc_ by '1B'b4,
         sub_ by '1A'b4,
         can_ by '18'b4;

%Replace p8_bel_ by '87'b4,            /* Chars with Top bit on. PR1ME 8. */
         p8_ff_ by '8C'b4,
         p8_lf_ by '8A'b4,
         p8_cr_ by '8D'b4,
         p8_esc_ by '8B'b4,
         p8_nul_ by '80'b4,
         p8_can_ by '98'b4;

%Replace def_pad_char_ by 'A0'b4;      /* Default value of pad character */

%Replace ascii_def_pad_char_ by '20'b4;
                                       /* Ascii (flipped) value */

   dcl pad_char char static internal;
   dcl large_print(7) char(132) var;
   dcl vs$ char(512) var static external;
   dcl Work_string char(1024) var;
   dcl max_big_letters bin;
   dcl i bin;
   dcl inst bin;
   dcl module char(32) var static init('PRINTRONIX');
   dcl 1 vs based,
          2 len bin,
          2 chars char(1024);
   dcl 1 ctl based,
          2 len bin,
          2 char1 bit(8) unaligned,
          2 char2 bit(8) unaligned,
          2 data(1022) bit(8) unaligned;
   dcl junk_string char(1024) var;
   dcl junk bin;

      code = 0;

      select(key);
      when(s$init_)
         do;
            envcom.printer_info(1) = '';
            envcom.printer_info(2) = '';
            envcom.banner_eject = '0'b;
            envcom.evfu_ptr = addr(vs$);

/* check on whether we're to use the default pad character */

            if pagcom.pad_char_flag
            then                       /* overwrite */
               pad_char = pagcom.pad_char;
            else
               pad_char = chr$(def_pad_char_);
                                       /* the default */

/* check if an evfu format file is present */

            if envcom.evfu_sw & pagcom.lines_ppage <= max_evfu_
            then
               call addr(envcom) -> envcom_based.read_evfu(envcom);
            else
               do;
                  envcom.evfu(1) = 1;
                  do i = 2 to max_evfu_;
                     envcom.evfu(i) = 0;
                  end;
               end;

/* assign device */

            call addr(envcom) -> envcom_based.sp$dtr(s$asgn, null(), 0, 0,
                      code);
            if code ^= 0
            then
               return;

            call send_evfu;

/* disk_dbg code */
           if envcom.device_type = SP$DISK_
              & envcom.sequence
           then
             call addr(envcom) -> envcom_based.sp$dtr(S$ENDT, null(), 0, 0,
                       code);
         end;
      when(s$print_)
         call proc_print(text);
      when(s$validate_)
         do;
         end;
      when(s$beep_)
         do;
         end;
      when(s$input_)
         text = '';
      when(s$stop_print_)
         code = e$null;
      when(s$status_)
         call addr(envcom) -> envcom_based.sp$dtr(s$1sta, null(), 0, 0, code);
      when(s$page_)
```

```
   do;
      select(envcom.device_type);
      when(sp$para_)
         call addr(envcom) -> envcom_based.sp$dtr(s$putb, null(), 0,
                   '20014'b3, code);
      otherwise
         do;
            Work_string = CHR$(p8_ff_);
            call MAP_CC_AND_OUTPUT(Work_string, Code);
         end;
      end;

      if length(text) > 0
      then
         call print(text);
   end;
when(s$start_job_)
   do;
     if envcom.device_type = SP$TCP_IP_
     then
       call send_evfu;
     call addr(envcom) -> envcom_based.sp$dtr(s$strt, null(), 0, 0, code);
   end;
when(s$start_doc_)
   do;
   end;
when(s$end_doc_)
   do;
   end;
when(s$end_job_)
   call addr(envcom) -> envcom_based.sp$dtr(s$endt, null(), 0, 0, code);
when(s$large_print_)
   do;
      max_big_letters = divide(pagcom.chars_pline, 6, 15);
      if max_big_letters > 22
      then
         max_big_letters = 22;

      if length(text) > max_big_letters
      then
         text = substr(text, 1, max_big_letters);

      call skip_lines(2);
      call addr(envcom) -> envcom_based.big_letters(text, large_print,
                 envcom.big_char);

      do i = 1 to 7;
         call print((large_print(i)));
      end;

      call skip_lines(2);
```

```
         end;
      when(s$make_ready_)                /* ready printer for printing */
         do;
            call addr(envcom) -> envcom_based.sp$dtr(s$mrdy, null(), 0, 0,
                      code);
         end;
      when(s$unready_)                   /* put printer back to idle */
         do;
            call addr(envcom) -> envcom_based.sp$dtr(s$urdy, null(), 0, 0,
         end;

/* disk_dbg code */

      when (S$ENDT)
         do;
            call addr(envcom) -> envcom_based.sp$dtr(s$endt, null(), 0, 0,
                      code);
         end;

      when(s$port)                       /* paper orientation */
         do;

/* this handler does not support paper orientations */

            code = e$nsuc;
         end;
      when(s$font)                       /* font */
         do;

/* this handler does not support fonts */

            code = e$nsuc;
         end;

      when(s$bin)                        /* paper_bin */
         do;

/* this handler does not support paper bins */

            code = e$nsuc;
         end;
                                         /* start next copy */
      when(s$start_copy_)
         do;
         end;

      otherwise                          /* unrecognized key */

         code = E$BKEY;
```

```
        end;
        return;


   /* ============================================================== SEND_EVFU */
   SEND_EVFU:
        procedure;

        dcl vs$ char(512) var static external;
        dcl inst bin;

   /* downline load the evfu stuff */

          if ^envcom.evfu_disabled_sw
          then
            select(envcom.device_type);
            when(sp$para_)
            do;
              if envcom.printronix_pi
              then
              do;
                inst = '20156'b3;
                vs$ = chr$(inst);
                call addr(envcom) -> envcom_based.sp$dtr(s$putb, null
                          (), 0, inst, code);

                junk = pagcom.lines_ppage;
                if junk > max_evfu_
                then
                  junk = max_evfu_;
                do i = 1 to junk;

                  if envcom.evfu(i) = 0
                  then
                  do;
                    inst = '20115'b3;
                    vs$ = vs$ || chr$(inst);
                  end;
                  else
                  do;
                    inst = '20077'b3 + envcom.evfu(i);
                    vs$ = vs$ || chr$(inst);
                  end;
                  call addr(envcom) -> envcom_based.sp$dtr(s$putb,
                            null(), 0, inst, code);

                end;

                inst = '20157'b3;

                vs$ = vs$ || chr$(inst);
```

```
    call addr(envcom) -> envcom_based.sp$dtr(s$putb, null
              (), 0, inst, code);
  end;
  else
     do;                  /* not Paper_Instruction type printronix
                             */
       inst = '20036'b3;

       vs$ = vs$ || chr$(inst);
       call addr(envcom) -> envcom_based.sp$dtr(s$putb, null
                    (), 0, inst, code);
       junk = pagcom.lines_ppage;
       if junk > max_evfu_
       then
          junk = max_evfu_;
       do i = 1 to junk;
          if envcom.evfu(i) = 0
          then
             do;
               inst = '20035'b3;

               vs$ = vs$ || chr$(inst);
             end;
          else
             do;
               inst = '20017'b3 + envcom.evfu(i);

               vs$ = vs$ || chr$(inst);
             end;
          call addr(envcom) -> envcom_based.sp$dtr(s$putb,
                               null(), 0, inst, code);
       end;
       inst = '20037'b3;
       vs$ = vs$ || chr$(inst);
       call addr(envcom) -> envcom_based.sp$dtr(s$putb, null
                            (), 0, inst, code);
     end;
  end;

when(sp$mpc4000_)

/*
 * Download EVFU table to the Legend board
 */

if envcom.printronix_pi
then
do;
  inst = '20156'b3;
  vs$ = chr$(inst);
  call addr(envcom) -> envcom_based.sp$dtr(s$putb, null
            (), 0, inst, code);
  junk = pagcom.lines_ppage;
  if junk > max_evfu_
  then
    junk = max_evfu_;
  do i = 1 to junk;             if envcom.evfu(i) = 0
    then
    do;
      inst = '20015'b3;
      vs$ = vs$ || chr$(inst);
    end;
    else
    do;
      inst = '20000'b3 + envcom.evfu(i);
      vs$ = vs$ || chr$(inst);
    end;
    call addr(envcom) -> envcom_based.sp$dtr(s$putb,
              null(), 0, inst, code);
  end;

  inst = '20157'b3;
  vs$ = vs$ || chr$(inst);
  call addr(envcom) -> envcom_based.sp$dtr(s$putb, null
            (), 0, inst, code);
end;

else
do;                 /* not Paper_Instruction type printronix
                                   */
  inst = '20036'b3;

  vs$ = vs$ || chr$(inst);
  call addr(envcom) -> envcom_based.sp$dtr(s$putb, null
            (), 0, inst, code);
  junk = pagcom.lines_ppage;
  if junk > max_evfu_
  then
    junk = max_evfu_;
  do i = 1 to junk;
    if envcom.evfu(i) = 0
    then
    do;
      inst = '20035'b3;

      vs$ = vs$ || chr$(inst);
    end;
    else
    do;
      inst = '20017'b3 + envcom.evfu(i);

      vs$ = vs$ || chr$(inst);
```

```
             end;
             call addr(envcom) -> envcom_based.sp$dtr(s$putb,
                       null(), 0, inst, code);

           end;
           inst = '20037'b3;

           vs$ = vs$ || chr$(inst);
           call addr(envcom) -> envcom_based.sp$dtr(s$putb, null
                      (), 0, inst, code);
         end;

         otherwise               /* serial */
         do;
           vs$ = chr$('36'b3);
           junk = pagcom.lines_ppage;
           if junk > max_evfu_
           then
             junk = max_evfu_;
           do i = 1 to junk;
             if envcom.evfu(i) = 0
             then
               vs$ = vs$ || chr$('35'b3);
             else
             do;
               inst = envcom.evfu(i) + '17'b3;
               vs$ = vs$ || chr$(inst);
             end;
           end;
           vs$ = vs$ || chr$('37'b3);
           call addr(envcom) -> envcom_based.sp$dtr(s$putb, addr(addr
                      (vs$) -> vs.chars), length(vs$), 0, code);
         end;
       end;
     end;


/* ========================================================== PROC_PRINT */
PROC_PRINT:
      procedure (text);

      dcl text char(1024) var;
      dcl channel bin;

         if length(text) ^= 0
         then
            select(addr(text) -> ctl.char1);

            when('02'b4)                /* found a control record */
               do;
                  call control(text);
                  if length(text) < 1
                  then
                     return;
               end;
            when('03'b4)               /* found a skip to EVFU channel */
               if ^envcom.evfu_disabled_sw
                                       /* to stop ctrl char being stripped */
               then
                  do;
                     if length(text) = 1
                     then
                        channel = 1;   /* set default */
                     else
                        channel = addr(text) -> ctl.char2 & '7F'b4;

                     call skip_channel(channel);

                     if length(text) > 2
                     then
                        text = substr(text, 3);
                     else
                        return;
                  end;

                  end;

            end;

         call print(text);

      end;


/* ========================================================= PRINT */
PRINT:
      procedure (text);

      dcl text char(1024) var;

      dcl code bin;

      dcl Required bit(1);

      dcl Current_position bin,
          Output_string char(1024) var,
          Last_status bin,
          Status bin;
      dcl scode bin;

      dcl sp$flp entry(char(*) var) options(shortcall),
          upper$case entry(char(*) var) options(shortcall),
          si$g entry(pointer, fixed bin(15), pointer, fixed bin(15), bit(1),
                    fixed bin(15));

         if envcom.uc_sw              /* convert to uppercase if necessary */
         then
            call upper$case(text);

/* print the data */

         select(envcom.device_type);
         when(sp$para_)
            do;

               Current_position = 0;
               Status = 0;
               Required = '1'b;
               if length(text) ^= 0
               then                   /* only if any text there */
                  do while(Required);

                     Last_status = Status;
                     Status = 0;

                     /* Map the text ready for output. */

                     if envcom.map_table_ptr = null()
                     then             /* don't call sp$map, do it
                                         ourselves */

                        do;
                           output_string = text;
                           call sp$flp(output_string);
                        end;
                     else
                        call SP$MAP(Envcom.sdb, Envcom.Map_table_ptr, Text,
                                  Current_position, Output_string, Last_status,
                                  Status);

                     if Status <= -1000
                     then

                        /* somebody has defined a file with 1:N
                           substitutions in it and it has overflowed the buffer
                           */

                        call si$g(envcom.sdb, handler_buffer_overflow_, null(),
                                  0, '1'b, scode);

                     else
                        if status > 0
                        then
                           do;
                              code = status;
                              return;
                           end;

/* make sure it's a whole number of words */

                     if mod(length(Output_string), 2) ^= 0
                     then
                        Output_string = Output_string || pad_char;

                     call addr(envcom) -> envcom_based.sp$dtr(s$putb, addr(addr
                               (Output_string) -> vs.chars), length
                               (Output_string), '40000'b3, code);
                                       /* Output the line but do not send CR/LF
                                          */

                     required = ^(status = 0);

                  end;

/* send appropriate line terminator */

               if pagcom.append_lf
               then
                  call addr(envcom) -> envcom_based.sp$dtr(s$putb, null(), 0,
                            '20012'b3, code);

            end;

         when(sp$plot_)
            do;

               if length(text)  ^= 0
               then                    /* only if any text there */
                  do;

/* append appropriate line terminator if not already present.  Also
   make the length of the output line an even number of characters. */

                     if mod(length(text),2) ^= 0
                     then do;
                       if substr(text,length(text),1) ^= CHR$(p8_cr_)
                       then
                         text = text || CHR$(p8_cr_); /* append carriage return
*/
                       else do;
                         substr(text,length(text),1) = pad_char; /* replace carr
iage return with pad char */
                         text = text || CHR$(p8_cr_); /* append carriage return
```

```
*/
                       end;
                     end;
                     else do; /* even length already */
                       if substr(text,length(text), 1) ^= CHR$(p8_cr_)
                       then        /* append with pad char and carriage return
*/
                         text = text || pad_char || CHR$(p8_cr_);
                     end;

/* Map the output and then print it. */

                     Current_position = 0;
                     Status = 0;
                     Required = '1'b;
                     do while(required);
                         Last_status = status;
                          Status = 0;

/* Map the text ready for output. */

                        if envcom.map_table_ptr = null()
                        then           /* don't call sp$map, do it
                                          ourselves */

                           do;
                              output_string = text;
                              call sp$flp(output_string);
                           end;
                        else
                           call SP$MAP(Envcom.sdb, Envcom.Map_table_ptr, Text,
                                     Current_position, Output_string,
                                     Last_status, Status);

                        if Status > 0
                        then
                           do;
                              code = status;
                              return;
                           end;

/* make sure it's a whole number of words */

                        if mod(length(Output_string), 2) ^= 0
                        then
                           Output_string = Output_string || pad_char;

                        call addr(envcom) -> envcom_based.sp$dtr(s$putb, ADDR
                                   (ADDR(Output_string) -> vs.chars), ADDR
                                   (Output_string) -> vs.len, 0, code);
                        if Status = 0 | Code ^= 0
                        then
                           Required = '0'b;
                        else
                           if Status > -1000
                           then
                              do;      /* Force print position back to first
                                          column */

                                 Work_string = CHR$(p8_cr_);
                                 call MAP_CC_AND_OUTPUT(Work_string, Code);

                              end;

                     end;
                  end;                 /* end only if something there loop */

               if envcom.crlf_sw & pagcom.append_lf
               then
                  do;
                     if length(text) = 0
                     then
                        do;
                           Work_string = chr$(p8_cr_);
                           call MAP_CC_AND_OUTPUT(Work_string, code);
                        end;
                  end;

            end;

         otherwise
            do;

               if length(text) ^= 0
               then                    /* only if something there */
                  do;

/* Append appropriate line terminator if not already present.  Also
   make the length of the output line an even number of characters. */

                     if mod(length(text),2) ^= 0
                     then do;
                       if substr(text,length(text),1) ^= CHR$(p8_cr_)
                       then
                         text = text || CHR$(p8_cr_); /* append carriage return
*/
                       else do;
                         substr(text,length(text),1) = pad_char; /* replace carr
iage return with pad char. */
                         text = text || CHR$(p8_cr_); /* append carriage return
*/
```

```
                        end;
                      end;
                      else do; /* even length already */
                        if substr(text,length(text), 1) ^= CHR$(p8_cr_)
                        then        /* append with pad char and carriage return
*/
                          text = text || pad_char || CHR$(p8_cr_);
                      end;

/* --- Map the output and then print it. */

                      Current_position = 0;
                      Status = 0;
                      Required = '1'b;
                      do while(Required);

                         Last_status = Status;
                         Status = 0;
                         if envcom.map_table_ptr = null()
                         then           /* don't call sp$map, do it
                                           ourselves */
                            do;
                               output_string = text;
                               call sp$flp(output_string);
                            end;
                         else
                            call SP$MAP(Envcom.sdb, Envcom.Map_table_ptr, Text,
                                      Current_position, Output_string,
                                      Last_status, Status);

                         if Status > 0
                         then
                            do;
                               code = status;
                               return;
                            end;

                         call addr(envcom) -> envcom_based.sp$dtr(s$putb, ADDR
                                    (ADDR(Output_string) -> vs.chars), ADDR
                                    (Output_string) -> vs.len, 0, code);

                         if Status = 0 | Code ^= 0
                         then
                            Required = '0'b;
                         else
                            if Status > -1000
                            then
                               do;       /* Force print position back to first
                                            column */

                                  Work_string = CHR$(p8_cr_);
```

```
                                 call MAP_CC_AND_OUTPUT(Work_string, Code);

                              end;

                     end;
                  end;                 /* end only if something there loop */

               if envcom.crlf_sw & pagcom.append_lf
               then
                  do;

                     Work_string = chr$(p8_cr_) || chr$(p8_lf_);
                     call MAP_CC_AND_OUTPUT(Work_string, code);
                  end;

            end;

         end;

/* increment line count */

         if pagcom.append_lf
         then
            do;
               pagcom.virt_line_count = pagcom.virt_line_count + 1;
               if pagcom.print_it | pagcom.override_from_to
               then                    /* only increment if printing */
                  pagcom.line_count = pagcom.line_count + 1;
            end;

      end;


/* ========================================================= CONTROL */

CONTROL:
      procedure (text);

      dcl text char(1024) var;

      dcl lines bin;

         if length(text) = 1           /* ignore invalid control record */
         then
            return;

         select(addr(text) -> ctl.char2);
         when(set_lpi_, set_cpi_, set_lpp_, set_cpl_, set_tab_int_, set_tabs_)
            if length(text) > 3
            then
```

```
                text = substr(text, 4);
            else
                text = '';
        when(clear_tabs_, set_uc_, clear_uc_)
            if length(text) > 2
            then
                text = substr(text, 3);
            else
                text = '';
        when(set_channel_)
            do;
                i = addr(text) -> ctl.data(2) & '7F'b4;
                if i <= max_evfu_
                then
                    envcom.evfu(i) = addr(text) -> ctl.data(1) & '7F'b4;
                if length(text) > 4
                then
                    text = substr(text, 5);
                else
                    text = '';
            end;
        when(clear_channel_)
            do;
                envcom.evfu(1) = 1;
                do i = 2 to max_evfu_;
                    envcom.evfu(i) = 0;
                end;
                if length(text) > 2
                then
                    text = substr(text, 3);
                else
                    text = '';
            end;
        when(skip_lines_)                  /* skip n lines */
            do;
                if length(text) = 2        /* no default */
                then
                    text = '';
                else
                    do;
                        lines = addr(text) -> ctl.data(1) & '7F'b4;
                        call skip_lines(lines);
                        if length(text) > 3
                        then
                            text = substr(text, 4);
                        else
                            text = '';
                    end;
            end;

        end;
```

```
      end;

/* ============================================================ SKIP_LINES */

SKIP_LINES:
      procedure (lines);

      dcl lines bin;

      dcl n15 bin;
      dcl r15 bin;

         if lines < 1                      /* nothing to do if zero or negative */
         then
            return;

         if lines > 256
         then
            lines = 256;

         select(envcom.device_type);
         when(sp$para_)

            do i = 1 to lines;
               call addr(envcom) -> envcom_based.sp$dtr(s$putb, null(), 0,
                         '20012'b3, code);
            end;
         otherwise                         /* serial */
            do;
               Work_string = '';
               do i = 1 to lines;
                  Work_string = Work_string || CHR$(p8_lf_);
               end;

               call MAP_CC_AND_OUTPUT(Work_string, code);
            end;
         end;

         pagcom.virt_line_count = pagcom.virt_line_count + lines;

      end;

/* ========================================================= SKIP_CHANNEL */

SKIP_CHANNEL:
      procedure (channel);

      dcl channel bin;

      dcl inst bin;
      dcl i bin;
      dcl lines bin;
      dcl found bin;
      dcl ff char(2);
      dcl ch char(2);
      dcl Work_string char(1024) var;
      dcl Output_string char(1024) var;

/* find out where a skip to this channel will take us
   so that line count may be kept up to date */

         found = 0;

         do i = pagcom.virt_line_count to max_evfu_ while(found = 0);
            if envcom.evfu(i) = channel
            then
               found = i;
         end;

         do i = 1 to pagcom.virt_line_count while(found = 0);
            if envcom.evfu(i) = channel
            then
               found = i;
         end;

/* if channel does not exist just ignore the skip */

         if found = 0
         then
            return;

         if ^envcom.evfu_disabled_sw
         then
            select(envcom.device_type);
            when(sp$para_)                /* parallel */
               if envcom.printronix_pi
               then
                  do;
                     inst = '20077'b3 + channel;
                     call addr(envcom) -> envcom_based.sp$dtr(s$putb, null(),
                               0, inst, code);
                  end;
               else
                  do;                     /* not Paper_Instruction type printronix
                                             */
                     inst = '20017'b3 + channel;
                     call addr(envcom) -> envcom_based.sp$dtr(s$putb, null(),
                               0, inst, code);
                  end;

            when(sp$mpc4000_)                /* parallel */
```

```
               if envcom.printronix_pi
               then
                  do;
                     inst = '20000'b3 + channel;
                     call addr(envcom) -> envcom_based.sp$dtr(s$putb, null(),
                                0, inst, code);
                  end;
               else
                  do;                   /* not Paper_Instruction type printronix
                                           */
                     inst = '20017'b3 + channel;
                     call addr(envcom) -> envcom_based.sp$dtr(s$putb, null(),
                                0, inst, code);
                  end;
            otherwise                   /* serial */
               do;
                  inst = '17'b3 + channel;
                  ch = chr$(inst) || chr$(def_pad_char_);
                  call addr(envcom) -> envcom_based.sp$dtr(s$putb, addr(ch), 1,
                            0, code);
                  pagcom.virt_line_count = found;
               end;
            end;
         else
            do;
               if found < pagcom.virt_line_count
               then
                  do;
                     select(envcom.device_type);
                     when(sp$para_, sp$mpc4000_)
                        call addr(envcom) -> envcom_based.sp$dtr(s$putb, null
                                   (), 0, '20014'b3, code);
                     otherwise
                        do;
                           Work_string = CHR$(p8_ff_);
                           call MAP_CC_AND_OUTPUT(Work_string, Code);
                        end;
                     end;

                     call addr(envcom) -> envcom_based.page_throw;
                  end;

               lines = found - pagcom.virt_line_count - 1;

               call skip_lines(lines);

            end;

      end;

MAP_CC_AND_OUTPUT:
      procedure (In_string, Code);

/* This routine maps and outputs control code sequences to the
   printer. It only works for serial lines. */

/* It DOES NOT cope with substitutions which require the printing of
    more than one line (Overprinting substitutions). */

      dcl In_string char(1024) var,
          Temp_string char(1024) var,
          Max bin,
          I bin,
          Length_input bin,
          Output_string char(1024) var,
          Code bin,
          Status bin,
          Last_status bin,
          Current_position bin;
      dcl sp$flp entry(char(*) var) options(shortcall);

         Length_input = LENGTH(In_string);
         Max = DIVIDE(Length_input - 1, 32, 15);

         do I = 0 to Max;
            if (I + 1) * 32 < Length_input
            then
               Temp_string = SUBSTR(In_string, I * 32 + 1, 32);
            else

/* this used to try and take length_input characters, causing a stringrange
   error every time,  just take the rest of the string */

               temp_string = substr(in_string, i * 32 + 1);

            Current_position = 0;
            Status = 0;
            Last_status = 0;

            /* Map the text ready for output. */

            if envcom.map_table_ptr = null()
            then                       /* don't call sp$map, do it
                                          ourselves */
               do;
                  output_string = temp_string;
                  call sp$flp(output_string);
               end;
            else
               call SP$MAP(Envcom.sdb, Envcom.Map_table_ptr, Temp_string,
                        Current_position, Output_string, Last_status, Status);
            if (Status ^= 0)           /* if not fully mapped */
            then
               do;
                  Code = Status;
                  return;
               end;

            call addr(envcom) -> envcom_based.sp$dtr(s$putb, ADDR(ADDR
                      (Output_string) -> vs.chars), length(Output_string), 0,
                      code);           /* Output the line */

         end;
      end;

/* ============================================================ CHR$ */

CHR$:
      procedure (i) returns(char);

      dcl i bin;

      dcl 1 ascii based,
             2 c1 bit(8) unaligned,
             2 c2 char(1) unaligned;

         return(addr(i) -> ascii.c2);

      end;

   end;


/* END-CODE */
```

C Spooler Subsystem Messages

Introduction

This appendix lists the messages displayed by the Spooler subsystem and explains how to correct the error condition.

Spooler subsystem messages are displayed in the following format:

```
Error from spooler (Spool-49)
Invalid command line option: -OPT
```

Spooler subsystem messages contain the following components:

Severity	:	Error
Message Body	:	Invalid command line option: –OPT
Localizable Subsystem id	:	Spooler
Non localizable subsystem id	:	spool
Message code	:	49

The Message code identifies the message. The Severity, Localizable subsystem id, and Non localizable subsystem id information are not included in this appendix.

Some messages that you receive while using the SPOOL and PROP commands may be messages from PRIMOS rather than from the Spooler subsystem which are also not listed in this appendix.

Messages

`-<000> must be followed by a decimal value`

Message code 83. You did not give a decimal value to an option. Check your SPOOL command line.

`aaa Reading top-level SPOOL* directory`

Message code 174. A PRIMOS error occurred while reading the SPOOL* directory. See your System Administrator.

`** Aborted **`

The previous print job was aborted in response to a PROP command with the –ABORT option. No action is required.

`Access to pre-rev. 21 style queue on disk aa failed with error nn`

Message code 22. The SPOOL command was unable to access the specified disk. The fault is not in the Spooler subsystem but in the disk access. Appropriate action depends on the reason for failure.

`Access to spool queue failed`

Message code 6. The SPOOL command was not able to access the spool queue. Check that the queue exists, is visible from your node, and has the correct ACL.

`Address not numeric`

Message code 127. There is a mistake in the environment file. Correct the environment file.

`Address too long`

Message code 128. There ia a mistake in the environment file. Correct the environment file or use a different environment.

`-ALIAS not allowed with pre-rev. 21 queue`

Message code 7. You used the –ALIAS option of the SPOOL command and also used –DISK to specify a pre-Rev. 21.0 spool queue. Issue another SPOOL command, either without the –ALIAS option, or directed to a Rev. 21.0, or post Rev. 21.0, queue.

`Allocating semaphore`

Message code 2. This is a status message from the despooler phantom. No action is required.

`-ALTernate has been used on an earlier line.`

Message code 191. More than one MAP directive has the –ALT option to it. Check the MAP directives.

`Argument needed with -SET_FONT`

Message code 58. You omitted *fontname*. Include *fontname*.

`Argument needed with -SET_PAPER_BIN`

Message code 59. You omitted *n* or MANUAL. Include *n* or MANUAL.

`AS GPn`

Assign the GP*n* command string.

`Attaching to SPOOL* directory`

Message code 8. A PRIMOS error message occurred while attaching to the SPOOL* directory. See your System Administrator.

`Attaching to top-level SPOOL* directory`

Message code 3. This is a status message. No action is required.

`Attaching to SPOOL*>LOG sub directory`

Message code 9. This is a status message. No action is required.

`Attribute too long (max 16 chars)`

Message code 149. There is a mistake in the environment file. Correct the environment file or use a different environment.

`** Backed Up **`

The backup instruction issued with the –BACKUP option of the PROP command successfully completed. No action is required.

`Cannot alter RWLOCK to UPDT for -SPOOL_WHILE_OPEN`

Message code 77. The Spooler cannot change the RWLOCK to UPDT. Change the RWLOCK on the file to UPDT.

`-COMPRESS is not available in this release`

Message code 156. You tried to use the –COMPRESS option of the PROP command. This option was removed at Rev. 21.0. Make all changes to environment files with a text editor.

`-CREATE is not available in this release`

Message code 157. You tried to create a printer environment with the –CREATE option of the PROP command. This option was removed at Rev. 21.0. See Chapter 3, Defining Printer Environments, for the correct procedure.

`Cannot attach to queue directory on specified disk`

Message code 26. The SPOOL command could not attach to the directory that contains the spool queue on the specified disk. Check the ACL on the directory that it needs to attach to.

`Cannot attach to SPOOL*>RESTRICTED_USERS directory`

Message code 10. The Spooler cannot attach to the directory. Check to see if the directory is there.

`Cannot cancel request nn as the despooler already has a request waiting`

Message code 123. You used the –CANCEL option of the SPOOL command to cancel a job that has already begun to print. In this particular case, the despooler phantom already has a request pending (for example, to stop at the end of the current job). It is only able to have one request pending and so cannot handle your request. Use the –DROP option of the PROP command instead. If you are not a privileged user, contact the Operator or System Administrator.

`Cannot cancel request nn as it is being printed remotely`

Message code 124. You can only cancel print jobs on the local node of a network. Either contact the Operator of the remote system, or log in on the remote system and cancel the print job yourself.

`Cannot find handler`

The despooler initialization routine could not find the print handler specified in the environment file. Check the environment file to make sure that the print handler is specified correctly. The problem could be a simple spelling mistake.

If the environment file is correct, check that the print handler exists. See Chapter 2, Setting Up the Spooler Subsystem, for a list of the standard handlers that Prime supplies. If the print handler is not in the standard library, ensure that you have provided a suitable nonstandard handler, and that the search rules in SPOOL*>ENTRY$.SR are modified so that it can be found.

`Cannot modify request as it includes information not recognized by this rev.`

Message code 53. You used the –MODIFY option of the SPOOL command to change the details of a print request that is not in Rev. 21.0 or later format. Cancel the request and issue a new SPOOL command.

`Cannot modify request as it is in rev. 18 format`

Message code 61. You used the –MODIFY option of the SPOOL command to change the details of a print request that is not in Rev. 21.0 or later format. Cancel the request and issue a new SPOOL command.

`Cannot open attributes file`

Message code 29. The SPOOL command tries to check the attributes in a SPOOL command against those in an attributes file. The file that cannot be opened could be user specific, project specific, or the default file. Find out which file is locked, and why.

`Cannot open file to print`

Message code 21. Either the directory specified does not exist, or the file does not exist, or a failure other than NO_RIGHTS (E$NRIT) occurred when trying to change the RWLOCK on the file, or the file cannot be opened for reading. Check that the file exists, is not locked, and that the ACL in force allows the despooler phantom to access the file.

`Cannot open queue file`

Message code 30. The SPOOL command cannot open the spool queue to put a new request in it. Find out which process has a lock on the file and why.

`Changing name of log file`

Message code 11. The log filename is being changed as specified in the LOG command of the environment file. No action is required.

`Check New Page`

Operator instruction. Check the page at the printer.

`Check Paper Alignment`

Operator instruction. Check the paper alignment at the printer.

`Closing semaphore`

Message code 155. Status message from the PROP command. For details of semaphores, see the *Prime 50 Series Technical Summary*. No action is required.

`Command line contains invalid characters`

Message code 42. You issued a SPOOL command with nonalphanumeric characters in it. Possibly you have pressed the Ctrl key inadvertently. Issue the correct SPOOL command.

`Command not available.  System date and time not set.`

Message code 15. The system date and time were not set at system cold start. Until they are set, the SPOOL command cannot be used. Set the system date and time and reissue the PROP –COLDSTART command. Ensure that the system date and time are correctly set at subsequent system cold starts.

`** Completed **`

Confirmation that the specified print request is completed. This message is only given if you used the –NOTIFY option of the SPOOL command. No action is required.

`Conflicting options`

Message code 114. There is a mistake in the environment file. Correct the environment file or use a different environment.

`Creating SEM file`

Message code 4. Status message. As part of despooler initialization, a SEM file is created to allow communication with the semaphore where the despooler phantom is queued when it has no work to perform. For details of semaphores, see the *Prime 50 Series Technical Summary*. No action is required.

`Decimal arguments, separated by spaces, must follow the PLOT_PAGE_DEF directive.`

This error message displays when processing the PLOT_PAGE_DEF environment directive. Check for an empty argument list.

`Default already specified`

Message code 16. You allocated –DEFAULT twice when writing SET statements in the environment file. Reconsider your SET statements.

`-DEFault has been used on an earlier line.`

Message code 192. More than one MAP directive has the –DEF option to it. Check the MAP directives.

`-DELETE is not available in this release`

Message code 159. You tried to delete an environment with the –DELETE option of the PROP command. This option was removed at Rev. 21.0. See Chapter 3, Defining Printer Environments, for the correct procedure.

`Device not responding`

Used when running a despooler over TCP/IP. If the device runs out of paper or goes offline, this message is reported in the log file.

`-DISK and -ON may not be used together`

Message code 17. You issued a SPOOL command that includes both the –DISK option and the –ON option. Use –DISK to specify a pre-Rev. 21 spool queue on that disk, and –ON to specify the queue on a Rev. 21.0 or later node. If you want to put the request in two different spool queues, issue separate commands with –ON and –DISK.

`Despooler already has a suspended item.`

You attempted to suspend a despooler phantom that has already been suspended for a print job.

`Despooler already has an outstanding request`

Message code 166. You issued a PROP command to tell a despooler phantom to take some action, but the despooler phantom has already been given its next instruction. Wait until the despooler carries out the pending instruction, and reassess the situation.

`Despooler not currently printing`

Message code 161. You issued the PROP command with the –DROP option, but the despooler phantom was not printing a file. No action is required.

`Despooler not currently hanging`

Message code 160. You issued the PROP command with the –CONTINUE option, but the despooler phantom was not hanging. No action is required.

`Device not recognised`

Message code 118. There is a mistake in the environment file. Correct the environment file.

`Device not responding`

Reports that a printer running over TCP/IP has gone offline, or run out of paper. This message appears in the log file.

`Disk aa not known`

Message code 18. You used the –DISK option of the SPOOL command to specify a disk, where *aa* is the disk name, but the disk is not known to your node of the network. Make sure that the partition is visible from your node of the network.

`DPTX: Inputting from Printer`

DPTX status information. No action is required.

`DPTX: Outputting to Printer`

DPTX status information. No action is required.

`** Dropped **`

Confirmation that a printing job has been dropped, in response to a PROP command. No action is required.

`Duplicated attribute name`

Message code 19. You gave the same attribute more than once on the SPOOL command line. Issue the correct SPOOL command.

`-ELEMENT must be specified`

Message code 188. You have not specified –ELEMENT when defining an AUXILIARY statement in the environment file. Include –ELEMENT.

`env_name ** Check printer - output buffer full - possible printer fault **`

Used by SP$AMLC (every five minutes) and TCP_IP_PHASE1 (variable time) to notify the user of a printer hang.

`Environment already active`

Message code 109. You tried to use the PROP command to start an environment that is already active. No action is required, unless the command you gave was not what you intended.

`Environment name missing`

Message code 167. You omitted an environment name from a PROP command. Issue the correct command.

`Environment name not required`

Message code 164. You gave an environment name in a PROP command where it was not needed. Issue the correct command.

`Environment name too long`

Message code 165. There is a mistake in the PROP command. Issue the correct command. If the name you gave really is the name under which the environment file is saved, then you must rename the environment file with a shorter name. Environment names can be a maximum of 16 characters.

`Environment not active`

Message code 163. You used the PROP command to stop or hang an environment that is not active, or to instruct an inactive environment to drop or abort a print job. This is most likely due to a typing mistake. Issue the command that you intended.

`Error reading file to be spooled`

Message code 12. This is a disk-handling error. Reissue the SPOOL command. If the error recurs, investigate the disk drive and controller.

`Error reading SWO_VALID_USERS file`

Message code 97. An error occurred in reading the file. Check with your System Administrator.

`Error writing data file copy`

Message code 13. This is a disk-handling error. Reissue the SPOOL command. If the error recurs, investigate the disk drive and controller.

`EVFU Channel 1 is for top of form only`

Message code 178. A mistake was found in the EVFU file. Correct the EVFU file.

`EVFU channel 1 must be top of form`

Message code 181. A mistake was found in the EVFU file. Correct the EVFU file.

`Facing banner disabled, cannot be used when header is set to 0`

This message is recorded in the log file as a warning.

`Failed to add entry`

Message code 24. The SPOOL command software could not add an entry to the spool queue. The reason is given in the rest of the message. Appropriate action depends on the reason for failure.

`Failed to add entry to <ldev>SPOOL_DATA* - n`

Message code 25. The SPOOL command software could not copy the file to be printed to the SPOOL_DATA* directory on the named partition. The reason is given in the rest of the message. Appropriate action depends on the reason for failure.

`Failed to delete request`

Message code 27. The SPOOL command software could not remove a request from the spool queue. The reason is given in the rest of the message. Appropriate action depends on the reason for failure. Note that you can only cancel your own requests, unless you are a privileged user.

`Failed to modify request`

Message code 28. The SPOOL command software could not modify a request in the spool queue. The reason is given in the rest of the message. Appropriate action depends on the reason for failure. Note that you can only modify your own requests, unless you are a privileged user.

`Failed to open PRT file`

Message code 23. The Spooler subsystem could not open a PRT file for your print request. The reason is given in the rest of the message. Appropriate action depends on the reason for failure.

`Failed to read queue file header`

Message code 31. The Spooler subsystem was unable to read the header data in the queue file. Reissue the command that led to the error. If the error recurs, the spool queue file might be corrupted. This could have resulted from a disk error.

Follow your installation procedures for recovering from data loss. Restore the Spooler directories to their most recent known state, but retain copies of files that were recently added to the SPOOL_DATA* directories, in case the originals have been deleted from the file system. Notify users of the event, in case they need to reissue some print requests.

```
Failed to read queue file
```

Message code 32. The Spooler subsystem could not read the spool queue file. The reason is given in the rest of the message. Appropriate action depends on the reason for failure.

```
Failed to read request
```

Message code 33. The Spooler subsystem could not read an entry in the spool queue. The reason is given in the rest of the message. Appropriate action depends on the reason for failure.

```
Failed to write queue entry
```

Message code 34. The SPOOL command software could not enter a request in the spool queue. The reason is given in the rest of the message. Appropriate action depends on the reason for failure.

```
Failed to write queue file header
```

Message code 35. The Spooler subsystem could not update the header of the spool queue file. The reason is given in the rest of the message. Appropriate action depends on the reason for failure.

```
File is being printed
```

Message code 93. You used the –MODIFY option of the SPOOL command to alter the options or attributes of a print request, but the changes cannot be made because the file is already printing. If the original command will produce an unacceptable printout, cancel the print job, and issue a new SPOOL command with the correct options and attributes.

```
File Last Modified: ** invalid date **
```

Printed on a line in the banner page with the DTM (date/time/modified) is invalid.

```
Filename only please, no wildcard
```

Message code 36. You used wildcards to name a map in the environment file. Use explicit filenames only.

```
File size exceeds maximum size allowed.  See your
administrator.
```

You attempted to spool a file whose size (the number of disk records) exceeds the maximum number of disk records allowed into the local spool queue. See your System Administrator.

`File size smaller than required minimum size. See your administrator.`

You attempted to spool a file whose size (the number of disk records) is less than the minimum number of records allowed into the local spool queue. See your System Administrator.

`Fontname required with -SET_FONT`

Message code 57. You omitted *fontname*. Include the *fontname*.

`-FROM not allowed with pre-rev. 21 queue`

Message code 72. You are trying to use the –DISK and –FROM options together. See your System Administrator.

`FROM value must not be greater than TO value`

Message code 38. You issued a FROM value that is greater than the TO value. Check the FROM and TO values.

`** Hanging **`

Confirmation that PROP command has been obeyed. No action is required.

`-HDR not allowed with pre-rev. 21 queue`

Message code 48. You are trying to use the –DISK and –HDR options together. See your System Administrator.

`Illegal channel number in EVFU file`

Message code 179. There is a mistake in the EVFU file. Correct the EVFU file.

`Illegal line number in EVFU file`

Message code 180. There is a mistake in the EVFU file. Correct the EVFU file.

`Incompatible attributes : aaa, aaa`

Message code 39. You issued a SPOOL command with two attributes that describe the same feature. Issue the correct SPOOL command.

`Incompatible options`

Message code 40. You issued a SPOOL command with two or more options that do not make sense together. Issue the correct SPOOL command.

`Initialized`

Confirmation that a print handler is successfully initialized. No action is required.

`Initializing printer handler`

Initialization of the despooler's print handler has started. No action is required.

`Insufficient access rights`

Message code 185. You are not privileged to use the failed PROP option. See your System Administrator.

`Invalid buffer returned from node aa`

Message code 47. Network problem. Check the status of the remote node. *aa* is the nodename. If the node is up, reissue the command.

`Invalid character in attribute name : aaa`

Message code 41. An attribute in your SPOOL command contains nonalphanumeric characters. *aaa* is the attribute that you entered incorrectly. If it appears to be correct, you may have accidentally entered a control character. Issue the correct SPOOL command.

`Invalid command line.  Type SPOOL -HELP for assistance`

Message code 43. Use SPOOL –HELP or refer to Appendix A, SPOOL and PROP Commands.

`Invalid command line option : -aaa`

Message code 49. You issued an incorrect option in a SPOOL command. *–aaa* is the incorrect option. Check for a spelling mistake. If the option appears to be correct, you may have accidentally entered a control character. Refer to Appendix A, SPOOL and PROP Commands, for details of the command options that you can use.

`Invalid device type`

An invalid device type is specified in the environment file. Check the environment file. Either specify one of the available devices, or provide a print handler for the new device type that you want to use.

`Invalid Environment File`

Message code 45. There is a mistake in the environment file. Correct the environment file.

`Invalid -FCTY data`

Message code 129. There is a mistake in the environment file. Correct the environment file.

`Invalid language name`

Message code 190. The call to SI$VLA failed while trying to validate a language name. You used an unsupported language name. Issue the command correctly.

`Invalid number of header pages (must be 0/1/2)`

Message code 15. There is a mistake in the environment file. Correct the environment file.

`Invalid page width`

Message code 152. There is a mistake in the environment file. Correct the environment file.

`Invalid parameter (must be ON/OFF)`

Message code 120. There is a mistake in the environment file. Correct the environment file.

`Invalid parameter (must be ON/OFF/SAVE)`

Message code 111. There is a mistake in the environment file. Correct the environment file.

`Invalid -PRID data`

Message code 130. There is mistake in the environment file. Correct the environment file.

`Invalid records increment, must be > 0`

Message code 115. There is a mistake in the environment file. Correct the environment file.

`Invalid second count for Spool While Open Wait Time (must be 1-600)`

Message code 117. There is a mistake in the environment file. Correct the environment file.

`Invalid -SET_FONT`

Message code 46. You have not given a valid *fontname*. Recheck the command.

`Invalid -SET_PAPER_BIN`

Message code 50. You specified a bin number outside the range 1 through 9, or not typed the string MANUAL correctly. Recheck the command.

`Invalid Spool While Open Times To Check (must be 0-32767)`

Message code 116. There is a mistake in the environment file. Correct the environment file.

`Invalid stationery feed type`

Message code 134. There is a mistake in the environment file. Correct the environment file.

`Invalid time in -DEFER`

Message code 44. You issued a SPOOL command with an incorrect defer time. The time must be in the range 0000 through 2359, though a number of formats are permitted. Reissue the SPOOL command with the time specified correctly. See Appendix A, SPOOL and PROP Commands, for details.

`-LINE option missing`

Message code 148. There is an error in the environment file; –LINE is missing from the ASYNC command. Correct the environment file.

`-LIST -ALL specified with -DISK or -ON`

Message code 52. You issued a SPOOL command to view the spool queue, but have used an incorrect combination of options. SPOOL –LIST –ALL shows all the spool queues, so neither –DISK nor –ON is needed. Issue the correct command to display the queues you wish to see.

`Looking for a further SPOOL_DATA* partition`

The SPOOL command copies files to be printed into a SPOOL_DATA* directory. It has found that the SPOOL_DATA* directory that was previously in use is full and is now looking for another SPOOL_DATA* directory on a different partition. No action is required.

`Map not defined`

Message code 54. This is a despooler log file message. You called a MAP that does not exist in the environment file. Reconsider required MAP.

`Max shutdown time out of range (MIN = 10 MAX = 300)`

When the despooler phantom is started up again, this information is recorded in the log file after the line containing the error.

`Missing Command Line Option`

Message code 168. You issued a PROP command with no command-line option. Issue the correct command.

`Missing map name or filename`

Message code 60. You wrote a MAP without giving a filename. Recheck MAP in the environment file.

`Missing parameter`

Message code 112. There is a mistake in the environment file. Correct the environment file.

`Missing qualifier for option -$`

Message code 122. Missing qualifier for option –option_name. The option expected an argument, but none were given. Issue the correct command.

`-MODIFY is not available in this release`

Message code 169. You tried to use the -MODIFY option of the PROP command to alter an environment. This option was removed at Rev. 21.0. See Chapter 3, Defining Printer Environments, for the correct procedure.

`More than one file specified`

Message code 101. You gave more than one pathname in a SPOOL command. This message could result from a typing mistake. If you miss the hyphen off an option, then the remainder of the option could be interpreted as a filename. If you do wish to print several files, then either issue a separate SPOOL command for each, or use the PRIMOS iteration facility.

`More than one print format specified`

Message code 63. You specified more than one print format in the same SPOOL command; for example, -COBOL and -NO_FORMAT. Reissue the command with only one formatting option.

`Multiple instances of SET_LANDSCAPE`

Message code 64. You issued more than one SET_LANDSCAPE statement in an environment file. Correct the environment file.

`Multiple instances of SET_PORTRAIT`

Message code 67. You issued more than one SET_PORTRAIT statement in an environment file. Correct the environment file.

`Multiple occurrences of -DISK not allowed`

Message code 62. You issued more than one -DISK option in the same SPOOL command. If you do want to enter the request in several spool queues, issue a separate SPOOL command for each one.

`Multiple occurrences of -ON not allowed`

Message code 65. You issued more than one -ON option in the same SPOOL command. If you do want to enter the request in several spool queues, issue a separate SPOOL command for each one.

`-NO_COPY not allowed with pre-rev. 21 queue`

Message code 69. You used -NO_COPY and -DISK in the same SPOOL command. If you want the request to go into the pre-Rev. 21.0 spool queue, you cannot use the -NO_COPY option. If you want to use -NO_COPY, then you must direct the request to a Rev. 21.0 spool queue.

`-NPH not allowed with pre-rev. 21 queue`

Message code 8. You are trying to use the -DISK and -NPH options together. See your System Administrator.

`No active environments`

Response to PROP –STATUS when no environments are active. No action is required.

`No despooler environments`

Response to PROP –STATUS –ALL when there are no environment files in the SPOOL* directory on your system. No action is required.

`No file name given`

Message code 70. You omitted the filename from a SPOOL command. Issue the SPOOL command with the filename in place.

`No PDN support`

Message code 132. Your system does not include Public Data Network support software. Check with your System Administrator.

`No queue entries found to display`

Response to a SPOOL –LIST command. This does not necessarily mean that the queue is empty. If you are not a privileged user, it means that none of your print requests are in the queue. Appropriate action depends on the reason for viewing the spool queue.

`No space in environment table`

Message code 110. The Spooler subsystem keeps data for all active despooler phantoms in an environment table. You tried to start a new despooler phantom, but there is insufficient space in the environment table to store data for it.

If there is no redundant data in the environment table, you could still get this message if there are many active despooler phantoms. There is no precise maximum, but the environment table should accommodate more than 100 active despooler phantoms, so the theoretical limit should never be approached in practice.

If a despooler phantom is not shut down correctly, for example, if it logs out abnormally, its data is not always successfully removed from the environment table. Shut down all despooler phantoms that are not required, with the –STOP option of the PROP command.

`Node name not required`

Message code 154. There is a mistake in the environment file. Correct the environment file or use a different environment.

`Non-candidate for despooling`

Refers to an item of data that is stored in the spool queue entry when a job is canceled. No action is required.

```
-NOTIFY has been disabled. It is not possible to use
-NOTIFY and -ON together
```

Disables the –NOTIFY option because you cannot use –NOTIFY with the –ON option.

```
Notifying semaphore
```

Message code 170. Status information only. See the *Prime 50 Series Technical Summary* for information on semaphores. No action is needed.

```
Numeric argument missing
```

Message code 74. You have not specified a paper bin number. Specify the paper bin number.

```
Numeric argument missing after -aaa
```

Message code 68. You omitted the numeric argument from an option in a SPOOL command. *–aaa* is the option that requires the numeric argument. Reissue the command with the numeric argument in place.

```
Only one map can be specified using -XLATE
```

Message code 86. More than one MAP was issued with the –XLATE option. Check your SPOOL command line.

```
-OPEN and -NO_COPY may not be used together
```

Message code 89. Reissue the SPOOL command with one or both of the incompatible options removed.

```
Opening CONFIG file
```

Message code 87. Status message while checking or initializing an environment. No action is required.

```
Opening Environment File
```

Message code 88. Status message while checking or initializing an environment. No action is required.

```
Opening EVFU file
```

Message code 182. Status message while initializing the EVFU for a printer. No action is required.

```
Opening top-level SPOOL* directory
```

Message code 17. Status information only. No action is required.

```
Opening semaphore
```

Message code 91. Information only. A semaphore is a queue of processes where the despooler phantom is queued when it has no work to perform. For details of semaphores, see the *Prime 50 Series Technical Summary*. No action is required.

`Option -v`

Message code 105. Option has an unexpected qualifier. You used an argument to an option where no argument was expected. Issue the command line correctly.

`Option conflict, -DATA_SENSE_ENABLE and -NO_DATA_SENSE_ENABLE`

Message code 142. You used conflicting options. Reconsider options.

`Option conflict, -DATA_SET_CONTROL and -NO_DATA_SET_CONTROL`

Message code 141. You used conflicting options. Reconsider options.

`Option conflict, -ECHO and -NO_ECHO`

Message code 143. You used conflicting options. Reconsider options.

`Option conflict, -ERROR_DETECTION and -NO_ERROR_DETECTION`

Message code 144. You used conflicting options. Reconsider options.

`Option conflict, -LINE_FEED and -NO_LINE_FEED`

Message code 145. You used conflicting options. Reconsider options.

`Option conflict, -REVERSE_XOFF and -NO_REVERSE_XOFF`

Message code 146. You used conflicting options. Reconsider options.

`Option conflict, -XOFF and -NO_XOFF`

Message code 147. You used conflicting options. Reconsider options.

`Option not recognised`

Message code 121. There is a mistake in the environment file. Correct the environment file.

`Overlapping margins`

Message code 151. There is a mistake in the environment file. Correct the environment file.

`Parameter not numeric`

Message code 113. There is a mistake in the environment file. Correct the environment file.

`Pathname not available`

The –SFI option in the SPOOL command prevented the pathname from being displayed. No action is required.

`Printer Warning: Carriage Power Off or Print Hung.  DPTX`

`Printer Failure: Data Check.  DPTX`

`Printer Failure: Equipment Check.  DPTX`

`Printer Failure: Mechanically Disabled.  DPTX`

`Printer Warning: Printer Timeout.  DPTX`

`Queue file header is corrupt`

Message code 14. The data in the spool queue is corrupt. This could have resulted from a disk error. Follow your installation procedures for recovering from data loss. Restore the Spooler directories to their most recent known state, but retain copies of files that were recently added to the SPOOL_DATA* directories, in case the originals have been deleted from the file system. Notify users of the event, in case they need to reissue some print requests.

`Queue not available at present`

Message code 125. The reason is given in the rest of the message. For example, a disk might be shut down. Appropriate action depends on the reason why the queue is not available.

`Reading CONFIG file`

Message code 95. Status information only. Occurs while checking, or initializing, an environment. No action is required.

`Reading Environment File`

Message code 96. Status information only. Occurs while checking, or initializing, an environment. No action is required.

`Reading EVFU file`

Message code 183. Status information only. Occurs during initialization of the EVFU for a printer. No action is required.

`-RELEASE cannot be applied -NOW`

Message code 172. The –RELEASE option to PROP only works with a timing of –FINISH or –STOP. Issue the correct PROP command.

`Re-opening log file`

Message code 98. Status information only. Occurs while checking, or initializing, an environment. No action is needed.

`Request nnn added to queue : aaa`

Confirmation that your SPOOL command is successful, and that the print request has been added to a pre-Rev. 21.0 queue. *nnn* is the request number. *aaa* is the queue name. You should note the request number if you think you could need to cancel or modify the request. No action is required.

```
Request nnn added to queue, file unit xx
```

Confirms that your SPOOL command, with –OPEN option, is successful and that the print request was added to a spool queue. Confirms the number of the file unit that has been opened. No action is required.

```
Request nnn added to queue, nn records : pathname
```

Confirmation that your SPOOL command is successful, and that the print request has been added to a Rev. 21.0 queue. *nnn* is the request number. *nn* is the size of the file to be printed (in records). You should note the request number if you think you could need to cancel or modify the request. No action is required.

```
Request nnn cancelled
```

Confirms that your SPOOL command successfully canceled a print request. No action is required.

```
Request nnn not found in spool queue
```

Message code 99. You issued a SPOOL command to cancel or modify a print request, but the request could not be found in the queue. This could be because the file has already been printed. If the printed output is not adequate, reissue the print request with the correct attributes and options.

```
RESET : New environment is aaa
```

Confirmation that a PROP command to reset an environment has been successfully obeyed. *aaa* is the name of the new environment. No action is required.

```
Re-setting SPOOL_QUEUE*
```

Message code 175. Status information to confirm the progress of PROP –COLDSTART. No action is required.

```
** Restarted **
```

Confirmation that printing has restarted. No action is required.

```
SET_FONT escape sequence missing
```

Message code 56. You omitted the SET_FONT escape sequence in the environment file. Include the escape sequence.

```
-SET_FONT not allowed with pre-rev. 21 queue
```

Message code 71. You used –DISK with –SET_FONT. If you require a pre-Rev. 21.0 queue, you cannot use –SET_FONT.

```
-SET_LANDSCAPE not allowed with pre-rev. 21 queue
```

Message code 73. You used –DISK with –SET_LANDSCAPE. If you require a pre-Rev. 21.0 queue, you cannot use –SET_LANDSCAPE.

`-SET_PAPER_BIN not allowed with pre-rev. 21 queue`

Message code 75. You used –DISK with –SET_PAPER_BIN. If you require a pre-Rev. 21.0 queue, you cannot use –SET_PAPER_BIN.

`-SET_PORTRAIT not allowed with pre-rev. 21 queue`

Message code 76. You used –DISK with –SET_PORTRAIT. If you require a pre-Rev. 21.0 queue, you cannot use –SET_PORTRAIT.

`Setting RWLOCK on SEM file`

Message code 5. This message is from the despooler phantom. Status information only. There is a SEM file for each active despooler phantom to allow communication with the semaphore where the despooler phantom is queued when it has no work to perform. For details of semaphores, see the *Prime 50 Series Technical Summary*. No action is required.

`-SFI not allowed with pre-rev. 21 queue`

Message code 85. You are trying to use the –DISK and –SFI options together. See your System Administrator.

`Sign on message too long`

Message code 131. There is a mistake in the environment file. Correct the environment file.

`Specified User name too long.`

A user name listed with the ERROR_NOTIFY directive in the environment file is too long.

`-SPOOL_WHILE_OPEN not allowed with pre-rev. 21 queue`

Message code 78. You are attempting to use –SPOOL_WHILE_OPEN with the –DISK option. Reconsider your options.

`Spooler Failure: Command Reject.  DPTX`

`Spooler Failure: Operation Check.  DPTX`

`Spooler database has not been initialized`

Message code 158. You tried to use the PROP command to start a despooler phantom, or the SPOOL command to print a file, before initializing the Spooler subsystem. Issue the PROP –COLDSTART command.

`Spooler database has not been shared`

Message code 171. The correct procedures were not followed when the Spooler subsystem was started. Perform the startup procedure correctly, as described in Chapter 2, Setting Up the Spooler Subsystem.

`Starting up phantom`

Message code 176. Confirmatory message from the PROP command..No action is required.

`Stop bits must be 1 or 2`

Message code 139. You used an invalid qualifier with the –STOP_BITS option of the ASYNC directive. Check the environment file.

`** Stopping **`

Confirmation that an environment is stopping as instructed in a PROP command. No action is required.

```
String range error with control code argument, page number
x, line number y.
```

Message written to log file. (*x* and *y* are page and line numbers, respectively.)

`-SUSPEND must occur -NOW`

Message code 162. The –SUSPEND option to PROP only works with a timing of –NOW. Issue the correct PROP command.

```
System print job limits (disk records) are:
minimum: n
maximum: n
```

This message reports what the current minimum and maximum job size limits are for the local spool queue.

```
The file used in -NOCOPY must be on the same node as the
spool queue
```

Message code 82. You issued a SPOOL command with the –NOCOPY option, but the file to be copied is not on the same node as the spool queue to which you directed your request. Either reissue the SPOOL command without the –NOCOPY option, or log in on the system where the file you want to print is located and reissue the same SPOOL command.

`The filename is too long, 1 to 32 characters please`

Message code 37. When writing a MAP environment statement, you specified too long a string for the filename. Respecify the correct filename.

`The map name is too long, 1 to 32 characters please`

Message code 55. When writing a MAP environment statement, you specified a string that is too long for the map name. Respecify the correct map name.

`The queue is empty`

Status information only, in response to a SPOOL –LIST command. Appropriate action depends on your reason for viewing the spool queue.

`The queue is full`

Message code 94. The SPOOL command is unable to add your request to a pre-Rev. 21.0 queue. Direct the request to a different queue, or wait until some items in the queue are removed.

`This map has been defined on an earlier line.`

Message code 193. This map name has already been given. Correct the environment file.

`-TO not allowed with pre-rev. 21 queue`

Message code 79. You are attempting to use –TO with the –DISK option. Reconsider options.

`Too many attributes (max 32)`

Message code 150. There is a mistake in the environment file. Correct the environment file.

`Too Many Command Line Options`

Message code 177. You issued too many options in a PROP command. Issue the correct command. See Appendix A, SPOOL and PROP Commands, for details of permitted combinations of options.

`Too many device attributes given`

Message code 100. You issued too many attributes in a SPOOL command. Issue a correct command. Details depend on the attributes in use at your installation.

`Too many fonts specified`

Message code 102. You specified more than one –SET_FONT in a SPOOL command. Retype the SPOOL command.

`Too many MAP commands.  The maximum is 64`

Message code 103. You specified too many MAP commands in an environment file. Reconsider the environment file.

`Too many nodes/disks`

Message code 1. There is a mistake in the environment file. Correct the environment file or use a different environment.

`Too many paper feed devices`

Message code 104. You stated two sources for your paper when using –SET_PAPER_BIN. Retype –SET_PAPER_BIN.

`Too many paper orientations`

Message code 66. You specified more than one paper orientation option when using the SPOOL command. Use one only of –SET_LANDSCAPE or –SET_PORTRAIT.

`Too many users identified for error message notification.`

You cannot enter more than eight users with the ERROR_NOTIFY directive in the environment file. Check the ERROR_NOTIFY directive in the environment file to be sure that only eight users are listed.

`Trying as rev. 21 style queue (-ON v)`

Status information only, during despooler initialization. No action is required.

`-TUNIT can only be used with -OPEN`

Message code 106. Your SPOOL command included the –TUNIT option but did not also include the –OPEN option. Reissue the SPOOL command, but add the –OPEN option.

`Unable to transliterate command line, SIT code = $`

Message code 189. The call to SI$TRB failed. See your System Administrator; this problem is with the Standard Internationalization Tool (SIT).

`Unable to transliterate command line, SIT error code`

Message code 186. The call to SI$TRB failed. See your System Administrator; this problem is with the Standard Internationalization Tool (SIT).

`Unknown device attribute : aaa`

Message code 107. Your SPOOL command contains an attribute that is not allowed on your system, or that is not allowed to your project group or user ID. *aaa* is the illegal attribute. This message could result from a spelling mistake. Check your SPOOL command and, if necessary, reissue it with the attribute spelled correctly. If the attribute exists, but you are not allowed to use it, see your System Administrator.

`Unknown PDN error status nnn`

Message code 126. This PDN error is not recognized. Refer to your network or device manual.

`Unrecognised Data_Set_Sense`

Message code 135. You specified an incorrect value for the –DATA_SET_SENSE option when using the ASYNC directive. Check the environment file.

`Unrecognised parameter type`

Message code 119. There is a mistake in the environment file. Correct the environment file.

`Unrecognised parity`

Message code 137. You specified an incorrect parity in the ASYNC directive. Check the environment file.

`Unrecognised parity, must be ODD or EVEN`

Message code 218 from PROP or DESPOOL. An unrecognized parity was given with the –SERIAL_PARITY or –PARALLEL_PARITY option. Correct the parity value.

`Unrecognised speed given with -BAUD`

Message code 217 from DESPOOL. You specified an unsupported baud rate with the MPC4000 –BAUD directive. Change the value of –BAUD in the environment file.

`Unrecognised token in EVFU file`

Message code 184. There is a mistake in the EVFU file. Correct the EVFU file.

`Unsupported character length`

Message code 136. You specified an incorrect qualifier to the –CHAR_LENGTH option of the ASYNC directive. Check the environment file.

`Unsupported protocol`

Message code 138. You specified an incorrect qualifier to the –PROTOCOL option of the ASYNC directive. Check the environment file.

`Unsupported speed`

Message code 140. You specified an incorrect qualifier to the –SPEED option of the ASYNC directive. Check the environment file.

`User data too long`

Message code 133. There is a mistake in the environment file. Correct the environment file.

Value of *a* not within allowed range of *b* to *c*.

The environment directive value is not within the range specified. This message displays the actual values for the variables *a*, *b*, and *c*.

`Value out of allowed range`

Message code 108. You specified a paper bin number outside the range of 1 through 9. Respecify the paper bin number.

```
Value out of range in -aaa.  Min xx, max yy
```

Message code 92. You specified an incorrect numeric argument to an option in a SPOOL command. *–aaa* is the option with the incorrect argument. Reissue the SPOOL command with the numeric argument in the correct range.

```
-XLATE not allowed with -PLOT
```

Message code 81. You chose two incompatible options. Reconsider the option.

```
-XLATE not allowed with pre-rev. 21 queue
```

Message code 80. You are attempting to use –XLATE with the –DISK option. Reconsider the options.

```
You do not have clearance to use -SPOOL_WHILE_OPEN
```

Message code 51. You are not privileged to use concurrent printing. See your System Administrator.

```
You do not own this request
```

Message code 84. You attempted to modify another person's spool request. You do not own this request. You tried to cancel or modify another user's print request. You must be a privileged user to modify other user's requests. The message could result if you typed the wrong request number in the SPOOL command.

Defining and Using PostScript Printers

D

Introduction

This appendix contains the following topics:

- What is PostScript
- Setting up PostScript printers
- Establishing the PostScript error log
- Spooling files to PostScript printers
- Defining page formatting options
- Defining PostScript fonts
- Using PostScript dictionary files
- Sample files for PostScript support
- User's view of PostScript printing
- SPOOL formatting options
- Reporting errors to users

What Is PostScript?

PostScript is a high-level language that intelligent printers use to format files. PostScript printers are intelligent because they contain their own computer and memory (some include a disk) for loading PostScript dictionaries and fonts, and handling other printer functions (explained later in this appendix). PostScript files are text files that contain PostScript language code.

Formatting Files

Non-PostScript printers intepret data as ASCII codes that they convert into characters at the printer. PostScript printers interpret data as a program in the PostScript language that is executed to produce formatted output. To print the word *hello* on most printers, you would normally send the word *hello* (ASCII text) to the printer, followed by a form-feed character to eject the page.

To accomplish this on a PostScript printer, you must send a PostScript program that establishes the proper graphical state, chooses an appropriate font, positions the point (similar to a cursor) to the appropriate place on a page, images the desired text, and tells the printer to transfer that image to the physical piece of paper. For example, the following PostScript program prints the word *Hello* on a PostScript printer:

```
%!PS-Adobe-2.1
%%Title: hello.ps
%%EndComments
%%EndProlog
%%Page: 1 1
  save
  /Courier findfont    % Select Courier font...
  12 scalefont setfont % ...with a 12-point size
  0  792  moveto       % Move to upper-left corner of US letter-sized page
  72 -72 rmoveto       % Move 1" right and 1" down to leave room for margin
  (Hello) show         % Image the word "Hello" in the selected font
  restore
  showpage             % Print the page
%%EOF
```

Communicating With PostScript Printers

PostScript printers communicate in two directions. Not only is PostScript language code sent to the printer but the printer itself sends status and other information about itself to the PostScript device handler. The Spooler subsystem's PostScript device handler performs the following functions:

- Translates the file you wish to print into a PostScript language program that achieves the desired result
- Interacts with the printer and notifies users of errors that occur because of problems with their files, and notifies Administrators of printer problems, such as paper jams or paper outages

Printing Text Files

The Spooler's PostScript device handler uses the power and flexibility of the PostScript language to provide the following capabilities for printing non-PostScript text files:

- Portrait and landscape printing
- Different fonts, styles, and sizes
- *n*-up, or thumbnail, printing that reduces and prints several document pages on a single sheet of paper
- Overlaying the word *DRAFT* or other similar text in outlined letters on each page
- Bold printing similar to the overstrike method used by impact printers to darken the type
- Highly accurate page-accounting for users (because the printer itself manages this function)

Printing PostScript Files

Using the –NO_FORMAT option of the SPOOL command, you can send files containing PostScript language code to the printer and bypass the Spooler's internal page formatting. The PostScript device handler supports the printing of files from software applications such as Scribe® or PC applications that generate PostScript language code. Because PostScript language is text based, PostScript language files are simply text files that contain PostScript code.

Note The PostScript device handler provided with the Spooler subsystem does not currently support PostScript Level 2 or greater. In addition, the device handler neither interprets spooled PostScript language files for the Adobe® Document Structuring Conventions (DSC) nor verifies that the printer's resources (fonts, for example) and capabilities (paper size, for example) match those needed by the spooled PostScript print job. Although you are encouraged to write DSC-compliant PostScript language code, the Spooler's PostScript device handler does not recognize it.

In the sections that follow, you will learn how to set up and configure the Spooler subsystem for using PostScript printers and learn the SPOOL command options for PostScript printing.

Setting Up PostScript Printers

The information in this section assumes you are familiar with Chapter 2, Setting Up the Spooler Subsystem, and Chapter 3, Defining Printer Environments, which provides a detailed explanation of the various directives mentioned in this section.

You can connect PostScript printers to a Prime system using either:

- Asynchronous lines
- TCP/IP

The methods used to connect a PostScript printer are described in the following sections.

Note Although some printers support the AppleTalk® interface, the Spooler subsystem does not support this communication protocol.

Using Asynchronous Lines

The simplest method of communicating with a PostScript printer is through a standard bidirectional asynchronous line. Many PostScript printers are preconfigured for either 1200 baud or 9600 baud RS-232 communication using a 7-bit character length and 1 stop bit. You should choose the higher speed option if available.

Setting the Printer Speed

It may be possible to configure even greater speeds through the printer's front panel, switch settings, or printer-specific PostScript code. See your printer manual for details.

For example, to configure an Apple® LaserWriter® for 19200-baud operation, you can send it the following PostScript program (at 9600 baud because the speed change does not take effect until after the file is sent):

```
%!PS-Adobe-2.1 ExitServer
%%Title: fixbaudrate.ps
%%BeginExitServer: 0
  0  % <------------\-- exitserver password
  serverdict begin exitserver
%%EndExitServer
  statusdict begin
  /setsccbatch where
  {   pop
  }{  (Refer to printer manual for proper way to set baud rate!) = flush
      stop
  } ifelse
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  25    % <-- Select Port: 25 = 25-pin
        %                   9 =  9-pin
  19200 % <-- Select Baud: 1200, 9600, 19200, etc.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  exch 2 copy sccbatch pop ne % Avoid EEROM update if no change.
  {   dup sccbatch exch pop 3 -1 roll exch setsccbatch
  }{  pop pop
  } ifelse
  end %statusdict
%%EOF
```

Note The printer should retain this setting even if powered off.

Setting Data Flow Control

Flow control is essential for PostScript printers because of their limited buffering and because of the time it can take for PostScript printers to print a complex page of graphics. Most PostScript printers support XON/XOFF flow control by default. This corresponds to the –XOFF option of the ASYNC environment file directive. See your printer manual for configuring the desired flow.

Setting Data Bits

Even though the PostScript language is defined as a 7-bit ASCII language, most printers can be configured for an 8-bit data path to allow all 256 possible characters in a font to be accessed without requiring PostScript's special escape mechanism for 8-bit data. While most software that generates PostScript files adheres to the 7-bit standard, the Apple Macintosh® generates PostScript files that use all 8 bits. If you want to be able to print such files, you should configure your printer for an 8-bit data path with no parity and 1 stop bit. See your printer manual for instructions.

For printers such as the Apple LaserWriter, this can be accomplished with the following program:

```
%!PS-Adobe-2.1 ExitServer
 %%Title: fix8bitdata.ps
 %%BeginExitServer: 0
   0  % <-----------\-- exitserver password
   serverdict begin exitserver
 %%EndExitServer
  statusdict begin
  /setsccbatch where
  {    pop
  }{   (Refer to printer manual for proper way to set 8bit data!) = flush
```

```
      stop
  } ifelse
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  25    % <-- Select Port:
        %     25 = 25-pin
        %      9 =  9-pin
  3     % <-- Select Option:
        %      0 = 7-bit data, ignore parity, XON/XOFF
        %      1 = 7-bit data,    odd parity, XON/XOFF
        %      2 = 7-bit data,   even parity, XON/XOFF
        %      3 = 8-bit data,     no parity, XON/XOFF
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  exch 2 copy sccbatch exch pop ne % Avoid EEROM update if no change.
  {   dup sccbatch pop 3 -1 roll setsccbatch
  }{  pop pop
  } ifelse
  end %statusdict
%%EOF
```

Note The printer should retain this setting even if powered off.

Configuring the Asynchronous Line for a PostScript Printer

Most PostScript printers offer the ability for interactive or batch operation. Batch operation is the most convenient for spoolers because it supports the concept of a self-contained PostScript job. After each job, the printer automatically resets itself to a known initial state that is consistent from one job to the next. Because each job is self contained, it is not possible for one job to affect the next job by leaving the printer in an inconsistent state.

The Spooler subsystem requires that you configure the printer for batch operation only. See your printer manual for full details. Finally, because PostScript printers send as well as receive, you must configure the asynchronous line so that it does not echo, otherwise the printer may get confused by its own messages being echoed back to itself.

Using the ASYNC Directive: Use the following ASYNC directive in the environment file to configure the PostScript printer to an asynchronous line:

ASYNC -LINE *line* -SPEED 9600 -CHAR_LENGTH 8 -XOFF -NO_ECHO -PRO TTY -NO_ECHO -NO_LINE_FEED

Selecting a Protocol: The TTY protocol is the only protocol suitable for asynchronous communication with a PostScript printer. Do not use TRAN, TTY8, TT8BIT, or any others, even if you have configured your printer for an 8-bit communication channel.

Establishing Input Buffering: Buffering is another aspect of asynchronous communication that you need to establish. If you are upgrading from an older printer to a PostScript printer, you may not have established an input buffer for the line to which the printer is connected. An input buffer is essential for the printer to send status information to the PostScript device handler, therefore a typical CAB command in your PRIMOS.COMI file (in the top-level directory CMDNC0) for an asynchronous line connection to a PostScript printer would be similar to the following:

```
CAB -LINE line -IBS 1024 -OBS 512 -DMQS 31
```

Notice that in this example a relatively small DMQ size is given; this may be necessary for some printers that do not perform flow control handshaking until their internal buffers are almost full. A larger DMQ size could cause this buffer to overflow, resulting in a PostScript I/O error. For more information on the CAB command and configuring asynchronous lines, see the *System Administrator's Guide, Volume II: Communication Lines and Controllers.*

Determining Printer Timeout Operations

Most PostScript printers implement timeouts for certain conditions that, if exceeded, will cause the printer to abort the current print job. The types of timeout operations are listed below:

- **Job timeout** – the amount of time the entire job is permitted to take before it is considered stuck in an infinite loop. Printers typically default to having this timeout disabled, allowing jobs of infinite length.
- **Manual feed timeout** – the amount of time the printer waits for a sheet to be manually inserted when manual feed operation is selected. Printers typically default to a wait time of one minute.
- **Wait timeout** (also known as **communications timeout**) – the amount of time the printer waits to receive additional data from the host before deciding the host has nothing more to send. Printers typically default to a wait time of 30 seconds.

Refer to your printer manual for more information on the default wait times for your particular printer, and for information to help you determine how to change these defaults to best suit your printing environment.

The manual feed timeout is the only timeout setting that the PostScript device handler adjusts itself. When manual feed operation is selected (discussed later), the handler will extend the manual feed timeout to five minutes, but only if it was originally set to the printer default of one minute. If you have changed the printer's default manual feed timeout to something other than one minute, the device handler uses the value that you have set and the device handler will not extend the timeout to five minutes.

For proper interaction with the PostScript device handler, it is necessary that the wait timeout be set to something reasonable in order to avoid having the printer assume that it has lost contact with the Prime host, particularly when the system is loaded and cannot service the printer as frequently as normal. Prime recommends that the default wait timeout be set to 5 minutes (300 seconds).

For printers such as the Apple LaserWriter, changing the default wait timeout to 300 seconds can be accomplished with the following program (which can be easily adapted to change the other default timeouts as well):

```
%!PS-Adobe-2.1 ExitServer
%%Title: fixtimeouts.ps
%%BeginExitServer: 0
  0  % <------------\-- exitserver password
  serverdict begin exitserver
%%EndExitServer
  statusdict begin
  /setdefaulttimeouts where
  {   pop
  }{  (Refer to printer manual for proper way to set timeouts!) = flush
      stop
  } ifelse
  [ defaulttimeouts ]
  [
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
   -1   % <-- Job         Timeout (seconds; 0=disable, -1=leave alone)
   -1   % <-- Manual Feed Timeout (seconds; 0=disable, -1=leave alone)
  300   % <-- Wait        Timeout (seconds; 0=disable, -1=leave alone)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  ]
  0 1 2
  {   3 copy get dup 0 ge
      {   3 -1 roll exch put
      }{  pop pop pop
      } ifelse
  } for
  pop
  [ defaulttimeouts ]
  false
  0 1 2
  {   4 copy exch pop get 3 1 roll exch get ne { pop true } if
  } for
  exch pop
  {   aload pop setdefaulttimeouts
  }{  pop
  } ifelse
  end %statusdict
%%EOF
```

Note The printer should retain this setting even if powered off.

Using TCP/IP

PostScript printers can also be connected asynchronously through an NTS or LTS terminal server line to TCP/IP.

The despooler phantom uses the TCP/IP environment directive to establish printer communications over TCP/IP. The TCP/IP environment directive requires the –ADDRESS or –NAME option (but not both), and the –PORT option.

Required TCP/IP Options:

–ADDRESS *a.b.c.d*

a.b.c.d is the internet address of the Terminal server (or another device) connected to the printer. This option cannot be used with –NAME. Use either –ADDRESS or –NAME.

–NAME $\begin{bmatrix} \textit{name} \\ \textit{alias} \end{bmatrix}$

name or *alias* is either the name or a recognized alias from the local HOSTS file. The despooler attempts to connect to the address associated with the name or alias in the HOSTS file. Use either –NAME or –ADDRESS.

–PORT *port_no*

Indicates which port the printer is connected to on the Terminal server. It is not the outgoing port from the host machine.

Optional TCP/IP Options:

–DEFER_TIME *nn*

Defines the number of minutes that jobs will be deferred when printer resources are unavailable. *nnnn* is 0–60 minutes.

–LTS_SLEEP *nn*

Sets the number of seconds that the despooler phantom will hold a socket connection after completing its last print job. This option is used to avoid a common Terminal server problem of dropping packets after the socket is closed. *nnnn* is 0–60 seconds.

–RELEASE *nnnn*

Sets the number of seconds to release the printer after printing each job. When you have a network of systems that share a common printer over

TCP/IP, one system can sometimes keep the printer busy for many consecutive jobs, which means that jobs in queues on the other systems queues wait longer for the printer. To share the printer in a more equitable manner, you may want to release the printer for *nnnn* seconds after each print job. *nnnn* is in the range 0–3600 seconds.

-PAD_LF

This option sends double-spaced input to printers connected over TCP/IP that require double-spaced input in order to produce single-spaced output.

For example:

```
TCP/IP -ADDRESS xxx.xxx.xxx.xxx -PORT xxx -DEFER 1 /* connect to printer over
                                                   /* tcp/ip and defer
                                                   /* time is one minute
   device hpjet -delay 4                           /* use HP LaserJet handler
                                                   /* with 4 second delay
                                                   /* between pages
   attribute letter letter_med letter_small        /* allow attributes
   attribute legal  legal_med  legal_small
   format -width 80 -length 65                     /* specify default page
                                                   /* layout
   set_landscape                                   /* allow landscape orientation
   set_portrait                                    /* allow portrait orientation
   set_font -name courier -seq ?E?&l2a0o6d2e64f?(s3t12H -string /* set font sequence
   log -on
   warning -on
   error_notify joe scott@sysa                     /* notify users of printer
                                                   /* problems
   header 1                                        /* one header page
   node sysa                                       /* look at sysa
   node -local                                     /* check local system
   priority_to_local_queue                         /* check local more often
   min_size 1                                      /* don't print empty files
   max_size 100                                    /* only jobs smaller
                                                   /* than 101 disk records
                                                   /* will print
```

Configuring the Port

The following settings are recommended for configuring the port to which the printer connects:

```
Port Type      Slave
   Flow in        NONE
   Flow out       XOFF        /* Xon/Xoff protocol
   Telnet Config
```

```
  Bin         ON         /* allow binary transfers
  MI          YES        /* machine interface
  Inact       0          /* inactivity timeout -none
UART Config
  Baud rate   9600
  BPC         8          /* bits per character
  Parity      None
```

Refer to Chapter 3 for information on configuring a TCP/IP connection.

Notifying Administrators of Printer Errors

While communicating with the PostScript printer, the PostScript device handler periodically asks the printer for its current status and continually checks for error conditions that the printer might report, including the following:

- Responses to queries made by the PostScript device handler itself, for example, the printer's name or present status
- Print engine errors, for example, out of paper, printer jam, no paper tray, and so on
- PostScript errors such as VMerror, limitcheck, undefined, and others that might result from a poorly formed PostScript language document or one that is too complex for the printer to handle
- Other messages that a spooled PostScript program might send back using the various PostScript message operators like print or ==

Print engine errors are sent to the supervisor terminal and are recorded in the printer environment log file in the SPOOL*>LOG directory, assuming that the LOG –ON or LOG –SAVE directive was specified in the environment file. Administrators can also designate up to eight users to whom the despooler will send an immediate message (similar to the MESSAGE –NOW command in PRIMOS). These users are identified with the –ERROR_NOTIFY option of the LOG directive; even users on remote systems may be designated, for example:

LOG –ON –ERROR_NOTIFY ADMIN OPERATOR OPERATOR@SYSB

In this example, the LOG directive establishes ADMIN and OPERATOR on the local system, and OPERATOR on system SYSB as the list of users who will be notified of printer error condition messages.

If PostScript error messages or PostScript program messages are received while the PostScript device handler is processing a job, the PostScript device handler collects the messages until the job is finished and then prints them on a separate PostScript Error Log page following the job. (This is not the same page as the trailer page, and occurs regardless of whether the TRAILER –OFF directive is specified in the environment file.)

A special ATTRIBUTE NOLOG may be specified in the environment file which allows users to control whether they want job-related messages and errors to be printed in this manner or ignored (by spooling the job with the –AT NOLOG option). Messages ignored in this fashion, while not printed on a PostScript Error Log page at the end of the user's job, are instead logged in the environment's log file if one is active.

Spooling Files to PostScript Printers

Users can spool three types of files to a PostScript printer:

- Non-PostScript text files that are formatted as they would be if sent to any other type of printer
- FORTRAN format or COBOL format files, which are also text files
- PostScript language files that are passed directly to the printer for interpretation

Determining File Formatting

The –NO_FORMAT option of SPOOL is normally used to disable the Spooler's formatting when spooling files that perform their own special formatting. If you do not use –NO_FORMAT when spooling specially coded files, such as PostScript files, the Spooler uses its own formatting to paginate and print the file contents (including the formatting codes).

If you use –NO_FORMAT when spooling any type of file to a PostScript printer, the PostScript device handler treats the file as a PostScript file, even if it does not contain any PostScript language codes.

The PostScript device handler inteprets –NO_FORMAT differently than other device handlers. If you use –NO_FORMAT to spool a non-PostScript text file to a PostScript printer, you will get a PostScript error. To achieve the correct formatting when spooling text files to a PostScript printer with –NO_FORMAT, you can include the following directive in the environment file:

```
ATTRIBUTE –AT TEXT
```

This directive allows users to spool text files to a PostScript printer using the following SPOOL command:

SPOOL ***textfile*** **–NO_FORMAT –AT TEXT**

To prevent PostScript errors and the confusion they may cause users, Spool Administrators should add the –POSTSCRIPT_CHECK or the –POSTSCRIPT_OVERRIDE option to the DEVICE POSTSCRIPT directive in the PostScript printer environment file. These options implicitly control the –NO_FORMAT option of the SPOOL command in the following ways:

- –POSTSCRIPT_CHECK automatically adds the –NO_FORMAT option if the spooled file is a PostScript file. –POSTSCRIPT_CHECK recognizes a PostScript file by checking the first four characters for the sequence %!PS.
- –POSTSCRIPT_OVERRIDE goes one step further and ignores any user-supplied –NO_FORMAT option when spooling a non-PostScript file, which would otherwise produce a PostScript error.

Tables D-1, D-2, and D-3 show the results of spooling different types of files to a PostScript printer using certain SPOOL command options. The tables also show what happens when spooling files with options that are incorrect for the desired formatting. The potentially undesirable results are highlighted in bold type.

Table D-1. Spooling Without a DEVICE –POSTSCRIPT_CHECK or –POSTSCRIPT_OVERRIDE Directive in the Environment File

File Format	*SPOOL Option(s)**	*Printed Result*
Text, FTN, or COBOL file	No options –NO_FORMAT –NO_FORMAT –AT TEXT	Paginated, formatted text **PostScript error** Unformatted text (no pagination)
PostScript file (contains PostScript language code)	No options –NO_FORMAT –NO_FORMAT –AT TEXT	**Paginated listing of PS file** PostScript formatted output **Unformatted listing of PS file**

*Assumes –FTN or –COBOL are used when appropriate.

Table D-2. Spooling With a DEVICE –POSTSCRIPT_CHECK Directive in the Environment File

File Format	*SPOOL Option(s)**	*Printed Result*
Text, FTN, or COBOL file	No options –NO_FORMAT –NO_FORMAT –AT TEXT	Paginated, formatted text **PostScript error** Unformatted text (no pagination)
PostScript file (contains PostScript language code)	No options –NO_FORMAT –NO_FORMAT –AT TEXT	PostScript formatted output PostScript formatted output PostScript formatted output

*Assumes –FTN or –COBOL are used when appropriate.

Table D-3. Spooling With a DEVICE –POSTSCRIPT_OVERRIDE Directive in the Environment File

File Format	*SPOOL Option(s)**	*Printed Result*
Text, FTN, or COBOL file	No options -NO_FORMAT -NO_FORMAT -AT TEXT	Paginated, formatted text Paginated, formatted text Paginated, formatted text
PostScript file (contains PostScript language code)	No options -NO_FORMAT -NO_FORMAT -AT TEXT	PostScript formatted output PostScript formatted output **PostScript formatted output**

*Assumes –FTN or –COBOL are used when appropriate.

Note The SPOOL command option -NO_FORMAT -AT TEXT no longer works when the -POSTSCRIPT_OVERRIDE option is added to the DEVICE POSTSCRIPT environment directive in the environment file, as shown in bold in Table D-3.

Establishing Paper Feed Options

The SET_PAPER_BIN environment file directive enables users to select different paper trays or manual feed operation for PostScript printers that provide that support. Refer to Chapter 3, Defining Environment Directives, for a description of SET_PAPER_BIN.

Defining Page Formatting Options

As the Spool Administrator, you can configure the PostScript print handler to provide users with significant flexibility in formatting their spooled output by including certain attributes in the environment file. For example, if you include *n*-up printing attributes in the environment file, users can select *n*-up printing by including the corresponding *n*-up attributes with the SPOOL command. *n*-up printing enables users to print several formatted pages on a single sheet of paper and is explained later in this section.

The following discussion applies primarily to spooled text files, not to spooled PostScript language files that handle their own formatting. However, some features that are useful for spooled PostScript language files are noted explicitly.

Descriptions of available formatting options, fonts, and attribute for files spooled to PostScript printers follow.

Setting Page Orientation

Including the SET_PORTRAIT and SET_LANDSCAPE directives in the printer's environment file allows the use of the corresponding SPOOL –SET_PORTRAIT and –SET_LANDSCAPE options to choose the page orientation. You must identify one choice as the default in the environment file, typically SET_PORTRAIT –DEFAULT.

The PostScript device handler also supports a special attribute called NOSCALE which affects landscape page formatting. The –SET_LANDSCAPE option of the SPOOL command normally scales the font to print the same number of lines on a landscape page as the number of lines printed on a portrait page, which is typically 66 lines. By including ATTRIBUTE NOSCALE in the environment file, users can disable font scaling by issuing the equivalent SPOOL command option –SET_LANDSCAPE –AT NOSCALE on the command line. Disabling font scaling causes fewer lines to print on the landscape page than are printed on a portrait page. The NOSCALE attribute has no affect in portrait orientation. Refer to Spooling Files to PostScript Printers later in this appendix for more information.

Setting Page Width and Length

In order for the portrait and landscape orientations to work correctly, do not include the FORMAT directive options –LENGTH or –WIDTH in the environment file. The PostScript device handler adjusts the number of lines and columns available for text on a page automatically. If you set the –LENGTH and –WIDTH FORMAT directives, the PostScript device handler does not control the line formatting adjustments itself, which may cause long lines to wrap at the wrong place in one orientation or cause lines to continue off the bottom of the page when using SPOOL –SET_LANDSCAPE –AT NOSCALE.

If you include either the FORMAT directive –WIDTH or –LENGTH option, these settings will be used and not adjusted according to the orientation. This may be useful if you are printing on odd-sized paper, in which case you may want to create separate environments for different orientations. If you define separate environment files for each page orientation, do not include both SET_PORTRAIT and SET_LANDSCAPE directives in the same environment file.

Providing n-up Printing

n-up, or thumbnail, printing enables you to print multiple document pages on a single sheet of paper. This method of printing is useful for checking page layouts and saves paper when printing large files. Attributes for *n*-up page printing are available for both portrait and landscape page formats.

To provide all available *n*-up printing page layouts, include the following ATTRIBUTE directive in the environment file:

```
ATTRIBUTE 2–UP 4–UP 4X–UP 8–UP 8X–UP 16–UP 16X–UP
```

Users can apply this *n*-up capability to PostScript language files in addition to text files.

Refer to Tables D-4, D-5, and D-6 for the SPOOL command *n*-up attributes and to Figure D-1 for the *n*-up printing page layout.

Providing *n*-up Options for Special Effects: The following additional attributes can be defined for *n*-up printing:

ATTRIBUTE FLIP

Allows users to flip each document page upside down, and is primarily useful for PostScript files that produce landscape orientation using a 270-degree rotation instead of the ususal 90-degree rotation.

ATTRIBUTE NOBORDER

Allows users to disable the border outlining on each page of the miniature document pages that print by default. It also disables the job header and page number normally placed on each sheet.

ATTRIBUTE NOCLIP

Enables users to spool PostScript files that might put marks outside the normal boundaries of a document page. Clipping the image of each page to the page boundary is the default.

You can provide all of the above attributes in the environment file using a single ATTRIBUTE directive, as shown below:

```
ATTRIBUTE FLIP NOBORDER NOCLIP
```

Defining PostScript Fonts

You can define up to 32 fonts in the environment file by using the SET_FONT directive, in the following format:

```
SET_FONT –NAME fontname –SEQ 'printer_font' [size] –STRING [–DEFAULT]
```

fontname is the font's name that users supply with the –SET_FONT option on the SPOOL command line.

printer_font must be the exact name of the font within the printer, and may be optionally followed by a font *size*, separated by a space. You should place both the *printer_font* and *size* within single quotes. It is recommended that you enclose *printer_font* in single quotes even if you do not include *size.*

size is in units of PostScript points (1/72") and may be a real number, not just an integer. It defaults to 11.5 points if not supplied. Refer to the sample files shown in Sample Files for PostScript Printer Support later in this appendix.

The –STRING option is required for all SET_FONT directives within PostScript printer environment files.

When users request a different font size, the PostScript device handler attempts to dynamically adjust the number of columns per page according to the ratio of the new font size to the previous font size. It does not adjust the number of lines per page. For the best printed results, use a monospaced font such as Courier.

To get a complete list of font names for fonts resident in your printer, spool the following PostScript file to it:

```
%!PS-Adobe-2.1 Query
%%Title: queryfontlist.ps
%%?BeginFontListQuery
  FontDirectory { pop == } forall
  /filenameforall where
  {   pop
      save
      (fonts/*)
      { dup length 6 sub 6 exch getinterval == }
      128 string
      filenameforall
      restore
  } if
  flush
 %%?EndFontListQuery: Unknown
 %%EOF
```

Defining Attributes That Produce Special Effects

You can define the following additional attributes with the ATTRIBUTE directive to enable users to produce special effects on their printed output.

ATTRIBUTE REVHEAD

The REVHEAD attribute allows users to request that the per-page headline (which typically includes the first line of the file and the page number) be printed in reverse color (for example, white text on a black background) so it stands out. This attribute has no meaning for spooled PostScript language files.

ATTRIBUTE DRAFT/ [*string*]

The DRAFT/ attribute enables users to have a message such as *DRAFT* overprinted on each page of their document. You can choose messages other than DRAFT by appending them after the slash. For example:

ATTRIBUTE DRAFT/ DRAFT/SAMPLE DRAFT/TOP-SECRET

allows users to have the word *DRAFT*, *SAMPLE*, or *TOP-SECRET* overprinted in large, outlined letters on each page of their document. This capability applies not only to text files but also to PostScript language files as well. *string* is from 1 through 10 characters.

Uppercase letters are used in the example above because the Spooler subsystem converts attribute names to uppercase letters.

Keep in mind that each environment file may contain a maximum of 32 attributes. You could conceivably use half of the available attributes with the special attributes supported by the PostScript device handler, not including the numerous DRAFT/ variations you might want in addition.

Of the 32 available attributes, you must reserve enough attributes to define the printer type, location, and other attributes that define the printer operation and setup.

Also, remember that if you maintain Attributes files in the SPOOL*>ATTRIBUTES directory, they must be updated appropriately for the set of attributes specified in the PostScript printer's environment file.

PostScript Procedures

The SPOOL –PROC option allows users to request a PostScript procedure to handle their text jobs. When users request a procedure, the PostScript device handler no longer translates the contents of the spooled file into PostScript, but sends straight ASCII text to the printer treating it like other non-PostScript printers.

This allows you to write printer emulators (using PostScript) that can handle such data. Prime provides two basic line printer emulator sets in the PRIMEDICT.PS dictionary (described later):

- A basic no-frills set for 7-bit Prime ASCII files supporting linefeed (^J) for newline, carriage return (^M) for overprinting a line, and form feed (^L at the beginning of a line) to eject the page. This set is adequate to handle the way FORTRAN-formatted and COBOL-formatted files are translated by the despooler for printing.

- A more sophisticated set for 8-bit Prime-ECS files with enhanced control character support for tab (^I) to standard 8-column tab stops, backspace (^H) for overstriking, and form feed (^L) anywhere in a line.

The attributes ATTRIBUTE 2-UP, 4-UP, and 4X-UP can be used with the –PROC emulators.

The following options and attributes have no effect with –PROC procedures:

SET_PORTRAIT
SET_LANDSCAPE
ATTRIBUTE 8-UP 8X-UP 16-UP 16X-UP
ATTRIBUTE FLIP NOCLIP NOBORDER
ATTRIBUTE REVHEAD
ATTRIBUTE DRAFT/ (and any variations)
SET_FONT

These emulators are discussed in more detail in Using PostScript Dictionary Files below, and in the section called User's View of PostScript Printing later in this appendix. Refer also to Table D-8 later in this appendix for a list of the available SPOOL –PROC options.

There is no method for validating or restricting the use of SPOOL –PROC procedures (other than removing them entirely by not downloading the dictionary that defines them). If users request an incorrect –PROC procedure, a PostScript `undefined` error will result.

Using Electronic Vertical Forms With PostScript Printers

Unlike some printers, PostScript printers do not recognize the control characters embedded in EVFU files. To use electronic forms, you must configure the PostScript print handler software to emulate this functionality by including the –EVFU_OFF option of the DEVICE directive in your PostScript printer's environment file.

Using PostScript Dictionary Files

PostScript dictionary files are special PostScript programs that contain definitions of procedures or alter the printer state using the PostScript language. PostScript dictionaries do not normally print anything. Font definition files are special PostScript dictionary files that can be sent to the printer to supplement the standard set of fonts resident in the printer's ROM.

Downloading PostScript Dictionaries

Dictionaries are downloaded to a PostScript printer outside the batch server loop, which enables their definitions and settings to remain resident in the printer's memory for every print job. If dictionaries are not resident in the printer's memory, they must be included with every print job, which consumes a lot of time. Since this ability to exit the batch server contradicts the principle of job separation inherent in PostScript batch operation, you should exercise caution when using this feature.

For this reason, PostScript protects the use of this feature with a password called the exitserver password. exitserver is the name of the PostScript operator that exits the batch server loop. For most PostScript printers, the exitserver password is a number set by the factory for the printer; usually 0. Your printer manual should show how to change this password. If you feel compelled to change the printer's default exitserver password, you must inform the PostScript device handler about the new password using the –PASSWORD option to the DEVICE POSTSCRIPT environment file directive. You will also have to change the password in any dictionary files that Prime supplies or files that you create (see below) that use exitserver.

Caution Prime recommends that you leave the default exitserver password unchanged because if the new password is lost or forgotten, your printer may require hardware service to restore the default password. You can prevent users from changing the printer's password, either deliberately or accidently, by using the NOSETPASSWORD.PS dictionary file discussed later.

To make a dictionary remain resident, you precede it with the PostScript code:

```
%!PS-Adobe-2.1 ExitServer
%%BeginExitServer: 0
  0  % <-----------\-- exitserver password
  serverdict begin exitserver
%%EndExitServer
```

Dictionaries downloaded in this manner will remain resident in the printer's memory until the printer power is turned off.

Testing PostScript Dictionary Files

While you are developing and testing PostScript dictionaries for specific print jobs, you should remove the exitserver code from the dictionary file, and include the PostScript dictionary at the beginning of test files that use the dictionary definitions. Although it takes longer to send dictionaries to the printer with each print job, it prevents the printer from running out of memory, or from being left in an inconsistent state because of an error in your dictionary.

If you continue to download new versions of test dictionaries to the printer outside the batch server loop, the new versions do not overwrite the old versions, and the printer will ultimately run out of memory. The memory available for handling print jobs is also reduced if you inadvertently reload a dictionary that is already loaded.

Note If the printer runs out of memory while working outside of the batch server loop, the only way you can recover is to power cycle the printer (turn the printer's power switch OFF and then ON).

Managing PostScript Dictionaries

The PostScript device handler automatically manages your dictionaries for you with the –DICT option to the DEVICE POSTSCRIPT environment file directive. Not only does it check that the dictionaries are properly resident before printing each job, it prevents them from being downloaded more than once. Using the –DICT option, you do not need to worry about manually downloading dictionaries even if the printer is powered OFF and then ON again to correct a problem or recover from a power failure.

One consequence of turning the printer's power switch OFF and then ON again (also known as power cycling) to reload dictionaries is that if you want to add, delete, or modify any dictionary, you must power cycle the printer to cause a new dictionary configuration to be downloaded.

An alternative to power cycling the printer would be to send the following job to the printer. This would produce the same result:

```
%!PS-Adobe-2.1 ExitServer
%%Title: resetprinter.ps
%%BeginExitServer: 0
  0  % <------------\-- exitserver password
 serverdict begin exitserver
%%EndExitServer
  systemdict /quit get exec  % Reset printer just like power-up!
%%EOF
```

Jobs that are sent to the printer after this job finishes will execute with the new dictionary set resident.

Prime supplies three dictionaries in SPOOL* for use with the PostScript device handler. Each is included in the SPOOL*>POSTSCRIPT_INIT file.

- NOSETPASSWORD.PS prevents the printer's exitserver password from being altered, either accidentally or deliberately by a user's job. It does *not* prevent a job from using exitserver successfully if the exitserver password is correct.

- NUP.PS is the *n*-up printing support dictionary that works with the internal text-to-PostScript translator as well as most reasonably well-formed PostScript files. See the ATTRIBUTE *n*-up directives discussed in Providing *n*-up Printing earlier.
- PRIMEDICT.PS dictionary contains the following –PROC line printer emulators:

  (LP)NORMALz
  (LP)PORT
  (LP)PORT2
  (LP)PORT4
  (LP)PORT4X
  (LP)LAND
  (LP)LAND2
  LP)LAND4
  (LP)LAND4X

  Refer to Table D-7 later in this appendix for a list of the –PROC options. These procedures implement *n*-up in an entirely different way than NUP.PS, and are suitable only for spooling text files. –PROC does not support graphics.

Required Dictionaries for Using SPOOL Options

In order to provide support for users to use the –PROC options or the *n*-up printing options, the PRIMEDICT.PS and the NUP.PS dictionary files must be resident in the printer's memory.

You are free to add dictionaries of your own, for example, to support a corporate logo or corporate-standard font.

POSTSCRIPT_INIT may also contain files other than dictionaries that do not use the exitserver operator. For example, if you want to print a page containing the text *Downloaded Printer* whenever the Spooler downloads the dictionary set from POSTSCRIPT_INIT, you can add the following PostScript language file to the end of the files listed in POSTSCRIPT_INIT:

```
%!PS-Adobe-2.1
%%Title: loaded-printer.ps
%%EndComments
  /cshow % center string then show
  {   dup stringwidth pop 2 div neg 0 rmoveto show
  } def
  /pname % get printer's name, if unavailable use "Printer"
  {   statusdict /printername 2 copy known
      {   get 100 string exch exec
      }{  pop pop (Printer)
      } ifelse
  } def
%%EndProlog
%%Page: 1 1
  save
  72 72 scale % Set an inch scale
  /Helvetica findfont 0.7 scalefont setfont
  4.25 8 moveto (Downloaded) cshow
  4.25 7 moveto    pname      cshow
  restore
  showpage % Print the page
%%EOF
```

Sample Files for PostScript Printer Support

The following sections contain sample files that you can use as templates to create your own files for your particular printing environment.

Sample PostScript Environment File

```
/* POSTSCRIPT.ENV
/* Sample PostScript Environment File
/*
   DEVICE POSTSCRIPT -EVFU_OFF -DICT SPOOL*>POSTSCRIPT_INIT -PASSWORD 0 &
                     -POSTSCRIPT_CHECK
/*
/* Use either ASYNC or TCP/IP, but not both!
   ASYNC -LINE <line> -SPEED 9600 -CHAR_LENGTH 8 -XOFF -NO_ECHO -PRO TTY &
                     -NO_LINE_FEED
/* TCP/IP -ADDRESS <ip-address> -PORT <port>
/* Printer destination name:  You choose these.
   ATTRIBUTE POSTSCRIPT -MANDATORY    /* Users must SPOOL -AT POSTSCRIPT
/* Attributes supported by the PostScript device handler itself:
   ATTRIBUTE TEXT
   ATTRIBUTE 2-UP 4-UP 4X-UP 8-UP 8X-UP 16-UP 16X-UP
   ATTRIBUTE FLIP NOBORDER NOCLIP
   ATTRIBUTE REVHEAD DRAFT/ DRAFT/SAMPLE
   ATTRIBUTE NOSCALE
   ATTRIBUTE NOLOG
/* Format of paginated files:
/*      Do not include -LENGTH or -WIDTH to let handler automatically
/*      determine these based on orientation, font, and scaling.
   FORMAT -TOP_MARGIN 2 -BOTTOM_MARGIN 2
/* Paper Bins:  Almost all PostScript printers allow manual feed; if
/*              your printer supports other bins, they can be added.
   SET_PAPER_BIN -MANUAL
   SET_PAPER_BIN 1
/* Page Orientations:  If specified, one must be identified as -DEFAULT
   SET_PORTRAIT -DEFAULT
   SET_LANDSCAPE
/* Fonts:  This set is based on the Apple LaserWriter Plus.
/*         If desired, can identify a default choice with -DEFAULT;
/*         if none are so identified Courier (11.5pt) is assumed.
/*         -SEQ must contain the actual name of the font in the printer
/*         and can include an optional size separated by a space.  If
/*         no size is specified in -SEQ, it defaults to 11.5pt.
/* *** THERE CAN BE NO MORE THAN 32 SET_FONT DIRECTIVES TOTAL! ***
   SET_FONT -NAME AvantGarde          -SEQ 'AvantGarde'           -STRING
   SET_FONT -NAME Bookman             -SEQ 'Bookman'              -STRING
   SET_FONT -NAME Courier             -SEQ 'Courier'              -STRING
   SET_FONT -NAME Courier-10          -SEQ 'Courier 10'           -STRING
   SET_FONT -NAME Courier-12          -SEQ 'Courier 12'           -STRING
   SET_FONT -NAME Courier-Bold        -SEQ 'Courier-Bold'         -STRING
   SET_FONT -NAME Courier-Bold-10     -SEQ 'Courier-Bold 10'      -STRING
   SET_FONT -NAME Courier-Bold-12     -SEQ 'Courier-Bold 12'      -STRING
   SET_FONT -NAME Courier-BoldOblique -SEQ 'Courier-BoldOblique'  -STRING
   SET_FONT -NAME Courier-Oblique     -SEQ 'Courier-Oblique'      -STRING
   SET_FONT -NAME Helvetica           -SEQ 'Helvetica'            -STRING
   SET_FONT -NAME Helvetica-Narrow    -SEQ 'Helvetica-Narrow'     -STRING
   SET_FONT -NAME Palatino            -SEQ 'Palatino-Roman'       -STRING
   SET_FONT -NAME NewCentury        -SEQ 'NewCenturySchlbk-Roman' -STRING
   SET_FONT -NAME Symbol              -SEQ 'Symbol'               -STRING
   SET_FONT -NAME Times               -SEQ 'Times-Roman'          -STRING
   SET_FONT -NAME Times-Bold          -SEQ 'Times-Bold'           -STRING
   SET_FONT -NAME ZapfChancery  -SEQ 'ZapfChancery-MediumItalic'  -STRING
   SET_FONT -NAME ZapfDingbats        -SEQ 'ZapfDingbats'         -STRING
/* Header, Trailer, and Warning pages:
   HEADER 1
   TRAILER -OFF
   WARNING -ON
/* Message for header page:
   MESSAGE Our Company's PostScript Printer
/* Despooler Logging:
   LOG -ON -ERROR_NOTIFY OPERATOR OPERATOR@SYSB
/* Put users who should be notified of printer errors (paper jam) above.
/* Queue Shenanigans
   PRIORITY_TO_LOCAL_QUEUE
   NODE -LOCAL
```

Sample Environment File for a Printronix Printer

```
PARALLEL pr0
DEVICE printronix                         /* use printronix
ATTRIBUTE wide long
ATTRIBUTE line -mandatory
FORMAT -width 132 -length 65               /* specify default page
                                           /* layout

LOG -on
WARNING -on
HEADER 2                                   /* one header page
NODE sysa                                  /* look at sysa
NODE -local                                /* check local system
PRIORITY_TO_LOCAL_QUEUE                    /* check local more often
MIN_SIZE 1                                 /* don’t print empty files
```

Sample Attributes File

```
/* .DEFAULT_ATTRIBUTES.
/* Needed additions to local attributes files for PostScript features.
/*
/* PostScript -AT n-UP (need .NO_DEFAULT. to make 1-up the default!)
.NO_DEFAULT.
2-UP
4-UP
4X-UP
8-UP
8X-UP
16-UP
16X-UP
/* PostScript -AT DRAFT/ & friends
.NO_DEFAULT.
DRAFT/
DRAFT/SAMPLE
/* PostScript -AT FLIP
.NO_DEFAULT.
FLIP
/* PostScript -AT NOBORDER
.NO_DEFAULT.
NOBORDER
/* PostScript -AT NOCLIP
.NO_DEFAULT.
NOCLIP
/* PostScript -AT NOSCALE
.NO_DEFAULT.
NOSCALE
/* PostScript -AT REVHEAD
.NO_DEFAULT.
REVHEAD
/* PostScript -AT TEXT
.NO_DEFAULT.
TEXT
/* POSTSCRIPT -AT NOLOG
.NO_DEFAULT.
 NOLOG
```

Sample POSTSCRIPT_INIT File

```
/* POSTSCRIPT_INIT, SPOOLSRC>SPOOL*, SPOOLER PROJECT, 08/15/91
/* List of pathnames of PostScript dictionaries to be loaded.
/*
SPOOL*>NOSETPASSWORD.PS  /* Prevent jobs from changing password
SPOOL*>NUP.PS            /* For -AT n-UP thumbnail printing
SPOOL*>PRIMEDICT.PS      /* For -PROC line-printer emulators
/* More may be added below:
```

User's View of PostScript Printing

As the Spool Administrator, it is important for you to understand the user's PostScript printing needs and the process by which users spool files to PostScript printers so that you can respond effectively to their problems and questions. This section might also serve as a basis for training users at your location in the proper use of your PostScript printing environment that you can tailor to the configuration you establish.

After determining the printing needs of users, you can then use that information to configure the PostScript printers. The printer environment file, attributes files for users, groups, and default attributes, and the POSTSCRIPT_INIT file must be properly configured to enable users to take full advantage of the PostScript printing capabilities and make efficient use of the PostScript printers.

Formatting Files on PostScript Printers

The Spooler supports both non-PostScript text files (including FORTRAN format and COBOL format) and PostScript language files (those with a .PS filename extension). Refer to Tables D-1, D-2, and D-3 in the section Spooling Files to PostScript Printers, earlier in this appendix.

When processing text files, the Spooler's own internal print handler is capable of translating Prime-ECS text to PostScript in the following way:

- Form-feed (^L) characters start a new page.
- Tab (^I) expands characters into the required number of spaces to the next tab stop. Tab stops are spaced every eight columns by default (the interval may be changed with the embedded Spooler control sequence ^002^020 *tab-interval*).
- Overprinted text appears bold, similar to the way an impact printer overstrikes the same characters on the printer ribbon to achieve a bold type.

Refer to Chapter 4, the section Embedded Control Codes in Printable Files, for more information. You may also wish to compare the capabilities of the –PROC procedure emulators discussed later,

Selecting a Paper Bin

Users may request that their job be printed on paper from a certain paper bin using the SPOOL –SET_PAPER_BIN option. This option accepts a number from 1 through 9 indicating the bin from which paper should be fed. Users should check with their Spool Administrator to determine which are valid for their printers and the type of paper or forms loaded in each.

Users may also specify –SET_PAPER_BIN MANUAL to request manual-feed operation. Most PostScript printers become impatient after a few minutes waiting for manually fed input, and will abort the job, so users should be prepared to quickly minister to the printer's needs when their job is started.

SPOOL Formatting Options

The various SPOOL options provide users with a wide choice of file formatting options for PostScript devices. Most of these features apply primarily to spooled text files, not to spooled PostScript language files that handle their own formatting. However, some useful features that also apply to spooled PostScript language files are noted explicitly in the sections that follow.

Selecting Page Orientation

Users may choose portrait and landscape page orientations using the SPOOL command options listed below. Each orientation uses its own line and column dimensions that are manipulated dynamically by the Spooler for proper line wrapping (or truncation if the SPOOL –TRUNCATE option is used). Since PostScript files determine their own page orientation, the following options apply only to text files:

–SET_PORTRAIT
Selects portrait orientation capable of fitting 66 lines of 80 columns each.

–SET_LANDSCAPE
Selects landscape orientation in which the font is scaled to fit a full 66 lines of 138 columns each.

–SET_LANDSCAPE –AT NOSCALE

Selects landscape orientation using the same font scale as portrait to achieve 50 lines of 105 columns each.

The –AT NOSCALE attribute has no effect with –SET_PORTRAIT.

Using n-up Attributes

Users can select any of the *n*-up printing options with the –ATTRIBUTE option of the SPOOL command (providing that they have been defined in the environment file). *n*-up printing is useful for the following tasks:

- Proofing the page layout of a document without wasting paper
- Reducing paper usage in facilities that charge by the page
- Grouping several presentation slides onto a single sheet for a handout for audience members

Tables D-4, D-5, and D-6 give the SPOOL command *n*-up attributes for both portrait and landscape page formats:

Table D-4. N-up Page Attributes for –SET_PORTRAIT

–SET_PORTRAIT	*Orientation*	*Lines x Chars per Line*	*Font Size*	*Width x Height*
Default (1 page per sheet)	Portrait	66 x 80	11.5 pts	8.0" x 10.5"
–AT 2-UP	Portrait	66 x 80	6.7 pts	4.7" x 6.1"
–AT 4-UP *or* 4X-UP	Portrait	66 x 80	5.1 pts	3.6" x 4.7"
–AT 8-UP *or* 8X-UP	Portrait	66 x 80	3.4 pts	2.3" x 3.1"
–AT 16-UP *or* 16X-UP	Portrait	66 x 80	2.6 pts	1.8" x 2.3"

Table D-5. N-up Page Attributes for –SET_LANDSCAPE

–SET_LANDSCAPE	*Orientation*	*Lines x Chars per Line*	*Font Size*	*Width x Height*
Default (1 page per sheet)	Landscape	66 x 138	8.8 pts	10.5" x 8.0"
–AT 2-UP	Landscape	66 x 138	5.1 pts	6.1" x 4.7"
–AT 4-UP *or* 4X-UP	Landscape	66 x 138	3.9 pts	4.7" x 3.6"
–AT 8-UP *or* 8X-UP	Landscape	66 x 138	2.6 pts	3.1" x 2.3"
–AT 16-UP *or* 16X-UP	Landscape	66 x 138	1.9 pts	2.3" x 1.8"

Table D-6. N-up Page Attributes for –SET_LANDSCAPE –NOSCALE

–SET_LANDSCAPE –AT NOSCALE	*Orientation*	*Lines x Chars per Line*	*Font Size*	*Width x Height*
Default (1 page per sheet)	Landscape	50 x 105	11.5 pts	10.5" x 8.0"
–AT 2-UP	Landscape	50 x 105	6.7 pts	6.1" x 4.7"
–AT 4-UP *or* 4X-UP	Landscape	50 x 105	5.1 pts	4.7" x 3.6"
–AT 8-UP *or* 8X-UP	Landscape	50 x 105	3.4 pts	3.1" x 2.3"
–AT 16-UP *or* 16X-UP	Landscape	50 x 105	2.6 pts	2.3" x 1.8"

When using *n*-up printing, each sheet of paper is divided into 2 to 16 equally sized sections, and each section contains 1 formatted page.

n indicates the number of document pages shrunk to fit on a single sheet of paper. The optional *X* following *n* indicates the order in which the pages are sequenced on the sheet of paper. The pages are sequenced in columns or in rows. For example, 4-UP prints page 2 under page 1, while 4X-UP prints page 2 to the right of page 1, as shown in Figure D-1.

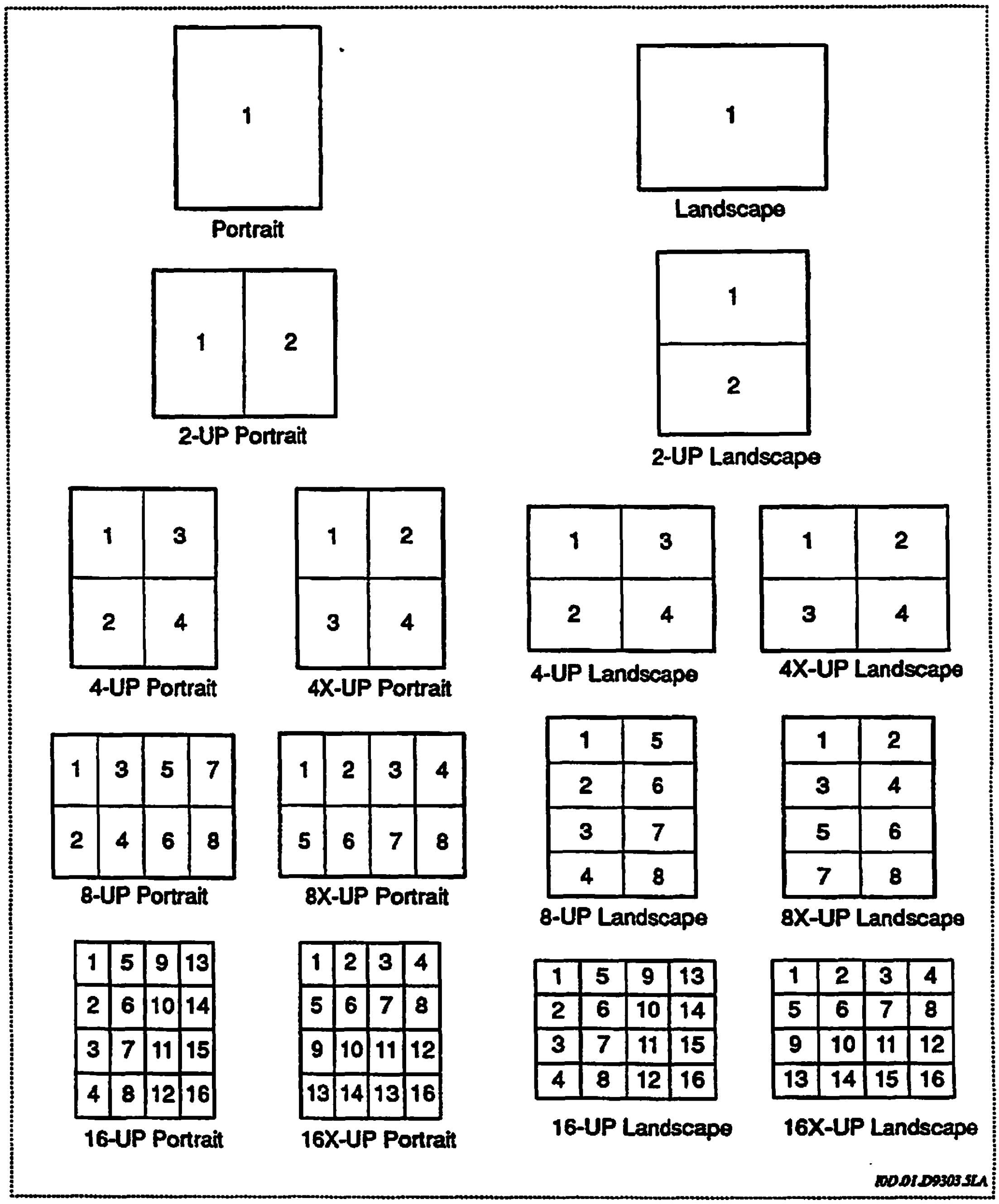

Figure D-1. PostScript n-up Page Layouts for 8.5 x 11 Inch Sheets

These options can be applied not only to spooled text files but also to spooled PostScript files. PostScript files should conform to Adobe's Document Structuring Conventions, although *n*-up can be accomplished with mildly nonconforming PostScript files.

When spooling a PostScript file that prints landscape, you must use SPOOL –SET_LANDSCAPE to use the *n*-up printing features so that the document pages are sequenced correctly on the page. For a 270-degree rotated landscape page (used by some software that generates PostScript output), you may also need to also use –AT FLIP.

In certain circumstances, *n*-up attributes may also apply when using the –PROC emulators. Refer to the later section Selecting a PostScript Procedure for more information on –PROC emulators.

Using *n*-up Special Effects Attributes: The following attributes are used to control special effects of *n*-up printing:

–AT FLIP

Flips document pages upside down. This is primarily useful for PostScript files that effect landscape through a 270-degree rotation rather than the more normal 90-degree rotation. PostScript files generated on an Apple Macintosh are such an example.

–AT NOBORDER

When used in combination with one of the *n*-up attributes, this attribute disables printing of the borders that outline each of the scaled pages (by default). It also disables printing of the job header and page number normally printed at the top of each sheet.

–AT NOCLIP

When used in combination with one of the *n*-up attributes, this attribute disables the image clipping that is performed (by default) to keep each image within its own page boundary. –AT NOCLIP would allow the contents of one document page to potentially bleed into the area of another document page. If your printed document appears to be missing text or part of an image, you could use –AT NOCLIP to find out if the text or image was written outside of the page boundary, which might happen if you are experimenting with page layouts, enlarged text, or some other page formatting feature. PostScript will attempt to write outside of the page boundary if the page format does not allow enough room to print the text or the image.

Selecting Fonts

The Spool Administrator can define up to 32 fonts in the printer environment file using the SET_FONT environment directive. Users can then select a font using the following SPOOL command option:

–SET_FONT *fontname*

Selects a font for printing a text file. *fontname* may also indicate the point size of the font. The default is 11.5 point Courier.

Refer to your printer documentation or check with your Spool Administrator for more information on available fonts for your particular printer.

Determining Font Size

The term **point size** is used by the publishing industry to indicate 1/72 of an inch. The default point size of 11.5 enables the best legibility while still permitting a full 66 lines of 80 columns each with reasonably set paper margins.

For best results, you should choose a monospaced font such as Courier. Proportionally spaced fonts may not produce the desired result. **Monospaced** fonts allow the same space for each character, whereas **proportionally spaced** fonts change spacing according to the width of the character. For example, the character *m* would be given more space than the character *t* or *r*.

If you select a font with a point size other than the 11.5 default, the number of characters on each line automatically adjusts to compensate for the larger or smaller font size. For example, when spooling a file with a larger font size, fewer characters fit on the line, which may cause a line to wrap to the next line (or to truncate if the SPOOL –TRUNCATE option was used).

Selecting n-up Special Effect Options

Users can select special options for *n*-up printing, provided that the Spool Administrator has defined them in the environment file.

–AT REVHEAD

This attribute highlights the Spooler-generated, per-page headline that is printed at the top of each page (which typically includes the first line of the file and the page number) by printing it in reverse color: white text on a black background.

Note that the Spooler does not provide such headlines when spooling –NO_FORMAT (–AT TEXT), or for FORTRAN-formatted or COBOL-formatted files.

This option has nothing to do with the per-sheet headline provided when using the *n*-up capability. See –AT NOBORDER discussed earlier.

–AT DRAFT/ [*string*]

This attribute enables you to print the word *DRAFT* (or other string that the Spool Administrator has defined in the environment file) overprinted in large, outlined letters on each page of the spooled document. You should check with the Spool Administrator for other words or short phrases that may be defined in the environment file and available for use. For example, if the DRAFT/ attribute in the environment file is defined with the string TOP-SECRET, you could select that wording with the attribute –AT DRAFT/TOP-SECRET.

This capability can be applied to both text and PostScript language files.

Selecting a PostScript Procedure

In addition to the standard internal text-to-PostScript translator built into the Spooler and its PostScript device handler, you can use the SPOOL –PROC option to select an alternate printer emulator. When using –PROC, the Spooler bypasses its internal text-to-PostScript translator and sends its formatted ASCII text directly to the printer, operating as if it were a non-PostScript printer.

The –PROC procedures themselves are really PostScript programs that make the PostScript printer behave like a standard line printer. (Thus, it makes no sense to spool a PostScript language file using –PROC, unless you want a listing of this file rather than have it interpreted as PostScript.)

Prime provides two sets of printer emulator procedures to use with –PROC as shown in Table D-7.

Table D-7. –PROC Options for Use With SPOOL

–PROC Option Set		*Description*
NORMAL PORT PORT2 PORT4 PORT4X	 LAND LAND2 LAND4 LAND4X	A standard emulator with nine variations that supports 7-bit Prime ASCII files that use linefeed (^J) for newline, carriage return, (^M) for overprinting (bold), and form feed (^L at the beginning of a line) to eject the page. This set is adequate for most text files including FORTRAN and COBOL files.
LPNORMAL LPPORT LPPORT2 LPPORT4 LPPORT4X	 LPLAND LPLAND2 LPLAND4 LPLAND4X	This set, prefixed with LP, supports all of the standard set features plus the following additional features: improved support for 8-bit Prime-ECS, tab (^I) to standard 8-column tab stops, backspace (^H) in addition to carriage return, (^M) for overprinting (bold), and form feed (^L) anywhere in a line to eject a page.

The –PROC emulators shown Table D-7 provide their own support for *n*-up printing; this support is implemented in an entirely different way than the *n*-up printing discussed earlier. However, as a convenience, the Spooler will translate a SPOOL command using both –PROC and –AT *n*-up into the appropriate emulator. For example:

```
SPOOL myfile -PROC LAND -AT 4X-UP
```

is transformed by the Spooler into

```
SPOOL myfile -PROC LAND4X
```

Note that the emulator sets do not support 8-up, 8X-up, 16-up, or 16X-up printing. The only features supported by the –PROC emulator sets are listed below:

```
–AT 2-UP
    4-UP
    4X-UP
```

The –PROC emulators do not support the following SPOOL options:

```
–SET_PORTRAIT
–SET_LANDSCAPE
–SET_FONT
–AT 8-UP
    8X-UP
    16-UP
    16X-UP
    FLIP
    NOCLIP
    NOBORDER
    REVHEAD
    DRAFT/ (and any variants)
```

Table D-8 shows the line and column dimensions, as well as other parameters. Refer to the section Spooling Files to PostScript Printers earlier in this appendix and to Tables D-1, D-2, and D-3 for the text-to-PostScript internal translator options.

Table D-8. PostScript Procedure Options for n-up Printing

–PROC Option	*Orientation*	*Lines x Chars per Line*	*Font Size*	*Width x Height*
–PROC (LP)NORMAL	Portrait	66 x 80	12.0 pts	8.2" x 10.8"
–PROC (LP)PORT	Portrait	66 x 90	10.0 pts	7.5" x 10.0"
–PROC (LP)PORT2	Portrait	66 x 90	6.5 pts	5.0" x 6.5"
–PROC (LP)PORT4	Portrait	66 x 90	4.7 pts	3.7" x 4.8"
–PROC (LP)PORT4X	Portrait	66 x 90	4.7 pts	3.7" x 4.8"
–PROC (LP)LAND	Landscape	66 x 140	8.5 pts	10.0" x 7.5"
–PROC (LP)LAND2	Landscape	66 x 140	5.5 pts	6.5" x 5.0"
–PROC (LP)LAND4	Landscape	66 x 138	4.1 pts	4.8" x 3.7"
–PROC (LP)LAND4X	Landscape	66 x 138	4.1 pts	4.8" x 3.7"

Reporting Errors to Users

While the Spooler is processing each user's job, any errors that the printer reports related to that job are summarized by the Spooler on a separate PostScript Error Log page following the end of the spooled job. Thus, when users pick up their output this page should provide some evidence as to why their job didn't print completely as expected. This error log may also prove helpful to Spool Administrators in identifying the approximate cause of the problem.

The SPOOL option –AT NOLOG suppresses the printing of a PostScript Error Log after the job is completed or aborted. You might use this if you suspect that your job might cause an error or produce some message that you wish to ignore.

Defining HP LaserJet Printers

E

■ ■ ■ ■ ■ ■ ■

Introduction

The following Spooler features are available for use with the HP LaserJet family of printers:

- Landscape page formatting
- Portrait page formatting
- Font sizes of 10, 12, and 16.66 (although not all printers support all three font sizes)

Environment File Directives

To use these features, the Spool Administrator needs to create (or modify) a printer environment file, which must include the following directives.

- DEVICE HPJET [–DELAY *nnn*]

  –DELAY is optional. *nnn* is the number of seconds that printing is delayed between each page. This is useful to prevent buffer overflows at the printer. The default is four seconds. You can set the value of *nnn* to a lesser or greater number of seconds, depending on the particular printer.

- SET_LANDSCAPE
- SET_PORTRAIT

  You must include both of these entries if you wish to use landscape and portrait page formatting.

- Attributes for defining the paper size and font are shown in Tables E-1 and E1-2. Notice that HP LaserJet font sizes are measured in characters per inch (for U.S. fonts) and characters per centimeter (for European fonts), rather than in point size.

Page Formatting ATTRIBUTES for HP LaserJet Printers

You must include one or more ATTRIBUTE *name* directives in the environment file to define the paper size and fonts available for your HP LaserJet printer. The available *name* values are shown in Tables E-1 and E-2.

Table E-1. Attributes for U.S. Paper Sizes			*Portrait*		*Landscape*	
Name	*Paper Size*	*Chars per Inch*	*Chars per Line*	*Lines per Page*	*Chars per Line*	*Lines per Page*
LETTER	8.5" x 11"	10	80	60	106	46
LETTER_MED	8.5" x 11"	12	96	82	127	62
LETTER_SMALL	8.5" x 11"	16.66	132	82	176	62
LEGAL	11" x 14"	10	80	78	134	46
LEGAL_MED	11" x 14"	12	96	105	163	62
LEGAL_SMALL	11" x 14"	16.66	132	105	226	62

Table E-2. Attributes for European Paper Sizes			*Portrait*		*Landscape*	
Name	*Paper Size*	*Chars per Centimeter*	*Chars per Line*	*Lines per Page*	*Chars per Line*	*Lines per Page*
A4	210x297mm	3.90	77	64	112	44
A4_MED	210x297mm	4.68	93	87	135	59
A4_SMALL	210x297mm	6.50	129	87	188	59
B5	182x257mm	3.90	66	54	97	37
B5_MED	182x257mm	4.68	80	74	116	50
B5_SMALL	182x257mm	6.50	111	74	167	50

PDN Support Directive

F

Introduction

The PDN support software allows you to direct your output to printers attached to Public Data Network (PDN) services in the same way as you can direct output to printers attached to asynchronous lines. It is an optional feature of the Spooler subsystem that you purchase separately.

When Would I Need the PDN Support Directive?

Suppose that you have a central office that houses your main computer system, and one or more branch offices that have terminals to the system. The terminals can connect to the computer on dialup connections, or through a PDN, such as the American TELENET®, the Canadian Datapac®, or the British IPSS. The same offices may often need to print files. The Spooler PDN support directive allows you to connect a printer at a remote site to a Public Data Network so that it can receive and print data sent from the central system.

To direct a print job to the remote printer, you would issue a SPOOL command and specify, with the –ATT option, whatever options the System Administrator had set up to select the remote printer. This is the same procedure as for any other printer environment, as described in Chapter 1, Introducing the Spooler Subsystem.

If the PDN directive were not available, you would need to attach the remote printer to an expensive dedicated printer line, or attach a small printer directly to the terminal, and accept a very much slower and poorer quality printing service.

What Other Software Is Needed?

To control a printer through a PDN using the Spooler subsystem, you should install PRIMENET with X.25 support. If you have remote terminals that use

X.25 multiplexers to attach to the Prime system, or if you have two or more Prime computers in a network, you must already have PRIMENET.

Installing the PDN Support Directive

The supplied tape contains a new library routine and an INFO directory. You should read the files in the INFO directory before installing the new software, in case the version that you obtain needs to be installed differently than described below.

The steps in the installation are as follows:

1. Restore the contents of the tape into a working directory.
2. Copy the file working-dir>LIBRARIES*>SP$PDN_SUPPORT.RUN to LIBRARIES*>SP$PDN_SUPPORT.RUN.
3. Edit the search rules file SPOOL*>ENTRY$.SR to include the line LIBRARIES*>SP$PDN_SUPPORT.RUN. Edit the system search rules file SEARCH_RULES*>ENTRY$.SR to include LIBRARIES*>SP$PDN_SUPPORT.RUN.

Using the PDN Support Directive

When the system is correctly set up, the printer environments for remote printers are used in the same way as any other printer environment. The environments are controlled with PROP commands by privileged users of the system. However, if there is no work in any of the spool queues for a despooler phantom that controls a remote printer, the despooler disconnects the PDN link and becomes idle. When you request a print for the remote printer, it is put into a spool queue and the despooler phantom is notified. It then connects to the printer, via the PDN, and remains active until it again has no work to do.

Creating Printer Environments That Use PDN

The despooler phantom that controls a remote printer is controlled with the PROP command in the same way as any other despooler phantom. There is one additional directive, PDN, that you must use in the environment file. Apart from this, you create and use the environment file for a remote printer in the same way as any other environment file, as described in Chapter 3, Defining Printer Environments.

Before you can use a printer environment that sends data over the PDN, there must be access to the PDN from your system. If there is not, either you, or the System Administrator, must use PRIMENET's CONFIG_NET command to configure the PDN. The configuration process determines the synchronous line numbers that correspond to PDN addresses, number of virtual circuits permitted, packet size, and so on. For full details, see the *Network Planning and Administration Guide*.

If the network is correctly set up, you must then find out

- What kind of multiplexer/PAD the remote printer is connected to
- The PDN address of the printer
- Which X.25 facilities you need, or wish to use

The PDN address consists of a maximum of 14 digits and can be given in decimal or octal. It can include a full Data Network Identification Code (DNIC) if necessary. A full PDN address consists of a DNIC field, a network address, and a port-number or subaddress. In some multiplexers, the PDN address is sufficient to connect the host system to the exact port on the multiplexer used by the remote printer. With other multiplexers, the PDN address may connect only to the Packet Assembler/Disassembler (PAD) itself, and further information must be supplied to select the port. Sometimes the PAD requires a sign-on message for security reasons. See NETLINK in the *PRIMENET Guide*, for further PDN address details.

PDN Environment File Directive

This directive is used in environment files for remote printers. It specifies the address of a printer connected to a PAD, which in turn is connected to a PDN. The options are similar to the corresponding options of the NETLINK command. The command is part of the Spooler PDN support directive, and is not available with the standard Spooler subsystem.

Syntax is as follows:

PDN ***address*** **[*options*]**

You must specify the address in numeric format. It can be a maximum of 14 digits, for example:

`PDN 63300089 -MPAC 3` (Memotec-connected printer on port 3)

Options to the PDN directive are as follows:

Option	*Description*
–MPAC *port*	Use this option if the printer is connected to a Memotec 2000 multiplexer. *port* is the one- or two-character Memotec port number.
–GANDALF_PIN *service_id* –PIN	Use this option if the printer is connected to a Gandalf Pin™ multiplexer. *service-id* is the one- or two-character PIN service ID.
–DYNAPAC *port* –DYNA	Use this option if the printer is connected to a Dynapac™ X.25 multiplexer. *port* is the Dynapac port number.

The options –MPAC, –PIN, and –DYNA show that a specific type of PAD is being used, and allow the despooler to configure some of the connection parameters automatically.

For any other PDN, use the –STANDARD option. Note that if you are connecting to a TELENET PAD, both the PAD and the printer should be configured for no parity. The syntax of the –STANDARD option is as follows:

```
           ┌–PRID            ┐ ┌nnn nnn nnn nnn┐
           │–FCTY            │ └nnn . . . nnn  ┘
–STANDARD  │–UDATA           │
–STD       │┌–SIGN_ON_MSG┐   │ text
           └└–SOM        ┘   ┘ text
```

The way to use the suboptions to the –STANDARD option of the PDN directive is outlined below:

Suboption	*Description*
–PRID [*nnn nnn nnn nnn*]	Sets the protocol identification field data. It specifies in decimal, the four bytes of the protocol identification field. If the four bytes are omitted, the default, 001 000 000 000, is used. Further details are in the *PRIMENET Guide* and your PAD/multiplexer manual.

Suboption — *Description*

–FCTY [*nnn. . . nnn*]

Sets the call request packet facilities. Each parameter is a decimal number in the range 0–255. You can specify up to a maximum of 64 parameters. Parameters are in pairs that correspond to CCITT standards X.3 and X.29. If you give the suboption with no parameters, no facilities are specified. If you omit the –FCTY suboption, the default facilities of 001, 001, 002, and 007 are specified. For more information and details of facilities codes, see the *PRIMENET Guide.*

–UDATA *text*

Some PADs have several ports, that is, they are multiplexers, but the PDN address in the PDN command only connects to the PAD itself. To select the correct port for the remote printer, you must supply some further data. The –UDATA suboption allows you to supply a string that consists of a maximum of 12 characters. For details of the data to supply with this suboption, see the manual for your multiplexer or PAD.

–SIGN_ON_MSG *text*
–SOM

Permits you to supply a sign-on message to the remote PAD/multiplexer. Some PADs require the first packet of data received to be a particular sign-on message. The connection between the host computer and remote printer is made using the information supplied in the PDN address and the other options of the PDN commands. The first data packet transmitted is then checked by the PAD/multiplexer. If the first packet is the correct sign-on message, the connection continues and further data can be sent to the printer. If the sign-on message is incorrect, the connection is severed. For more information, see the manual for your PAD/multiplexer.

Below is an example of a PDN address that uses some of the –STANDARD suboptions:

```
PDN 123456789012 –STD –PRID 001 000 000 000 –FCTY 12 1 –UDATA 3
```

where

- –PRID is the standard PRIMENET protocol ID.

- –FCTY turns XON/XOFF.
- –UDATA indicates the port number on the PAD you wish to use.

Error Messages

When you use the PDN support directive, messages can also arise from the networking software. For full details, see the *PRIMENET Guide*.

```
Access bar
Address is unknown
Bad facility field in users call
Bad parameter(s)
Behind another user in the subprc assign list
Callee refuses collect call
Circuit is cleared
Circuit is idle
Circuit is reset
Clear cause is xxx
DTE busy
DTE clearing
DTE out of order
Destination node not operating
Diagnostic is xxx
Host is down
Illegal address
Illegal or unknown address
Invalid call
Local procedure error
Maximum number of pending operations exceeded
Network congestion
Network server logged out
Networks not configured
No call requests pending
No remote users
No virtual circuits left to accept call
Not enough memory for operation
Not obtainable
Operation complete
Operation illegal now
Operation in progress
Port not assigned
Remote procedure error
Route through protocol error
Route through server down
Routed through call request looping
Routed through circuit timeout
Routed through not enough memory
```

```
System not up yet
Timeout on call request
Timeout on clear request
Timeout on reset request
Unknown clearing cause
Unknown diagnostic message
Unknown error status
Unknown error status <EEE>
User does not own virtual circuit
```

Index

Index