Installation/User's Guide: DECprint™ Utility for PostScript® to Sixel Printing for VMS

Order Number: AA–NH56A–TE

This manual describes how to install and use the DECprint Utility for PostScript to Sixel Printing for VMS software on a VAX/VMS computer system.

Revision/Update Information:	This is a new manual.
Operating System and Version:	VAX/VMS, Versions 5.1, 5.2, and 5.3
Software Version	DECprint Utility for PostScript to Sixel Printing for VMS, Version 1.0

Digital Equipment Corporation

January 1990

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation.

Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

Any software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license. No responsibility is assumed for the use or reliability of software or equipment that is not supplied by Digital Equipment Corporation or its affiliated companies.

Restricted Rights: Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

© Digital Equipment Corporation 1990. All rights reserved.

Printed in U.S.A.

The postpaid Reader's Comments forms at the end of this document request the user's critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DEC	DIBOL	UNIBUS
DEC/CMS	EduSystem	VAX
DEC/MMS	IAS	VAXcluster
DECnet	MASSBUS	VMS
DECsystem–10	PDP	VT
DECSYSTEM–20	PDT	
DECUS	RSTS	
DECwriter	RSX	digital™

PostScript is a registered trademark of Adobe Systems, Incorporated.

MLO-S1325

Contents

PREFACE xi

I INSTALLATION GUIDE

CHAPTER 1 BEFORE INSTALLATION 1–1

1.1 OVERVIEW 1–1
1.1.1 PostScript Page Description Language 1–3
1.1.2 Sixel Graphics 1–3

1.2 PRINTER SUPPORT 1–3

1.3 HOST HARDWARE REQUIREMENTS 1–4

1.4 HOST SOFTWARE REQUIREMENTS 1–4

1.5 SOFTWARE DISTRIBUTION KIT 1–4

1.6 SOFTWARE LICENSE 1–4

1.7 REQUIRED DISK SPACE 1–5

1.8 INSTALLATION TIME 1–5

1.9 INSTALLATION VERIFICATION PROCEDURE (IVP) REQUIREMENTS 1–5

1.10 INSTALLING THE DECPRINT UTILITY WITH THE LN03 IMAGE SOFTWARE 1–5

CHAPTER 2 INSTALLING THE SOFTWARE 2–1

2.1 INSTALLATION CHECKLIST 2–1

2.2 VMSINSTAL CONVENTIONS 2–1

2.3 VMSINSTAL RELEASE NOTES 2–2

2.4 INSTALLATION PROCEDURE 2–2

2.5 RUNNING THE IVP AFTER INSTALLATION 2–7

2.6 SAMPLE INSTALLATION SESSION 2–8

2.7 IVP SAMPLE OUTPUT 2–9

CHAPTER 3 POSTINSTALLATION REQUIREMENTS 3–1

3.1 CREATING LOGICAL NAMES FOR QUEUED PRINTING 3–1

3.2 ENABLING OTHER NODES ON VAXCLUSTER SYSTEMS 3–3

3.3 HOST AND PRINTER CHARACTERISTICS 3–4

II USER'S GUIDE

CHAPTER 4 USING THE DECPRINT UTILITY COMMAND 4–1

4.1 THE COMMAND LINE 4–2
4.1.1 The DECprint Utility Command and Queued Printers 4–2
4.1.2 The DECprint Utility Command and Attached Printers 4–3

4.2 DEVICE SUPPORT 4–5

4.3 DECPRINT UTILITY COMMAND QUALIFIERS 4–6
4.3.1 /ATTACHED Qualifier 4–6
4.3.2 /LAYUP Qualifier 4–7
4.3.3 /MEDIUM Qualifier 4–8
4.3.4 /MESSAGES Qualifier 4–9
4.3.5 /ORIENTATION Qualifier 4–10
4.3.5.1 Page Orientation and /LAYUP • 4–11

4.4 PRINT QUALIFIERS THAT WORK DIFFERENTLY 4–12
4.4.1 /AFTER Qualifier 4–12
4.4.2 /PARAMETERS Qualifier 4–13

4.5 UNSUPPORTED PRINT QUALIFIERS 4–13
4.5.1 Formating Qualifiers 4–13
4.5.2 File Selection Qualifiers 4–13
4.5.3 /FORM and /SETUP Qualifiers 4–14
4.5.4 Miscellaneous Qualifiers 4–14

4.6 SUBMITTING DECPRINT UTILITY COMMANDS FOR BATCH PROCESSING 4–15

4.7 COLOR PRINTING WITH DECPRINT UTILITY 4–18

4.8 MISCELLANEOUS HELPFUL HINTS 4–18

4.8.1 Using Transparencies with the LJ250 Companion Color Printer 4–18
4.8.2 Printing PostScript Documents From MAIL 4–19
4.8.3 Running the DECprint Utility in the Background 4–19
4.8.4 Printer Margin Information 4–20

4.9 SUMMARY OF DECPRINT UTILITY QUALIFIERS 4–20

CHAPTER 5 SYSTEM MESSAGES 5–1

5.1 MESSAGE FORMAT 5–1

5.2 SEVERITY LEVEL 5–2

5.3 MESSAGE DESCRIPTIONS 5–2

APPENDIX A CREATING A PRINT QUEUE FOR THE DECPRINT UTILITY A–1

A.1 RENAMING AND EDITING THE PRINTER STARTUP COMMAND FILE A–2
A.1.1 Renaming the Startup File A–2
A.1.2 Editing the Startup File A–2

A.2 UPDATING THE SYSTEM STARTUP FILE A–4

A.3 PSPRINT$STARTUP QUEUE SETUP TOPICS A–4
A.3.1 Device Control Library A–4
A.3.2 Terminal Device Type A–5
A.3.3 Separation Pages A–5
A.3.4 /PROCESSOR Qualifier A–5

APPENDIX B POSTSCRIPT LANGUAGE EXTENSIONS B–1

B.1 USING POSTSCRIPT EXTENSIONS B–3

B.2 PERSISTENT AND VOLATILE PARAMETERS B–3

B.3 IDENTIFICATION EXTENSIONS B–4
B.3.1 Product B–4
B.3.2 Device Components B–5
B.3.3 Version B–5
B.3.4 Revision B–5
B.3.5 Job Name B–5

B.4 PAGECOUNT AND SHEETCOUNT OPERATORS B–6

APPENDIX C COLOR EXTENSIONS FOR THE POSTSCRIPT LANGUAGE C–1

C.1 ABOUT THE POSTSCRIPT LANGUAGE COLOR EXTENSIONS C–1

C.2 ABOUT THIS APPENDIX C–1

C.3 INTRODUCTION C–2

C.4 NEW FEATURES C–3
C.4.1 CMYK Color Specification C–3
C.4.2 Black Generation and Undercolor Removal C–4
C.4.3 Direct CMYK Color Specification C–6
C.4.4 Color Screens, Transfer Functions, and Images C–7
C.4.4.1 **colorimage** • C–7
C.4.4.1.1 Data formats for **colorimage** operator • C–8
C.4.4.1.2 Examples for **colorimage** operator • C–9
C.4.5 Color Implementations C–12

C.5 OPERATORS C–13
COLORIMAGE C–14
CURRENTBLACKGENERATION C–16
CURRENTCMYKCOLOR C–17
CURRENTCOLORSCREEN C–18
CURRENTCOLORTRANSFER C–19
CURRENTUNDERCOLORREMOVAL C–20
SETBLACKGENERATION C–21
SETCMYKCOLOR C–22
SETCOLORSCREEN C–24
SETCOLORTRANSFER C–27
SETUNDERCOLORREMOVAL C–29

APPENDIX D TROUBLESHOOTING PROCEDURES D–1

D.1 YOUR DECPRINT UTILITY JOB DOES NOT PRINT D–1

D.2 YOUR JOB REPORTS UNDEF ERROR MESSAGES D–1
D.2.1 Unsupported File Extensions D–1
D.2.2 Unsupported PostScript Operators D–2
D.2.3 Problems With Files Containing CTRL/D D–2

D.3 TROUBLESHOOTING COLOR PRINTING PROBLEMS D–3
D.3.1 Your Color Output is Black D–3
D.3.2 Thin Lines Don’t Print D–3
D.3.3 Incorrect Colors D–3

D.4 YOUR SETUP MODULES DO NOT WORK D–4

D.5 /PARAMETERS=(NUMBER_UP=2) DOES NOT WORK D–4

D.6 UNEXPECTED PAGE ORIENTATION D–4

D.7 DECPRINT UTILITY JOBS OCCUPY YOUR TERMINAL FOR TOO LONG D–5

D.8 FAINT DIAGONAL LINES WITH A WRITE-WHITE ENGINE D–5

D.9 THE POSITION OF THE IMAGE IS WRONG D–5

GLOSSARY Glossary–1

INDEX Index–1

EXAMPLES

2–1 Sample Installation Session 2–8
3–1 Defining Device Type Logical Names 3–3
4–1 Example with /ATTACHED 4–7
4–2 Examples with /LAYUP 4–8
4–3 Example with /MEDIUM 4–9
4–4 Examples with /MESSAGES 4–10
4–5 Examples Using the Batch-PSPRINT.com File 4–16
4–6 The Batch-PSPRINT Command File 4–17
A–1 Sample Execution Queue A–3
B–1 Determining Whether a Name Is Defined B–2
B–2 Accessing Extensions Located in statusdict B–3

FIGURES

1–1 Printer Attached to a Terminal 1–2
1–2 Printers on a Queue 1–2
2–1 DECprint Utility IVP Output 2–11
4–1 Printing Your Portrait File 4–11
4–2 Printing Your Landscape File 4–12

TABLES

3–1	Supported Devices	3–3
4–1	Attached Printers and Control Characters	4–4
4–2	Media Keywords	4–8
4–3	Printer Margins in Inches	4–20
4–4	DECprint Utility Qualifiers	4–21
4–5	Qualifiers Supported for Attached Printers	4–22
B–1	Persistent Parameters	B–4
B–2	Volatile Parameters	B–4

Preface

Intended Audience

The *Installation/User's Guide: DECprint™ Utility for* POSTSCRIPT® *to Sixel Printing for VMS* is intended for the following users:

- System managers who will install the DECprint Utility for POSTSCRIPT to Sixel Printing for VMS software
- Users who want to print POSTSCRIPT files on sixel printers

Document Structure

This manual contains the following parts, chapters, and appendixes:

Part I — Installation Guide

- Chapter 1 provides an overview and describes preinstallation requirements for the DECprint Utility software.
- Chapter 2 explains how to install the software.
- Chapter 3 describes postinstallation requirements.

Part II—User's Guide

- Chapter 4 explains how to use the DECprint Utility command to print POSTSCRIPT files on your ANSI/sixel printer.
- Chapter 5 describes the DECprint Utility system messages and, if necessary, recovery procedures.
- Appendix A explains how to create a print queue to use with the DECprint Utility command.

- Appendix B describes POSTSCRIPT operators specific to the DECprint Utility software.
- Appendix C describes POSTSCRIPT color operators.
- Appendix D describes troubleshooting steps.

Conventions

This document uses the following conventions:

Convention	Meaning
UPPERCASE notation	In command format, type the word or letter exactly as shown.
lowercase notation	In command format, substitute a word or value of your choice.
Boldface	**Boldface** words and letters used in formats, examples, and text specify POSTSCRIPT keywords. In page descriptions, you should type the keyword exactly as shown.
Italics	When *italic* words and letters appear in command formats, substitute an appropriate word or value for the word or letter.
[ ]	Square brackets in command formats indicate that the enclosed item is optional.
{ }	Braces in command formats enclose lists from which you must choose one alternative. The choices are listed vertically or separated by the vertical bar symbol (|).
...	A horizontal ellipsis in command formats indicates that the preceding item(s) can be repeated one or more times.
. . .	A vertical ellipsis in examples indicates that code has been omitted.
Ctrl/X	This symbol indicates that you should press the key labeled Ctrl while you simultaneously press another key, for example, Ctrl/Z, Ctrl/C, Ctrl/O.
Return	This symbol indicates that you should press the Return key.

Associated Documents

The installation procedures are based on the VMSINSTAL command procedure of VAX/VMS. Additional information on this procedure can be found in the *Guide to VAX/VMS Software Installation*.

Other documents that may be helpful for users of the DECprint Utility include:

- *Programmer's Supplement: P*OST*S*CRIPT *Printers* (AA–HL84C–TE)
- *VMS System Messages and Recovery Procedures Reference Manual: Part I A–L* (AA–LA17A–TE)
- *VMS System Messages and Recovery Procedures Reference Manual: Part II M–Z* (AA–LA18A–TE)
- *P*OST*S*CRIPT *Language Reference Manual* (AA–HL86A–TE)
- *P*OST*S*CRIPT *Language Tutorial and Cookbook* (AA–HL87A–TE)
- *VMS DCL Dictionary* (AA–LA12A-TE)
- *VMS License Management Utility Manual* (AA–LA33A–TE)

I Installation Guide

This section describes how to install the DECprint Utility for POSTSCRIPT to Sixel Printing for VMS software on a VAX/VMS system, and details preinstallation and postinstallation requirements.

Chapter 1

Before Installation

You can install the DECprint Utility for POSTSCRIPT to Sixel Printing for VMS software on any VAX/VMS computer system. This chapter provides an overview of the DECprint Utility product and lists the preinstallation requirements for the DECprint Utility software.

1.1 Overview

DECprint Utility for POSTSCRIPT to Sixel Printing for VMS software lets you print POSTSCRIPT files on Digital sixel printers. The software uses a host-based Adobe POSTSCRIPT interpreter to translate your POSTSCRIPT input files. The DECprint Utility prints to a printer attached to your terminal (see Figure 1–1) or submits to a queue for printing (see Figure 1–2).

Figure 1–1: Printer Attached to a Terminal

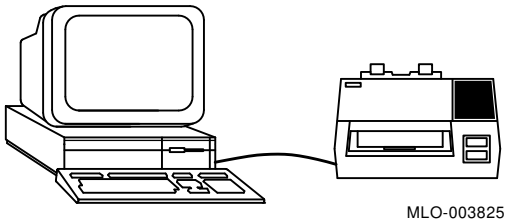
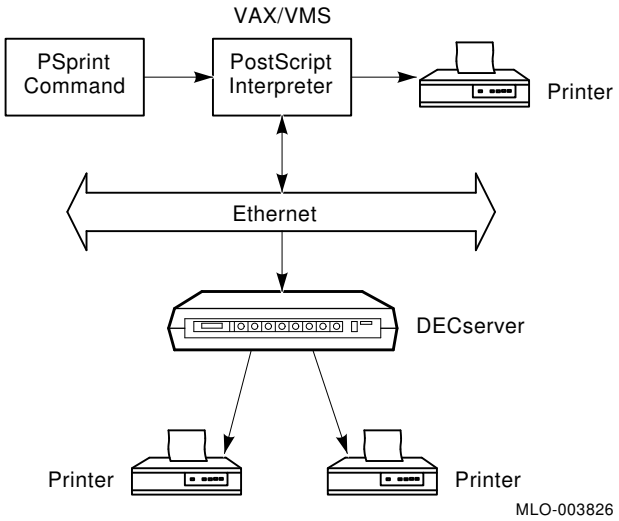


Figure 1–2: Printers on a Queue



The DECprint Utility software provides a single command line that accepts qualifiers similar to the DCL PRINT command. You invoke the software with the DECprint Utility command, PSPRINT:

```
$ PSPRINT/qualifier filename.PS[,...]
```

Most DCL PRINT qualifiers give the same results when used with the DECprint Utility command. For information on the DECprint Utility command see Chapter 4.

1.1.1 PostScript Page Description Language

Digital uses the POSTSCRIPT page description language by Adobe as a standard for its POSTSCRIPT printers, such as the LN03R ScriptPrinter and the PrintServer family of network printers. This language describes the appearance of text, images, and graphics on the printed page. POSTSCRIPT provides high-quality printing that fully integrates text and graphics.

1.1.2 Sixel Graphics

The sixel protocol is compatible with Digital printers that use the ANSI text protocol. The sixel protocol defines and transmits encoded bitmap data over a stream-oriented communications line. Sixels allow devices to receive and print black and white (or color) bitmap data at various sizes to create a graphic image. Six bits (a sixel) of each 7-bit or 8-bit character code represent the bitmap data. Remaining values control the context of the communications line and fit the sixels to the ANSI text syntax.

1.2 Printer Support

DECprint Utility software supports printing on queued printers and printers attached to your terminal. You can use the utility to print to the following sixel printers:

- LA75 Companion Printer
- LA210 Letterprinter
- LA324 MultiPrinter
- LJ250 Companion Color Printer
- LN03 PLUS Desktop Laser Printer

In addition, the DECprint Utility software supports the above printers attached to the following terminals:

- VT100, VT102, and VT131 with printer ports
- VT200 Series
- VT300 Series

1.3 Host Hardware Requirements

The host system must have an input device that corresponds to the DECprint Utility for POSTSCRIPT to Sixel Printing for VMS Software Distribution Kit. The distribution kit is available on the following media:

- Magnetic tape (1600 BPI)
- TK50 COMPACTape cartridge

DECprint Utility software runs on any VAX/VMS computer system and requires no specific serial line communications hardware. The DECprint Utility software uses the VMS job controller to communicate with queued printers and the terminal driver for attached printers.

1.4 Host Software Requirements

The host must be running VMS, Version 5.1 or 5.2. The installation procedure uses the VMSINSTAL utility.

1.5 Software Distribution Kit

Check the contents of the DECprint Utility for POSTSCRIPT to Sixel Printing for VMS Software Distribution Kit against the bill of materials. You should have the following:

- Software labeled DECprint Utility for POSTSCRIPT to Sixel Printing for VMS, Version 1.0, in one of the listed media forms
- *Installation/User's Guide: DECprint™ Utility for POSTSCRIPT® to Sixel Printing for VMS*

1.6 Software License

DECprint Utility for POSTSCRIPT to Sixel Printing for VMS uses the VMS License Management Facility (LMF) software and requires a Product Authorization Key (PAK). The DECprint Utility product name for its PAK is PS-PRINT.

The DECprint Utility software has an *activity license* that allows the execution of one DECprint Utility job at a time. For two users to access the DECprint Utility at the same time, you need two licenses.

See Section 4.6 for a command file that submits a DECprint Utility job for batch processing, checks for the DECprint Utility activity on the system, and enters the DECprint Utility command for processing when the software is available.

1.7 Required Disk Space

The installation of the DECprint Utility software requires 3,400 disk blocks. After installation, this software requires approximately 3,000 disk blocks for operation.

1.8 Installation Time

DECprint Utility installation on the host system can take up to 15 minutes, depending on the type of distribution media, and on the activity on the host system.

1.9 Installation Verification Procedure (IVP) Requirements

To print the file created by the Installation Verification Procedure, requires the following:

- A supported printer on a queue
- A product authorization key (PAK) for PS-PRINT

1.10 Installing the DECprint Utility with the LN03 Image Software

If you are installing the DECprint Utility, Version 1.0, software and the LN03 Image Support Software, Version 1.0, on your system, you must do the following:

1. Install the following kits for the LN03 Image printer:

   VAX LN03 Image Support Software, Version 1.0
   VAX ScriptPrinter Software, Version 2.1

2. Install the DECprint Utility software
3. Execute the following command after completing the installation procedures:

```
$ @SYS$UPDATE:PSPRINT$UPDATE_LN03Q.COM
```

Your screen displays the following message when the command procedure completes:

```
Your VAX LN03 Image Support Software and VAX ScriptPrinter Software
have been successfully updated.  If you reinstall these products,
you must rerun this command file.
```

Reinstalling DECprint Utility software does not require rerunning the command file.

NOTE

Following these steps, which includes executing this command file, is only necessary if you use the LN03 Image printer and the DECprint Utility software on the same system. Do not run the command file to use Vax ScriptPrinter software and the DECprint Utility software on the same system.

Chapter 2

Installing the Software

This chapter describes how to install the DECprint Utility software on a VAX/VMS system, using the VMSINSTAL procedure. If you are not familiar with the VMSINSTAL utility, see the *Guide to VAX/VMS Software Installation*.

2.1 Installation Checklist

Make sure you have the following prerequisites before beginning the installation procedure:

- A backup copy of the software distribution media
- Distribution media on an appropriate input device
- Adequate system disk space (see Section 1.7)
- Access to a privileged account on the host system

2.2 VMSINSTAL Conventions

The VMSINSTAL procedure is interactive. The system prompts and responds with statements, queries, and messages.

Instructions guide you through the procedure or provide you with information relevant to subsequent queries. Statements inform you of the status or progress of the installation procedure.

Queries are preceded by an asterisk (*) and can be followed by default responses enclosed in brackets ([ ]). Enter your response, followed by Return. Just press Return to enter the default response. If you are not sure of the response, enter a question mark (?). The VMSINSTAL procedure displays explanatory text and repeats the prompt.

Messages specific to VMSINSTAL are documented in the *Guide to VAX/VMS Software Installation*. Messages specific to the installation of the DECprint Utility interpreter are found in Chapter 5.

2.3 VMSINSTAL Release Notes

A machine-readable copy of the DECprint Utility Release Notes is included on the software distribution media. If you wish to display or print the release notes, specify OPTIONS N on the VMSINSTAL command line:

```
$  @SYS$UPDATE:VMSINSTAL PSPRINT010 device-id OPTIONS N
```

PSPRINTnnn is the name of the software saveset for the DECprint Utility for POSTSCRIPT to Sixel Printing for VMS. The variable *nnn* represents the version number of the product. For example, *010* represents Version 1.0.

If you did not specify OPTIONS N on the command line, the release notes are automatically copied to SYS$HELP. To print the release notes after installation of the product, enter the following command:

```
$ PRINT/QUEUE=queue-name SYS$HELP:PSPRINT010.RELEASE_NOTES
```

See Section 2.4 for more information.

2.4 Installation Procedure

This section contains excerpts from the installation procedure and provides explanatory text. For a sample installation session, see Section 2.6.

Before invoking VMSINSTAL, log in to the system manager's account and set your default device and directory to SYS$UPDATE.

```
Return
Username:  SYSTEM  Return
Password:          Return
$  SET DEFAULT SYS$UPDATE   Return
```

Invoke the VMSINSTAL procedure as follows:

```
$  @SYS$UPDATE:VMSINSTAL PSPRINT010 device-id OPTIONS N
```

where:

PSPRINT010	is the name of the DECprint Utility for POSTSCRIPT to Sixel Printing for VMS software saveset. The *010* indicates this is Version 1.0.
device-id	is the device on which the distribution media is mounted.
OPTIONS N	is the release notes option, which allows you to display and/or print a copy of the release notes.

The VMSINSTAL utility displays its banner line, the date and time, the statement that you can enter a question mark (?) at any time for help, and one or more warning messages if appropriate. You can continue the installation or exit to modify the conditions causing the warning messages. If the DECnet network is up and running, you always get the following messages:

```
%VMSINSTAL-W-DECNET, Your DECnet network is up and running.
* Do you want to continue anyway [NO]?
```

Answer YES to continue. The DECnet network has no impact on the installation of this product. You are then asked:

```
* Are you satisfied with the backup of your system disk [YES]?
```

Answer YES to continue or NO to exit. If you answered NO, make a backup and begin VMSINSTAL again.

If you did not provide the *device_id* (location of the distribution media) on the VMSINSTAL command line, the VMSINSTAL procedure then asks:

```
* Where will the distribution volumes be mounted:
```

Provide the name of the device on which the distribution media is to be mounted. If the distribution media is mountable, VMSINSTAL then asks for confirmation that the distribution media has been mounted.

```
 Please mount the first volume of the set on device-id:
* Are you ready?
```

Answer YES when ready.

The procedure displays the name and version of the product being installed, the start time of the installation, and confirmation of the restoration of each saveset on the distribution media.

```
The following products will be processed:

  PSPRINT V1.0

Beginning installation of PSPRINT V1.0 at hh:mm

%VMS-I-RESTORE, Restoring product saveset A ...
```

Choosing release notes options

If you specified OPTIONS N in the command, the following release notes options are displayed:

```
   Release Notes Options:
          (1) Display release notes
          (2) Print release notes
          (3) Both 1 and 2
          (4) Copy release notes to SYS$HELP
          (5) Do not display, print or copy release notes
```

If you select 1, the release notes are displayed on your terminal and copied to PSPRINT010.RELEASE_NOTES in SYS$HELP.

If you select 2, the release notes are queued to the printer and copied to PSPRINT010.RELEASE_NOTES in SYS$HELP.

If you select 3, the release notes are displayed on your terminal, queued to the printer, and copied to PSPRINT010.RELEASE_NOTES in SYS$HELP.

If you select 4, the release notes are copied to PSPRINT010.RELEASE_NOTES in SYS$HELP.

If you select 5, the release notes are deleted without being printed, displayed, or copied to SYS$HELP.

If you did not specify OPTIONS N, the release notes are automatically copied to SYS$HELP. When the release notes are copied to SYS$HELP, VSMINSTALL displays this message:

```
%VMSINSTAL-I-RELMOVED, The product's release notes have been
successfully moved to SYS$HELP.
```

VMSINSTAL then displays a copyright message and information on the product you are installing.

```
Installing DECprint Utility for PostScript to Sixel Printing, V1.0
© Digital Equipment Corporation 1989. All Rights Reserved.

     Product:      PS-PRINT
     Producer:     DEC
     Version:      1.0
     Release Date: 15-DEC-1989
```

VMSINSTAL asks if you have registered and loaded the authorization key for the DECprint Utility product.

```
* Does this product have an authorization key registered and
loaded?
```

If you answer no to this question, you cannot run the Installation Verification Procedure (IVP) as part of the installation procedure. The VMSINSTAL procedure displays the following message and suggestions:

```
%PSPRINT-E-NOLICENSE, DECprint Utility execution requires a valid
license.
%PSPRINT-E-NOIVP, The IVP will not be run as part of the
installation.
%PSPRINT-E-NOIVP, It is recommended that the IVP be run after
a valid license is loaded.
```

Running the IVP

If you have registered and loaded the product authorization key (PAK) and answer yes to the last question, the VMSINSTAL procedure asks if you wish to run the installation verification procedure (IVP):

```
* Do you want to run the IVP after the installation [YES]?
```

NOTE

You cannot run the IVP on a printer attached to the printer port of your terminal.

If you answered yes to running an IVP, the procedure asks you for the name of the execution queue where the IVP will print and for the device type of the printer on the queue.

```
* Enter an execution queue name for the IVP:  queue-name_ansi
* Enter the device type for QUEUE-NAME_ANSI:  lj250
```

NOTE

The queue entered for the IVP must be an execution queue, otherwise, you receive an error message stating that the DECprint Utility could not determine the device type.

For valid device types, see Table 3–1. For a sample IVP printout, see Section 2.7. See Section 2.5, for instructions on how to run the IVP after installing the software.

VMSINSTALL asks if you wish to purge files replaced by the installation:

```
* Do you want to purge files replaced by this installation
[YES]?
```

At this point, VMSINSTAL displays a message telling you that no more questions will be asked:

```
PSPRINT-I-NOMOREQUEST, No more questions will be asked

%VMSINSTAL-I-RESTORE, Restoring product saveset B ...
```

The VMSINSTAL procedure now creates a directory for the installation verification procedure (IVP) test files, installs the host-based POSTSCRIPT interpreter, and displays the following messages:

```
%VMSINSTAL-I-SYSDIR,  This product creates system disk
directory VMI$ROOT:[SYSTEST.PSPRINT].

%CREATE-I-EXISTS, VMI$ROOT:[SYSTEST.PSPRINT] already exists

%VMSINSTAL-I-SYSDIR, This product creates system disk
directory  VMI$ROOT:[SYSHLP.EXAMPLES.PSPRINT].

%PSPRINT-PSINSTAL, Installing version V1.1 of the HostPS
interpreter
```

The installation procedure checks to see if a version of the HostPS interpreter already exists on your system. If the version of the HostPS interpreter in the installation kit is the same or later than the version already installed on your system, the procedure installs the HostPS version in the kit. If the version of the HostPS interpreter already installed is a later version than the interpreter in the installation kit, the procedure does not install the HostPS interpreter and displays a message:

```
%PSPRINT-I-PSEXISTS, Version V1.2 of the HostPS interpreter
already exists
```

VMSINSTAL then moves files from their temporary locations used during installation to their permanent directories.

```
%VMSINSTAL-I-MOVEFILES, Files will now be moved to their target
directories...
```

If you answered yes to running the installation verification procedure (IVP), it runs now.

```
Verifying DECprint Utility for PostScript to Sixel Printing, V1.0.

Job PSPRINT$IVP (queue QUEUE-NAME_ANSI, entry 920) started
on QUEUE-NAME
```

VMSINSTAL displays this message to indicate that the IVP was successful:

```
Verification succeeded.
```

VMSINSTAL issues its terminating messages:

```
        Installation of PSPRINT V1.0 completed at hh:mm

        VMSINSTAL procedure done at hh:mm
$
```

DECprint Utility for POSTSCRIPT to Sixel Printing for VMS software is now installed and can be invoked with the PSPRINT command.

The installation procedure modifies the DCL command table so that the PSPRINT command is recognized and processed. However, the old command table is still in effect for those users who are currently logged in. Users who are already logged in and who want to use the PSPRINT command must log out of the system and log in again.

2.5 Running the IVP After Installation

To run the Installation Verification Procedure after installing the software, enter the following command at the prompt:

```
$ @SYS$TEST:PSPRINT$IVP
```

PSPRINT$IVP.COM displays a beginning message, asks you to enter a queue name and device type, submits the IVP to the queue, and displays a message saying that the procedure was successful:

```
Verifying DECprint Utility for PostScript to Sixel Printing, V1.0.

Enter a queue name: my_lj250  Return

Enter the device type for MY_LJ250: lj250  Return

Verification succeeded.
```

2.6 Sample Installation Session

Example 2–1 shows a sample installation session of the DECprint Utility software on a VAX/VMS host:

Example 2–1: Sample Installation Session

```
$ @sys$update:vmsinstal psprint sys$update

        VAX/VMS Software Product Installation Procedure V5.1

It is 23-AUG-1989 at 09:15.
Enter a question mark (?) at any time for help.

%VMSINSTAL-W-DECNET, Your DECnet network is up and running.
%VMSINSTAL-W-ACTIVE, The following processes are still active:
        COYOTE
        EAGLE
        COBRA_1
        GAZELLE
* Do you want to continue anyway [NO]? yes
* Are you satisfied with the backup of your system disk [YES]? yes

The following products will be processed:

  PSPRINT V1.0

        Beginning installation of PSPRINT V1.0 at 09:16

%VMSINSTAL-I-RESTORE, Restoring product saveset A ...
%VMSINSTAL-I-RELMOVED , The product's release notes have been
 successfully moved to SYS$HELP.
```

Example 2–1 Cont'd on next page

Example 2–1 (Cont.): Sample Installation Session

```
Installing DECprint Utility for PostScript to Sixel Printing, V1.0

© Digital Equipment Corporation 1989. All Rights Reserved.

        Product:      PS-PRINT
        Producer:     DEC
        Version:      1.0
        Release Date: 15-DEC-1989

* Does this product have an authorization key registered and loaded?
yes
* Do you want to run the IVP after the installation [YES]? yes
* Enter an execution queue name for the IVP: lj250_b
* Enter the device type for LJ250_B: LJ250
* Do you want to purge files replaced by this installation [YES]?
yes

%PSPRINT-I-NOMOREQUEST, No more questions will be asked
%VMSINSTAL-I-RESTORE, Restoring product saveset B ...
%VMSINSTAL-I-SYSDIR, This product creates system disk directory
VMI$ROOT:[SYSTEST.PSPRINT].
%VMSINSTAL-I-SYSDIR, This product creates system disk directory
VMI$ROOT:[SYSHLP.EXAMPLES.PSPRINT].
%PSPRINT-I-PSINSTAL, Installing version V1.1 of the HostPS
 interpreter
%VMSINSTAL-I-MOVEFILES, Files will now be moved to their target
 directories...

Verifying DECprint Utility for PostScript to Sixel Printing, V1.0.

Job PSPRINT$IVP (queue LJ250_B, entry 920) started on LJ250

Verification succeded.

        Installation of PSPRINT V1.0 completed at 09:26

        VMSINSTAL procedure done at 09:26

$
```

2.7 IVP Sample Output

Figure 2–1 shows the black-and-white IVP output of the sample POSTSCRIPT program that the DECprint Utility software submitted to a print queue for printing on a sixel printer.

If you enter the execution queue name for a printer with color for your IVP, the color output is as follows:

- **DECprint Utility** prints with red letters
- **Successfully** prints in green
- **Installed** prints in blue

Figure 2–1: DECprint Utility IVP Output

Chapter 3

Postinstallation Requirements

This chapter explains the operations to be performed after the installation of the DECprint Utility software. These operations should be performed by a system or cluster manager.

3.1 Creating Logical Names for Queued Printing

DECprint Utility software needs to know the device type of the printer. For queued printers, which you indicate in the DECprint Utility command with the /QUEUE qualifier, the software gets the device type from a logical name. Logical names must be defined by the system manager. Define a logical name for each execution queue to which system users will direct DECprint Utility jobs.

To define device type logical names, add the following command to the system startup file SYS$MANAGER:SYSTARTUP_V5.COM for each DECprint Utility execution queue:

```
$ DEFINE/SYSTEM/EXEC/NOLOG PSPRINT$queue_DEVICE_TYPE -
  device_type
```

where:

queue	is the name of the execution queue for the printer.
device-type	is the device type of the printer (see Table 3–1 for supported device types).

DECprint Utility software resolves queue logical names and looks for the device type of the execution queue. If the device type is not defined you receive the following error message:

```
PSPRINT-E-NODEVTYPE, could not determine device type for
queue-name
```

Make sure that generic queue names used with the /QUEUE qualifier for the DECprint Utility command point to execution queues that serve only one type of printer; otherwise you receive this error message:

```
PSPRINT-E-NOMIXDEV, all execution queues must have the same
device type
```

Example:

Example 3–1 shows sample commands to add to the system startup file to define the device type logical names for DECprint Utility printing. For example, you have the following printers that you plan to use to print the output from the DECprint Utility command:

Printer	Execution Queue Name	Device Type Keyword
LJ250 Companion Color Printer	LJ250_COLOR	LJ250
LJ250 Companion Color Printer	NEDS_LJ250	LJ250
LN03 PLUS Desktop Laser Printer	LN03_A	LN03S
LN03 PLUS Desktop Laser Printer	LN03_B	LN03S

Example 3–1: Defining Device Type Logical Names

```
$ DEFINE/SYSTEM/EXEC/NOLOG PSPRINT$lj250_color_DEVICE_TYPE lj250
$ DEFINE/SYSTEM/EXEC/NOLOG PSPRINT$neds_lj250_DEVICE_TYPE lj250
$ DEFINE/SYSTEM/EXEC/NOLOG PSPRINT$ln03_a_DEVICE_TYPE ln03s
$ DEFINE/SYSTEM/EXEC/NOLOG PSPRINT$ln03_b_DEVICE_TYPE ln03s
```

Table 3–1 lists the printers supported by the DECprint Utility software that includes valid values for the *device-type* keyword:

Table 3–1: Supported Devices

Device Name	Device Type Keyword	Resolution (dpi) horiz × vert
LA75 Companion Printer	LA75	144 × 144
LA210 Letterprinter	LA210	132 × 72
LA324 MultiPrinter	LA324	180 × 180
LA324 MultiPrinter (color ribbon)	LA324COLOR	180 × 180
LJ250 Companion Color Printer	LJ250	180 × 180
LN03 PLUS Desktop Laser Printer	LN03S	300 × 300

NOTE

PSPRINT$*queue*_DEVICE_TYPE logicals must be created and maintained by the system manager. They are not created during the installation of the software. Future products may not support this mechanism.

3.2 Enabling Other Nodes on VAXcluster Systems

The installation procedure modifies the DCL command table on the node where you installed the DECprint Utility software. To enable the DECprint Utility command on other nodes of the VAXcluster Systems environment, you must install the new DCL command table on the other nodes. Complete this procedure to install the DCL command table on the other nodes in the cluster:

```
$ INSTALL :== $SYS$SYSTEM:INSTALL/COMMAND_MODE Return
$ INSTALL  Return
INSTALL> REPLACE SYS$LIBRARY:DCLTABLES Return
INSTALL> EXIT Return
$
```

The old DCL command table is in effect for those users who are logged in. Logged-in users who want to use the DECprint Utility command must log out of the system and log in again.

3.3 Host and Printer Characteristics

Make sure the following characteristics are the same on the printer and the host computer:

- Baud rate
- Data bits
- Parity
- Protocol

II User's Guide

This part includes sections that describe how to use the DECprint Utility command to print your POSTSCRIPT files on ANSI/sixel printers, explain system messages and recovery procedures, and describe the printer environment.

Chapter 4

Using the DECprint Utility Command

You invoke the DECprint Utility for POSTSCRIPT to Sixel Printing for VMS software with the single command, PSPRINT. Using the DECprint Utility command is similar to using the DCL PRINT command. The DECprint Utility command accepts qualifiers specific to DECprint Utility as well as many DCL PRINT qualifiers. In addition, the DECprint Utility software processes some PRINT qualifiers differently and does not support others.

This chapter discusses the following topics concerning the use of the DECprint Utility command:

- The command line syntax
- Using the command line for queued printing
- Using the command line with printers attached to a terminal
- Defining the printer type for queued and attached device printing
- PRINT qualifiers supported by the DECprint Utility software
- Command qualifiers specific to the DECprint Utility software
- PRINT qualifiers that work differently
- Unsupported DCL PRINT qualifiers
- Submitting DECprint Utility commands for batch processing
- Color printing with the DECprint Utility software
- Miscellaneous helpful hints to use with the DECprint Utility software

4.1 The Command Line

To translate your POSTSCRIPT files for printing on a sixel printer, enter the DECprint Utility command as follows:

```
$ PSPRINT/qualifier filename.PS[,...]
```

You can print the converted sixels on either a printer attached to your terminal or a printer on a queue, by selecting the /ATTACHED or /QUEUE qualifier with the DECprint Utility command line. Sections 4.3 and 4.1.1 discuss other qualifiers supported by the DECprint Utility software.

To print to an LJ250 Companion Color Printer on a queue, use the /QUEUE qualifier with the DECprint Utility command:

```
$ PSPRINT/QUEUE=LJ250_queue-name fileA.ps
```

For more information on using the DECprint Utility command with printers on queues, see Section 4.1.1.

To print to an LJ250 Companion Color Printer attached to your terminal, use the /ATTACHED qualifier with the DECprint Utility command:

```
$ PSPRINT/ATTACHED=LJ250 fileA.ps
```

For more information on using the DECprint Utility command with printers attached to your terminal, see Section 4.1.2.

The default file type for input files to the DECprint Utility command is .PS not .LIS as for the DCL PRINT command.

The software resets the POSTSCRIPT interpreter after each print job.

4.1.1 The DECprint Utility Command and Queued Printers

DECprint Utility software submits print jobs for printing on queued printers when you include the /QUEUE qualifer on the command line. Without the /QUEUE or /ATTACHED qualifier, the default is /QUEUE=SYS$PRINT.

NOTE

If you include both /QUEUE and /ATTACHED on the command line, the DCL software generates an error message.

Section 4.3 discusses qualifiers that are specific to the DECprint Utility software. DECprint Utility software also supports several DCL PRINT qualifiers for queued printers by passing them through unaffected to the VMS print job controller. The following qualifiers take effect when the DECprint Utility software queues the sixel file for printing:

/AFTER	/HOLD	/OPERATOR
/BURST	/IDENTIFY	/PRIORITY
/CHARACTERISTICS	/JOB_COUNT	/RESTART
/COPIES	/NAME	/TRAILER
/FLAG	/NOTE	/USER
	/NOTIFY	

In addition to passing qualifiers from the command line that take effect at queue time, the DECprint Utility software always submits the sixel file for printing with the following qualifiers:

- /PASSALL and /NOFEED — ensures that the symbiont does not change the sixel file format.
- /NAME — provides a recognizable filename. You can specify this name on the command line by using the /NAME qualifier. Otherwise, the DECprint Utility software uses the filename of the first POSTSCRIPT file in the print job. This behavior mimics that of DCL PRINT.
- /DELETE — deletes the sixel file after the file has been printed.

For information on the DCL PRINT command qualifiers, refer to the VMS DCL Dictionary.

4.1.2 The DECprint Utility Command and Attached Printers

Attached printers are printers that connect directly to the printer port on your video terminal through an interface cable. If you have not already attached your printer to the printer port of your terminal, Digital recommends that you use a queue system for printing. To create a queue for your printer, refer to Appendix A.

With the /ATTACHED qualifier on the command line, the DECprint Utility software sets the terminal to *printer controller mode* and then sends the sixels to the terminal. Printer controller mode causes the sixels to be printed on the printer attached to the terminal printer port rather than displayed on the terminal screen. No output goes to

the terminal while the terminal is in printer controller mode. After printing of the sixels is complete, the DECprint Utility software resets output to the screen.

To ensure that no other output goes to the screen during printing of the sixels, the DECprint Utility software temporarily sets the /NOBROADCAST terminal characteristic. In addition, if necessary, the software sets other terminal characteristics during the printing process and resets them to their original state on successful completion of the printing.

NOTE

Because /NOBROADCAST is set, the terminal does not print broadcast messages when you use PSPRINT/ATTACHED. These messages are lost.

While the terminal is in printer controller mode, the terminal sends characters typed at the keyboard to the host. For most characters, the host stores the data in a type-ahead buffer. DCL then interprets the data when the DECprint Utility command activity terminates. This is not true for the control characters listed in Table 4–1.

Table 4–1: Attached Printers and Control Characters

Control Character	Action
Ctrl/O	Stops and starts the flow of sixels to the printer. No announcement displays on the terminal screen. (VMS normally displays ''Output On'' and ''Output Off'' messages in reverse video.)
Ctrl/Y	Terminates the DECprint Utility command and returns the DCL prompt. (DECprint Utility software disables normal DCL handling of Ctrl/Y.)
Ctrl/C	Terminates the DECprint Utility command and returns the DCL prompt.
Ctrl/S and Ctrl/Q	Effect flow control. Do not use these control characters as the printer or the terminal may be handling its own flow control.

Section 4.3 discusses qualifiers that the DECprint Utility command supports for printing to a printer attached to the terminal's printer port. DCL PRINT qualifiers supported for attached printers are listed in Section 4.9, Table 4–5.

4.2 Device Support

To print with DECprint Utility software requires defining the type of printer. The method depends on whether the software is queuing the print job or printing to an attached printer.

Queued printing

For queued jobs, the DECprint Utility software translates the systemwide logical PSPRINT$*queue*_DEVICE_TYPE. The value of this logical must be a device type keyword from Table 3–1. You can send DECprint Utility jobs to the execution queue for the supported printer or to generic queues or to a queue with a logical name pointing to a generic queue. DECprint Utility software translates any valid queue name into execution queue names before reading the device type logical name.

However, you cannot send DECprint Utility jobs to generic queues that point to execution queues for more than one type of printer; otherwise, the DECprint Utility ends the program and generates an error message. If an execution queue is not defined, the DECprint Utility also ends the program and generates an error message.

NOTE

The PSPRINT$*queue*_DEVICE_TYPE logical must be created and maintained by the system manager. Future products may not support this logical.

For more information on defining the device type logical, see Section 3.1.

Attached Device Printing

When printing to an attached printer you include the device type keyword as the value of the /ATTACHED qualifier on the DECprint Utility command line:

```
$ PSPRINT/ATTACHED=device-type filename.ps
```

If you use the /ATTACHED qualifier, you must include a *device-type*. See Table 3–1 for the accepted values for the device type keyword. For more information on the /ATTACHED qualifier, see Section 4.3.1.

4.3 DECprint Utility Command Qualifiers

This section describes the command qualifiers specific to DECprint Utility:

- /ATTACHED=*device-type*
- /[NO]LAYUP
- /MEDIUM=*keyword*
- /[NO]MESSAGES
- /ORIENTATION=*logical-orientation*

4.3.1 /ATTACHED Qualifier

Format: /ATTACHED=*device-type*

/ATTACHED indicates that the printer attached to the terminal's printer port receives the converted sixels for printing. DECprint Utility software automatically directs the sixels to the printer rather than to the screen. When the printer finishes printing the sixels, the software resets the output to the screen. You cannot send output to the terminal during the printing process.

You must associate a device type keyword with the /ATTACHED qualifier to indicate the type of printer attached to your terminal. Table 3–1 lists the valid device type keywords.

Only certain DCL PRINT qualifiers are valid when using the DECprint Utility command to print to the printer attached to your terminal. For a list of qualifiers you can use with /ATTACHED, refer to Section 4.9, Table 4–5.

Default: /NOATTACHED

Example:

Example 4–1 shows a command that converts the files in fileA*.ps from POSTSCRIPT to sixels, and sends the resulting sixels to an LJ250 Companion Color printer attached to your terminal.

Example 4–1: Example with /ATTACHED

```
$ PSPRINT/ATTACHED=LJ250 fileA*.ps
```

4.3.2 /LAYUP Qualifier

Format: /LAYUP=*integer*

/LAYUP is a qualifier that specifies the number of logical pages to be printed per physical sheet of paper. The value *integer* is a number from 1 to 100. If you use this qualifier on the command line, you must supply a number.

You should not apply /LAYUP to one file and not to other files on the DECprint Utility command line. Therefore, position the /LAYUP qualifier after the DECprint Utility command verb and before any file names.

/LAYUP controls the use of the layup feature in POSTSCRIPT to sixel interpretation. It is equivalent to the Number_Up parameter in the existing POSTSCRIPT printers with the following exception:

- Number_Up allows a value of zero; DECprint Utility does not. /PARAMETERS=(Number_Up=0) is the same as PSPRINT/NOLAYUP.

Default: /NOLAYUP

Example:

The command in Example 4–2 sends fileX.ps and fileZ.ps to a queued printer, *queue-name*, and uses the /LAYUP feature to print both files four pages to a sheet on A-size paper (8.5 × 11 in.) in landscape orientation.

Example 4–2: Examples with /LAYUP

```
$ PSPRINT/QUEUE=queue-name/LAYUP=4/ORIENTATION=landscape/MEDIUM=A -
  fileX.ps,fileZ.ps
```

4.3.3 /MEDIUM Qualifier

Format: /MEDIUM=*keyword*

/MEDIUM specifies the medium used in the printer. For *keyword* values, see Table 4–2. DECprint Utility software uses this value to determine the size of medium required for the sixel image. If you provide the /MEDIUM qualifier on the command line, you must supply a *keyword* value.

Table 4–2: Media Keywords

Keyword	Medium
LETTER or A	8.5 × 11 in. (216 × 279 mm)
A4	210 × 297 mm (8.3 × 11.7 in.)

You cannot mix media sizes when you use the DECprint Utility command as you cannot select media on the supported printers. Therefore, you must position the /MEDIUM qualifier after the DECprint Utility command verb and before any file names. Using mixed media generates an error message.

Default: /MEDIUM=A

Example:

Example 4–3 translates fileB.ps and fileC.ps for A4-size paper and queues the resulting sixels for printing. The position of the /MEDIUM qualifier in this example guarentees that the qualifier applies to both files.

Example 4–3: Example with /MEDIUM

```
$ PSPRINT/QUEUE=queue_name/MEDIUM=A4 fileB.ps,fileC.ps
```

4.3.4 /MESSAGES Qualifier

Format: /MESSAGES=*keyword*

/MESSAGES specifies the destination for messages and data returned by the POSTSCRIPT interpreter. If you use this qualifier you must supply one of the following values for *keyword*:

- TERMINAL
- FILE

/MESSAGES=TERMINAL sends the messages and data from the POSTSCRIPT interpreter to the user's terminal. When you use this qualifier with /ATTACHED, the DECprint Utility software manages the terminal so that the messages and data appear on the terminal and not on the printed page.

/MESSAGES=FILE=*file-name* takes an additional option, the name of the file to create. If you do not provide the file name, the default for /MESSAGES=*file-name* is SYS$LOGIN:*name*.LOG, where *name* is the name of the print job provided in either of the following:

- /NAME qualifier on the DECprint Utility command
- Filename of the first POSTSCRIPT file translated

Do not specify /MESSAGES=FILE=TT:; instead use /MESSAGES=TERMINAL.

/NOMESSAGES specifies that the DECprint Utility create no log file and discard any messages from the interpreter.

Default: /NOMESSAGES

Examples:

The first example in Example 4–4 shows a command that translates fileX.ps, queues the resulting sixel file to *queue-name*, and enters any data or messages from the POSTSCRIPT interpreter in a log file named *fileX.log* in your login directory.

The second example in Example 4–4 shows a command that translates fileZ.ps, sends the resulting sixels to an LN03 PLUS attached to the terminal, and creates the log file named *messages.log* in you login directory.

Example 4–4: Examples with /MESSAGES

```
$ PSPRINT/QUEUE=queue-name/MESSAGES=FILE fileX.ps
$ PSPRINT/ATTACHED=LN03/MESSAGES=FILE=messages fileZ.ps
```

4.3.5 /ORIENTATION Qualifier

Format: /ORIENTATION=*logical-orientation*

/ORIENTATION is a *positional qualifier* that declares the orientation of the page images in the document. DECprint Utility software only uses /ORIENTATION when you invoke LAYUP; otherwise, the utility ignores the qualifier. The /ORIENTATION qualifier should match the orientaion of pages in the input POSTSCRIPT file. If you provide this qualifier on the command line, *logical-orientation* must be one of the following values:

- PORTRAIT
- LANDSCAPE

Default: /ORIENTATION=PORTRAIT

Example:

For an example using the /ORIENTATION qualifier, see Example 4–2.

4.3.5.1 Page Orientation and /LAYUP

When you print using /LAYUP, the aspect ratio of the page is preserved. This means, for example, a portrait-shaped page retains its portrait shape. However, because LAYUP scales pages to make the best use of the available space, the page and sheet may have different orientations.

Figures 4–1 and 4–2 show the four different cases of page and sheet orientation with /LAYUP. Text in the first page spot is represented by the parallel lines. The POSTSCRIPT origin of the sheet and first page are represented by the quarter circles.

Figure 4–1 shows how your portrait file prints with /LAYUP=2 and /LAYUP=4. With /LAYUP=2, the pages remain portrait-shaped, but you hold the sheet in landscape orientation to read them.

Figure 4–2 shows how your landscape file prints with /LAYUP=2 and /LAYUP=4. With /LAYUP=2/ORIENTATION=LANDSCAPE, the pages have a landscape shape, but you hold the sheet in portrait orientation to read them.

Figure 4–1: Printing Your Portrait File

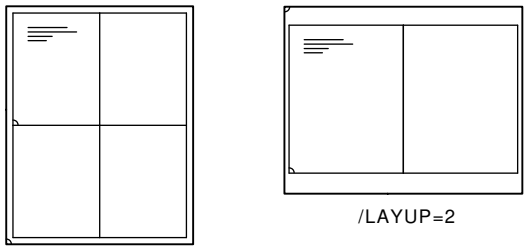
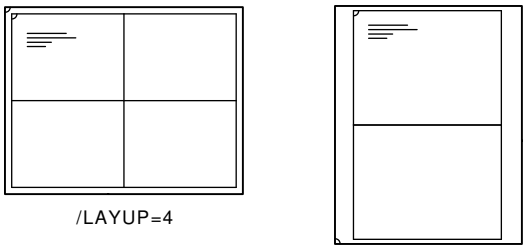


Figure 4–2: Printing Your Landscape File



4.4 PRINT Qualifiers That Work Differently

This section describes PRINT command qualifiers that do not work the same with the DECprint Utility command as they do with the DCL PRINT command.

4.4.1 /AFTER Qualifier

In DCL PRINT, /AFTER specifies when the printing of the file begins. The DECprint Utility software passes /AFTER to $SNDJBC. Therefore, /AFTER specifies when the printing of the sixels begins. Conversion from POSTSCRIPT to sixels by the DECprint Utility software begins immediately.

See Section 4.6 for information on how to submit a command file to run DECprint Utility jobs in batch or at a later time.

4.4.2 /PARAMETERS Qualifier

/PARAMETERS extends the PRINT line command for existing POSTSCRIPT printers. DECprint Utility software uses new command qualifiers to provide the features of the /PARAMETERS qualifier (see Section 4.3). DECprint Utility software does not parse or interpret values specified by /PARAMETERS, but passes them to the VMS job controller ($SNDJBC), in case your symbiont reads the /PARAMETERS qualifier.

4.5 Unsupported PRINT Qualifiers

Several DCL PRINT command qualifiers do not work with the DECprint Utility command. If you supply these qualifiers on the DECprint Utility command line, you receive an error indicating that the qualifier is not valid when using the DECprint Utility command. This section describes and lists these qualifiers.

4.5.1 Formating Qualifiers

As the DECprint Utility software only converts and prints POSTSCRIPT files, it does not support the following ASCII text formating print qualifiers:

/FEED	/PASSALL
/HEADER	/SPACE

4.5.2 File Selection Qualifiers

DECprint Utility software does not support the following file selection print qualifiers:

/BACKUP	/CONFIRM	/EXPIRED
/BEFORE	/CREATED	/MODIFIED
/BY_OWNER	/EXCLUDE	/SINCE

4.5.3 /FORM and /SETUP Qualifiers

The DECprint Utility does not support the following qualifiers that are associated with forms and setup modules.

/FORM Qualifier

DECprint Utility software ignores much of the form functionality as it submits jobs for printing with the /PASSALL qualifier. In addition, DECprint Utility does not support setup modules. For these reasons, /FORM is not supported.

/SETUP Qualifier

DECprint Utility software does not support /SETUP for the same reasons that it does not support /FORM. DECprint Utility submits jobs for printing with the /PASSALL qualifier that ignores much of the form functionality and does not support setup modules.

If your POSTSCRIPT file requires setup modules to work properly, you can do either of the following:

- Rewrite your files to be independent of setup modules, which increases their portability.
- Extract the required modules into a file and include the name of the file on the command line.

```
$ PSPRINT/QUEUE=queue-name setup_filename.PS,myfile.PS
```

4.5.4 Miscellaneous Qualifiers

Miscellaneous print qualifiers that DECprint Utility software does not support include:

/DELETE	/LOWERCASE
/DEVICE	/REMOTE

/DELETE Qualifier

When using the DCL PRINT command, /DELETE refers to the file that was named in the PRINT command line. Since the DECprint Utility software does not know when printing completes, it cannot delete the original POSTSCRIPT file. You can delete the original POSTSCRIPT file.

/DEVICE Qualifier

DCL allows /DEVICE as a synonym for /QUEUE. The DECprint Utility does not support /DEVICE; use /QUEUE instead.

4.6 Submitting DECprint Utility Commands for Batch Processing

This section contains an example of a command file and the DCL command that you can use if you want to submit a DECprint Utility command for processing in batch or at a later time. The command file, Batch-PSPRINT.com, is supplied during the installation process in the directory SYS$COMMON:[SYSHLP.EXAMPLES.PSPRINT]. To use the command file to submit your DECprint Utility job to a batch queue, enter the following DCL command:

```
$ SUBMIT /PARAMETERS=(parameter[,...]) Batch-PSPRINT
```

You can include up to eight parameters with the SUBMIT command. These parameter values equate to the symbols named **p1** through **p8** in Batch-PSPRINT.com (see Example 4–6). Specify all DECprint Utility qualifiers and parameters, including the the POSTSCRIPT file name, in one of these parameters. If a parameter contains spaces, special characters, or lowercase letters, enclose the parameter in quotation marks. A parameter can have from 1 to 255 characters.

NOTE

The batch process receives your disk and directory defaults and the same resource quotas and privileges that were given to your interactive process when you logged in.

Examples:

The first command in Example 4–5 shows the DECprint Utility job submitted to the default batch queue SYS$BATCH for immediate processing. In batch, the DECprint Utility software converts fileA.ps and prints to a queue named *lj250b*. */queue=lj250b/notify fileA.ps* can be included in a single parameter, as in this example, or in separate parameters, as in the other two examples.

The next command in Example 4–5 shows a command that sends the DECprint Utility job to a selected batch queue to be processed after 6:30 pm today. In this case, /AFTER is the DCL SUBMIT/AFTER qualifier not the DECprint Utility /AFTER qualifier.

The third command in Example 4–5 shows a DECprint Utility job involving LAYUP that is submitted to a selected batch queue for processing immediately.

Example 4–5: Examples Using the Batch-PSPRINT.com File

```
$ SUBMIT /PARAMETERS=("/queue=lj250b/notify fileA.ps") Batch-PSPRINT

$ SUBMIT /PARAMETERS=("/queue=lj250b","/notify","fileB.ps")-
  /AFTER=18:30 /QUEUE=psprint_batch Batch-PSPRINT

$ SUBMIT /PARAMETERS=("/queue=ln03s_A","/layup=4",-
  "/orientation=landscape","/medium=A","/messages=file=fileX",-
  "fileX.ps") /QUEUE=psprint_batch /NOTIFY Batch-PSPRINT
```

Example 4–6 shows the command procedure Batch-PSPRINT.com that you can use to submit your DECprint Utility job to a batch queue for processing. This procedure checks for the following:

- A valid software license
- DECprint Utility activity (units) on the system

If another DECprint Utility job is active on the system, the command procedure waits one minute and submits your DECprint Utility job again. Batch-PSPRINT.com continues to check and process your job when the software is available.

Example 4–6: The Batch-PSPRINT Command File

```
$!
$! Batch-PSPRINT.com
$!
$!
$
$ outfile := sys$scratch:batch-psprint-'f$getjpi("","pid").log
$
$ try_psprint:
$
$ on error then continue
$ define/user sys$output 'outfile
$ psprint 'p1 'p2 'p3 'p4 'p5 'p6 'p7 'p8
$
$ !
$ ! Check the result status.
$ ! If it isn't LICENSE-F-NOAUTH, then we exit with it.
$ !
$
$ ppstatus = $status
$ if ppstatus .ne. %x1A948AF4 then goto clean_up
$
$ !
$ ! If we did exit because PSPRINT wasn't authorized, then we check to
$ ! see if it was because there weren't enough units.
$ !
$
$ open/read out 'outfile
$ read out line
$ read out line
$ close out
$ if f$extract(0, 19, line) .nes. "-LICENSE-F-EXCEEDED" then goto clean_up
$
$ !
$ ! If we exited because there weren't enough units, then we will wait for
$ ! one minute and try again. By then, the license may have been released.
$
$ wait 00:01
$ goto try_psprint
```

Example 4–6 Cont'd on next page

Example 4–6 (Cont.): The Batch-PSPRINT Command File

```
$
$ !
$ ! To clean up we just have to delete the file we used to capture
$ ! the messages.
$ !
$
$ clean_up:
$ delete 'outfile';*
$ exit ppstatus
```

4.7 Color Printing with DECprint Utility

DECprint Utility software converts POSTSCRIPT files with color information for printing on the LJ250 Companion Color Printer and the LA324 MultiPrinter.

DECprint Utility software converts colors to shades of gray for printing on sixel printers that do not have color capability.

4.8 Miscellaneous Helpful Hints

This following sections contain helpful hints for use with the DECprint Utility.

4.8.1 Using Transparencies with the LJ250 Companion Color Printer

To print transparencies with the LJ250 printer, you must first put the printer in transparency mode, as follows:

1. Turn off power to the printer
2. Press and hold the DEC/PCL switch on the front panel while pressing the power switch.

The printer is now in transparency mode and you can submit your POSTSCRIPT file to the DECprint Utility software for printing on the LJ250 printer.

4.8.2 Printing PostScript Documents From MAIL

You can use the DECprint Utility to print POSTSCRIPT files received from electronic MAIL on a sixel printer. First, copy the POSTSCRIPT file to a file in your directory using the EXTRACT/NOHEAD command. Then submit the file for processing with the DECprint Utility command.

```
MAIL> EXTRACT/NOHEAD mailfile.PS
MAIL> EXIT

$ PSPRINT/QUEUE=queue-name mailfile.PS
```

4.8.3 Running the DECprint Utility in the Background

You can free your terminal for other applications, by doing either of the following:

- Run the DECprint Utility in a second session on your terminal
- Submit the DECprint Utility job to a batch queue

Before running the DECprint Utility in a second session, make sure your terminal is set up properly. Dual sessions on the VT330/VT340 video terminals require two communications cables or one communications cable and the Session Support Utility (SSU) software. The terminal server also provides dual sessions for your terminal. Refer to the user documentation for your video terminal for information on setting up dual sessions.

Section 4.6 provides information on how to submit your DECprint Utility job to a batch queue for processing.

4.8.4 Printer Margin Information

Table 4–3 lists the top, bottom, left, and right margins, in inches, for the printers supported by the DECprint Utility software. These margins cannot be changed.

Table 4–3: Printer Margins in Inches

Printer Name	Top	Bottom	Left	Right
LA75 Companion Printer	.125	.125	.17	.32
LA210 Letterprinter	.125	.125	.125	.125
LA324 MultiPrinter	.04	.04	.16	.16
LJ250 Companion Color Printer	.04	.04	.26	.16
LN03 Plus Desktop Laser Printer (A-Size)	.25	.29	.25	.25
LN03 Plus Desktop Laser Printer (A4-Size)	.45	.79	.25	.25

4.9 Summary of DECprint Utility Qualifiers

This section lists the qualifiers supported by the DECprint Utility and provides the following information about them:

- Type of value associated with the qualifier: integer, keyword, string
- Default value of the qualifier
- If the qualifier can be negated or not, for example /NOLAYUP
- Whether the qualifier can be applied to files in a job (positional) or only to the job (global)
- If the qualifier is new with the DECprint Utility command or is a DCL PRINT command qualifier

Table 4–4 lists qualifiers that the DECprint Utility command supports for queued printing.

Table 4–4: DECprint Utility Qualifiers

Qualifier	Value	Default	Negatable	Global(G)/ Positional(P)	Source
AFTER	date	–	Yes	G	PRINT
ATTACHED	device type	–	Yes	G	NEW
BURST	–	–	Yes	P	PRINT
CHARACTERISTICS	list	–	Yes	G	PRINT
COPIES	integer	1	Yes	P	PRINT
FLAG	keyword[1]	ALL	Yes	P	PRINT
HOLD	–	No	Yes	G	PRINT
IDENTIFY	–	Yes	Yes	G	PRINT
JOB_COUNT	integer	1	Yes	G	PRINT
LAYUP	integer	NOLAYUP	Yes	P	NEW
MEDIUM	keyword[2]	LETTER	No	P	NEW
MESSAGES	keyword[2]	NOMESSAGES	Yes	G	NEW
NAME	string	–	Yes	G	PRINT
NOTE	string	–	Yes	G	PRINT
NOTIFY	–	NONOTIFY	Yes	G	PRINT
OPERATOR	string	–	Yes	G	PRINT
ORIENTATION	keyword[2]	PORTRAIT	No	P	NEW
PAGES	integers	–	Yes	P	PRINT
PARAMETERS	list	–	Yes	G	PRINT
PRIORITY	integer	–	Yes	G	PRINT
QUEUE	queue name	SYS$PRINT	Yes	G	PRINT
RESTART	–	–	Yes	G	PRINT
TRAILER	keyword[2]	–	Yes	P	PRINT
USER	username	–	Yes	G	PRINT

[1]See the *VMS DCL Dictionary* for supported keywords for this qualifier.

[2]See Section 4.3 for supported keywords for this qualifier.

Table 4–5 lists those qualifiers that the DECprint Utility command supports for printers attached to your terminal (/ATTACHED).

Table 4–5: Qualifiers Supported for Attached Printers

Qualifier	Value	Default	Negatable	Global(G)/ Positional(P)	Source
LAYUP	integer	–	Yes	P	NEW
MEDIUM	keyword[2]	–	No	P	NEW
MESSAGES	keyword[2]	–	Yes	G	NEW
NAME	string	–	Yes	G	PRINT
ORIENTATION	keyword[2]	–	No	P	NEW
PAGES	numlist	–	Yes	P	PRINT

[2]See Section 4.3 for supported keywords for this qualifier.

Chapter 5

System Messages

This chapter contains the system messages issued by the DECprint Utility software and the POSTSCRIPT interpreter. Refer to the *Guide to VAX/VMS Software Installation* for explanations of messages issued by VMSINSTAL during product installation. Refer to the *VMS System Messages and Recovery Procedures* for explanations of messages issued by the DCL software. Refer to *VMS License Management Utility Manual* for an explanation of the the Licence Management Facility (LMF) messages.

5.1 Message Format

System messages issued by the DECprint Utility software have the following format:

```
%PSPRINT-s-ident, text
-PSPRINT-s-ident, text
```

where:

%	is the prefix to all primary messages.
–	is the prefix to all continuation messages.
s	is the severity level of the message.
ident	is an abbreviation of the message text.
text	is the expanded text of the message.

5.2 Severity Level

The severity levels of DECprint Utility system messages are as follows:

Code	Meaning
S	Success—successful completion of the request.
I	Informational—may or may not require user action.
W	Warning—request may not have completed and may require user action.
E	Error—system encountered an error that may be recoverable.
F	Fatal—system encountered a fatal error and cannot continue processing the request.

5.3 Message Descriptions

The following message descriptions are alphabetized by the *ident* portion of the messages code. The message prefix, source designation, and severity code are not shown.

BADPS, couldn't use PostScript interpreter

Explanation: DECprint Utility software cannot access the host-based POSTSCRIPT interpreter. Another message follows indicating the problem.

User Action: Correct the problem indicated by the accompanying message.

CREJOB, error creating job

Explanation: The print job could not be submitted to the queue. Another message accompanies this message to explain the problem.

User Action: Correct the problem indicated by the accompanying message.

DICTFULL, dictfull: No more room in dictionary — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

DICTSTKOV, dictstackoverflow: Too many begins

Explanation: The POSTSCRIPT interpreter sensed too many *begin*s without corresponding *end*s.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

DICTSTKUF, dictstackunderflow: Too many ends

Explanation: The POSTSCRIPT interpreter sensed too many *end*s without corresponding *begin*s.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

EXECSTKOV, execstackoverflow: Exec nesting is too deep — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

FLUSHING, rest of job will be ignored

Explanation: An error or some other event has occurred that is causing POSTSCRIPT to ignore the rest of the job. One or more messages preceding this one should provide information on why the job is being flushed.

User Action: Check the other messages that accompanied this one.

INTERNERR, internal error *code*; submit an SPR

Explanation: A fatal error has occurred. The *code* pinpoints the location of the problem.

User Action: Submit a Software Performance Report (SPR) that includes details of the situation under which the message occurred.

INCQUETYUP, inconsistent queue type

Explanation: The queue specified with /QUEUE is not suitable for printing. (Specifying a batch queue will give this message.)

User Action: Provide a valid name for the /QUEUE qualifier and re-enter your DECprint Utility command.

INTERUPT, interrupt: The job has been interrupted

Explanation: The POSTSCRIPT interpreter sensed an external request to interrupt the POSTSCRIPT program. This message also appears as part of the Abort process.

User Action: None. This message is a confirmation of a requested action.

INVACC, invalidaccess: Attempt to access restricted object or capability — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

INVEXIT, invalidexit: Exit not in loop — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

INVFILACC, invalidfileaccess: Bad file access string — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

INVFONT, invalidfont: Bad font name or dictionary — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

INVLOGFIL, invalid log file specification

Explanation: The file name you gave for /MESSAGES=*file* is not a valid log file specification. An additional message provides an explanation of the problem.

User Action: Perform the action suggested by the message indicating the problem, which could be correcting the spelling of a disk name, for example.

INVREST, invalidrestore: Improper restore — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

IOERROR, ioerror: System I/O error occurred — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

LIMCHK, limitcheck: Implementation limit exceeded — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

NODEVTYPE, couldn't determine device type for *queue*

Explanation: DECprint Utility software could not determine the device type. The device type logical is not defined.

User Action: Define the PSPRINT$*queue*_DEVICE_TYPE logical for queued printing jobs. See Section 3.1 for information on how to define this logical.

NOCURPT, nocurrentpoint: Path is empty — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

NOLOCMED, /MEDIUM must be used globally

Explanation: /MEDIUM should only be used globally, that is, after the DECprint Utility command verb and before any files.

User Action: Place /MEDIUM after the command verb and before the file names. Then enter your command again.

NOMIXDEV, all execution queues must have the same device type

Explanation: You specified a generic queue that points to execution queues with different device types.

User Action: Specify an execution queue or a generic queue that points to execution queues that have the same *device type*. Enter your command again.

NOPRINTER, /ATTACHED requires the terminal characteristic Printer Port

Explanation: Set terminal characterics do not indicate that the terminal has an attached printer. The DECprint Utility software requires this.

User Action: Perform the SET TERMINAL/PRINTER_PORT command and resubmit your print job.

NOTATTY, device TT: is not an available terminal

Explanation: The TT device is not a terminal or is not available. PSPRINT/ATTACHED only sends output to a terminal device.

User Action: Use the DECprint Utility /ATTACHED qualifier only from a normal DCL session.

OPENIN, error opening *filename* for reading

Explanation: For some reason, one of the POSTSCRIPT files could not be opened. Other messages accompany this error message to explain the problem.

User Action: Make corrections suggested by the accompanying messages.

RANGECHK, rangecheck: Argument out of bounds — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

STKOFLO, stackoverflow: Operand stack overflow — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

STKUFLO, stackunderflow: Operand stack underflow — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

SYNERR, syntaxerror: Input ended in string or procedure body — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

TOOCOMPLEX, print job too complex

Explanation: You have exceeded the number of qualifiers (or values for qualifiers) allowed on the DECprint Utility command line.

User Action: Simplify your print job. Submit it again with fewer qualifiers (or values for qualifiers) on the command line.

TYPCHK, typecheck: Argument of wrong type — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

UNDEF, undefined: Name not known — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error). If the application is supplied by Digital, submit a Software Performance Report (SPR).

UNDEFRES, undefinedresult: Number overflow or underflow — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

UNDFILNAM, undefinedfilename: File not found — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

UNMATCH, unmatchedmark: Expected mark not on stack — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

UNRECDEV, unrecognized device type: *device*

Explanation: You selected a device type for the printer that is not recognized by the DECprint Utility. Either the device type with the /ATTACHED qualifier is not valid or the device type value in the logical PSPRINT$*queue*_DEVICE_TYPE is not correct.

User Action: Check the /ATTACHED=*device-type* qualifier to make sure it is valid for the DECprint Utility command. Check that the PS$PRINT*queue*_DEVICE_TYPE logical is

defined correctly. Refer to Table 3–1. Then resubmit your file with the DECprint Utility command.

UNREGIST, unregistered: PostScript has encountered a system error — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

VMERROR, vmerror: PostScript virtual memory exhausted — offending command is *string*

Explanation: The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command represented by *string*.

User Action: Either the POSTSCRIPT file or the application that generated the POSTSCRIPT file is in error. If the application is supplied by Digital, submit a Software Performance Report (SPR).

Appendix A

Creating a Print Queue for the DECprint Utility

This appendix discusses how to set up print queues using the four optional files that are included with the DECprint Utility installation kit. These files include a startup command file, PSPRINT$STARTUP.TEMPLATE, and three command files that are invoked by this startup file. Together they set up and start your print queues.

If you already have print queues set up, you only need to define the device type logical (see Section 3.1). DECprint Utility software does not require that you use these command file to set up your print queue. These command files are provided for those system managers who would like help setting up print queues. Therefore, if you do not have print queues, you may want to use the command file SYS$STARTUP:PSPRINT$STARTUP.TEMPLATE.

NOTE

Running the PSPRINT$STARTUP command file on queues not created with this command file may not produce the desired result.

The following operations affect system files and should be performed by a system manager or cluster manager. These procedures include renaming and editing the SYS$STARTUP:PSPRINT$STARTUP.TEMPLATE and updating the system startup file. This appendix also discusses behaviors specific to use of PSPRINT$STARTUP.COM to start your print queues.

A.1 Renaming and Editing the Printer Startup Command File

The DECprint Utility, Version 1.0 provides an optional template for a printer startup file, SYS$STARTUP:PSPRINT$STARTUP.TEMPLATE. If you wish to use this command file to set up your print queues, you must rename the file, edit the file to reflect your own queue structure, and add a command to execute the command file in the system startup file.

A.1.1 Renaming the Startup File

Rename the printer startup template file using the COPY command as follows:

```
$ SET DEFAULT SYS$COMMON:[SYS$STARTUP]
$ COPY PSPRINT$STARTUP.TEMPLATE PSPRINT$STARTUP.COM
```

A.1.2 Editing the Startup File

This section includes portions of text from the printer startup command file. Text after an exclamation mark (!) to the end of the line denotes a comment that the startup procedure does not execute. Following this example and the instructions provided in PSPRINT$STARTUP.COM, use an editor of your choice and edit the file to meet your printing requirements.

Sample Queue Structure

PSPRINT$STARTUP provides a sample queue structure that includes commands to setup an execution queue on device TXA%%. Modify these commands or add commands to meet your requirements. Example A–1 shows the command that provides parameters to set up an execution queue:

Example A–1: Sample Execution Queue

```
$
$!+----------
$! Enter your queue structure below this line
$!----------
$
$! Here is a sample queue structure for one execution queue on device
$! TXA0:  Modify these commands and add additional ones to reflect
$! your desired queue structure.
$
$ on error then continue
$ @sys$startup:PSPRINT$execution_queue -
❶       PSPRINT  -                    ! P1 - Execution queue name
❷       txa%% -                       ! P2 - Device name
❸       "" -                          ! P3 - Device control library
❹       LJ250 -                       ! P4 - Printer device type
❺       "" -                          ! P5 - Default queue qualifiers
❻       "" -                          ! P6 - Communication speed
❼       ""                            ! P7 - Device characteristics
$
$ exit
```

@SYS$STARTUP:PSPRINT$EXECUTION_QUEUE executes the file PSPRINT$execution_queue.com. This file uses the following parameters from Example A–1 to set up your execution queue:

❶ Parameter **p1** *PSPRINT* is the queue name. Substitute the name of your execution queue.

❷ Parameter **p2** *txa%%* is name of the device that drives the execution queue. This parameter is used with the /ON qualifier of the INITIALIZE/QUEUE command. Substitute the name of the device that drives your execution queue. TXA%% is not a valid name. For example:

- txa0 — for an LJ250 on a serial line
- lta1 — for an LJ250 on a DECserver port
- foo::txa4 — for an LJ250 on TXA4 on node FOO in a cluster

❸ Parameter **p3** *""* indicates that no device control library is associated with the queue. The DECprint Utility does not require a device control library. However, you may want to associate a device control library with the queue, if you plan to use a DECprint Utility queue for other ANSI printing tasks.

❹ Parameter **p4** *LJ250* is the printer device type for the queue. Edit to include the device type for your printer.

❺ Parameter **p5** *""* selects no default qualifiers for the queue. Use this parameter to override or specify other qualifiers to the INITIALIZE/QUEUE command.

❻ Parameter **p6** *""* selects the default communications speed of 9600 BAUD. This parameter has no effect for LAT devices.

❼ Parameter **p7** *""* selects no additional device characteristics for the queue. If you want the device spooled, replace the *""* in **p7** with *"/SPOOL=(PSPRINT,disk-name)"*.

A.2 Updating the System Startup File

To update the system startup file:

Edit the SYS$COMMON:[SYSMGR]SYSTARTUP_V5.COM file and check for the following command procedure call:

```
$ @SYS$STARTUP:PSPRINT$STARTUP
```

If this line is not present, insert it.

NOTE

If your printer is configured on a DECserver in a local area Ethernet, execute PSPRINT$STARTUP.COM after LTLOAD.COM.

A.3 PSPRINT$STARTUP Queue Setup Topics

This section discusses how using the PSPRINT$STARTUP command file differs from standard queue initialization procedures.

A.3.1 Device Control Library

PSPRINT$STARTUP.COM does not associate a device control library with the DECprint Utility queue. DECprint Utility printing does not require a device control library. However, if you plan to use the DECprint Utility queue for other ANSI printing tasks, you may want to associate a device control library with the queue. To do this, add the device control library name to **p3** in the execution queue definition in PSPRINT$STARTUP.COM.

A.3.2 Terminal Device Type

PSPRINT$STARTUP.COM sets the terminal device type to unknown. This ensures that the command file is more robust and printer independent. Some of the printer device types supported by the DECprint Utility software are unknown to the VMS terminal driver.

A.3.3 Separation Pages

PSPRINT$STARTUP.COM initializes the queues with only the job flag page, instead of burst and trailer pages. If you want to print transparencies on an LJ250 printer or if your printer uses cut sheet media, you may want to disable all job and file separation pages.

Do not use file separation pages.

A.3.4 /PROCESSOR Qualifier

The PSPRINT$EXECUTION_QUEUE.COM, invoked by PSPRINT$STARTUP.COM, selects the setting of the /PROCESSOR qualifier as follows:

- PRTSMB — if the device is a serial line communications card
- LATSYM — if the device is a local area transport (LAT) port

PSPRINT$EXECUTION_QUEUE.COM can only determine this on the CPU where the symbiont is running. In a cluster, a processor is only initialized on a node that is running the symbiont.

For queue characteristics to be correct, that is, for the right processor to be selected:

- The node that runs the symbiont must boot and run the startup file.

PSPRINT$EXECUTION_QUEUE.COM does not select the processor in the following instances:

- The device does not exist. PSPRINT$EXECUTION_QUEUE.COM assumes that no information exists to determine which symbiont to use.

- The device is already allocated.
  PSPRINT$EXECUTION_QUEUE.COM assumes that the queue is started and the processor already selected.

Appendix B

PostScript Language Extensions

Every POSTSCRIPT device has features that do not apply, or apply differently, to other POSTSCRIPT devices. For example, not all devices can select different media sizes. In addition, *system parameters*, such as a system password and timeout values, may not apply or may apply differently to various devices.

Extensions to the POSTSCRIPT language allow you to take advantage of the unique features of your device. This appendix describes the POSTSCRIPT extension operators used with DECprint Utility software. For information on the standard POSTSCRIPT operators, refer to the *POSTSCRIPT Language Reference Manual*.

The DECprint Utility software does not support all extension operators described for Digital POSTSCRIPT devices. If you try to use an extension that the DECprint Utility software does not have, POSTSCRIPT returns an UNDEF error. If the undefined name is an extension operator, and if ignoring the extension operator does no harm, then you can do the following to successfully convert and print your file:

1. Create a POSTSCRIPT file that defines the unknown extension operator.

   In some cases, the undefined extension operators have operands. The procedure must remove (pop) the operands from the stack. Examine the file causing the problem to determine the correct number of pops. For example, **letter** does not take an operand. For **letter**, create a POSTSCRIPT file that contains the following line:

   ```
   /letter {} def
   ```

For example, operator **setjobtimeout** takes one operand. To define **setjobtimeout**, create a POSTSCRIPT file with the following line:

```
/setjobtimeout {pop} def
```

For example, operator **setdefaulttimeouts** takes three operands. To define **setdefaulttimeouts**, create a POSTSCRIPT file that contains the following line:

```
/setdefaulttimeout {pop pop pop} def
```

2. Name this file on the DECprint Utility command line before the POSTSCRIPT file causing the UNDEF error. For example, if you named the file *letter.ps* and the file causing the error is named *fileE.ps*, then enter the following command:

```
$ PSPRINT/QUEUE=lj250_b letter.ps, fileE.ps
```

As shown in Example B–1, you can use the **where** operator to determine if a name is defined. (See the *POSTSCRIPT Language Reference Manual* for information about **where**.)

Example B–1: Determining Whether a Name Is Defined

```
% Check whether the name 'setduplexmode' is defined.

statusdict begin
      /setduplexmode where
      {                 % yes, it's defined

          pop           % discard the dictionary returned by where.

                        % ...use setduplexmode...
         }

      {                 % no, it's not defined

                        % ...do something else...
         }

      ifelse
      end
```

This describes the Digital POSTSCRIPT extensions required to specify the POSTSCRIPT model used by the DECprint Utility software. For more information on Digital POSTSCRIPT language extensions, refer to the *Programmer's Supplement: POSTSCRIPT Printers*.

B.1 Using PostScript Extensions

POSTSCRIPT stores all system-specific extensions in a dictionary named **statusdict**. This dictionary contains objects that constitute the POSTSCRIPT extensions for each Digital POSTSCRIPT device. By comparison, the **systemdict** and **userdict** dictionaries contain only the standard POSTSCRIPT operator and procedure definitions.

To use a POSTSCRIPT extension, you must explicitly access **statusdict**. As shown in Example B–2, you execute **statusdict begin** to push **statusdict** on the dictionary stack. Until POSTSCRIPT consumes the matching **end** operator, you can directly execute the objects defined in **statusdict**.

Example B–2: Accessing Extensions Located in statusdict

```
statusdict begin
a3tray              %  Digital-specific operator
.
.
.
end
```

Like standard POSTSCRIPT operators, some extension operators take an operand. Others return a result. Some extensions neither take an operand nor return a result. (For information on the syntax of extension operators see the *Programmer's Supplement: POSTSCRIPT Printers*.)

B.2 Persistent and Volatile Parameters

There are two kinds of system parameters: persistent and volatile. *Persistent parameter* values persist even after the host-based POSTSCRIPT interpreter has been reset. Persistent values for the DECprint Utility software are constants; they cannot be modified. Table B–1 lists persistent parameters specific to the DECprint Utility software and shows the Digital-specific operators and values used to access them.

Table B–1: Persistent Parameters

Parameter	Operator or Value
Identification	**componentids**, **revision**
Name of product	**product** (string)

Volatile parameters, on the other hand, remain in effect only during the execution of a single job. When a job completes, volatile parameters revert to the values in effect before the job began. Table B–2 lists the volatile parameters for the DECprint Utility software and the Digital-specific operators used to access them.

Table B–2: Volatile Parameters

Parameter	Operator or Value
Page count	**pagecount**, **sheetcount**
Paper size and load direction	**papersize**
Name of current job	**jobname**

B.3 Identification Extensions

Some POSTSCRIPT extensions identify the type of device you are using. These extenions are used for information only. You should not use them as a basis for decision-making in POSTSCRIPT code, as the code is then device-dependent.

B.3.1 Product

The name **product** is defined in **statusdict**. The product name identifies the product as a whole, not its specific capabilities.

The value returned by **product** depends on the type of printer you use. The value is the string PSprint/*printer*, where *printer* is one of the following keywords:

- LA75
- LA210
- LA324

- LA324 Color
- LJ250
- LN03

For example, when you print to an LJ250 Companion Color printer, the value of **product** is PSprint/LJ250.

You should not change the value of **product** from the name Digital provides with the device.

B.3.2 Device Components

The **componentids** extension returns a list of the components of the device. For the DECprint Utility software, **componentids** returns a single item, other than the mark — a string that indicates the version number of the host-based POSTSCRIPT interpreter.

B.3.3 Version

The **version** operator identifies the version of the base portion of the POSTSCRIPT interpreter in your device. This is a standard POSTSCRIPT operator.

B.3.4 Revision

The name **revision** is defined in **statusdict**. It identifies the revision of the device-dependent portion of the POSTSCRIPT interpreter in your device.

B.3.5 Job Name

The current job name is a string that appears as part of all status responses generated during a print job. By default, the value is a null string. Your POSTSCRIPT program can set the job name by associating it with **jobname** in **statusdict**.

B.4 Pagecount and Sheetcount Operators

The origin for the value of the **pagecount** operator is the first page imaged by the current invocation of the DECprint Utility command. This operator counts each imaged side of a sheet as one page. So when you image multiple pages on one side of a sheet, the value returned by **pagecount** increases by 1. The **pagecount** counter is reset at the end of every DECprint Utility job.

The **sheetcount** operator also starts its count at the first sheet imaged and returns the number of sheets imaged by the current DECprint Utility command.

Appendix C

Color Extensions for the PostScript Language

Appendix C contains the information presented by Abode in *POSTSCRIPT Language Color Extensions*, October 1988. The text has been reformatted for processing with VAX DOCUMENT, Version 1.2.

For information on the standard POSTSCRIPT operators, refer to the *POSTSCRIPT Language Reference Manual*. See Appendix B for POSTSCRIPT language extensions that apply to the DECprint Utility.

C.1 About the PostScript Language Color Extensions

The POSTSCRIPT language has been extended to provide more complete color functionality. This includes cyan-magenta-yellow-black (CMYK) color specification, black generation and undercolor removal functions, screen and transfer functions for four separate color components, and extension of the **image** concept to a **colorimage** operator that accepts multiple color components.

C.2 About this Appendix

This appendix describes the extensions to the POSTSCRIPT language that support new color functionality, and it provides detailed information on the associated color operators. The third section is a general introduction to color functionality in the POSTSCRIPT language. The fourth section discusses the extended color capabilities of the POSTSCRIPT language. The fifth section is an alphabetical listing

of descriptions for all color operators that have been added to the POSTSCRIPT language.

C.3 Introduction

The earlier versions of the POSTSCRIPT language support color using the **setrgbcolor** and **sethsbcolor** operators, which enable the POSTSCRIPT interpreter to paint filled regions, strokes, image masks, and characters in color. On black-and-white machines, these operators generate an equivalent gray shade, which is printed or displayed.

To support color more fully, the POSTSCRIPT language has been extended to provide the following functions:

- Most significantly, multiple color images: the **colorimage** operator renders a multiple color image; it functions analogously to the **image** operator, but uses red-green-blue (RGB) or cyan-magenta-yellow-black (CMYK) color input and generates full-color output.
- Halftone screen definitions: the **setcolorscreen** operator specifies halftone screen definitions for red, green, blue, and gray, or cyan, magenta, yellow, and black concurrently; it is the logical expansion of **setscreen** as it takes the same three arguments to define each screen for each printing ink.
- Color correction: the **setcolortransfer** operator sets the transfer function parameters for red, green, blue, and gray; it is an expansion of **settransfer** to four color components. The **setblackgeneration** operator provides a black generation function that establishes a black component from a cyan, magenta, and yellow specification. The **setundercolorremoval** operator provides undercolor removal from the cyan, magenta, and yellow components to compensate for the addition of black by the black generation function.
- CMYK color specification: the **setcmykcolor** operator allows the user to set the current color in the graphics state to a cyan-magenta-yellow-black color directly, bypassing the color correction operators.

The POSTSCRIPT language supports one-color, three-color, and four-color output devices. The color devices can be binary (one-bit-per-pixel per color component) or gray-scale (multiple-bits-per-pixel per color component, representing a range of intensities of each color component). A binary device uses halftoning to produce intermediate

shades of its color components. If a device has eight-bits-per-pixel per component, it is called a *full gray-scale* device, and uses no halftoning. Devices with more than one and fewer than eight bits per pixel use a combination of built-in intensities and halftoning to produce the full range of desired shades of their color components. Three-color devices may be either red-green-blue (RGB), typically for displays and film recorders, or cyan-magenta-yellow (CMY) for printers. Four-color devices are cyan-magenta-yellow-black (CMYK) for color printers and color separation making devices.

The color operators described in this document are available in two forms. New versions of the POSTSCRIPT interpreter have these operators built in. For older versions of the POSTSCRIPT interpreter, a package of POSTSCRIPT language programs that emulate these operators will be available.

C.4 New Features

The color extensions to the POSTSCRIPT language include CMYK color specification, black generation, and undercolor removal for tunable conversion from RGB to CMYK color specification, and operators that expand the scope of screens, transfer functions, and images to full color input and output. These new color features are described conceptually in the sections below.

C.4.1 CMYK Color Specification

Colors are formed either by adding light to black or by subtracting light from white. Computer displays and film recorders typically add colors, while printing inks typically subtract colors. These two methods for forming colors give rise to the two major complementary color specifications, the additive RGB specification and the subtractive CMYK specification.

Accordingly, a color component in these specifications either indicates the amount of light it reflects or the amount of light it absorbs. Each one of three standard printing process colors, *cyan*, *magenta*, and *yellow*, absorb one of the standard light components, *red*, *green*, and *blue*, respectively. *Black*, a fourth standard printing process color, absorbs all components of light. In the red-green-blue (RGB) color specification, each of its red, green, and blue components is associated with a real number between 0.0 and 1.0, inclusive, where 0.0 represents dark (no light) and 1.0 represents full light. In the

cyan-magenta-yellow-black (CMYK) color specification, each of the four components are associated with real numbers between 0.0 and 1.0 inclusive, where 0.0 represents full light (no ink), and 1.0 represents dark (full ink).

The following equations demonstrate the relationship between the RGB and CMYK color specifications. Since cyan is the absence of red light,

$$cyan = 1.0 - red.$$

Similarly,

$$magenta = 1.0 - green$$

and

$$ye\ llow = 1.0 - blue,$$

so that a color that is 0.2 red, 0.7 green, and 0.4 blue can also be expressed as $1.0 - 0.\ \ 2\ =\ 0.8\ cyan$, $1.0 - 0.7\ =\ 0.3\ magenta$, and $1.0 - 0.4\ =\ 0.6\ yellow$. To improve the fidelity of blacks and grays, a fourth process color, black, is often available on color printers. Just as red in the RGB specification is the opposite of cyan in the CMYK specification, a black value is the opposite to a PostScript language gray value; that is,

$$black = 1.0 - gray.$$

C.4.2 Black Generation and Undercolor Removal

Logically, cyan, magenta, and yellow are all that are needed to generate a printing color completely. Thus an equal percentage of cyan, magenta, and yellow should create the equivalent percentage of black. In reality, colored printing inks do not mix perfectly, and such combinations often form dark brown shades instead. Thus, it is often desirable to substitute real black ink for the mixed-black portion of a color to obtain a truer color rendition on a printer.

Black generation is the process of calculating the amount of black to be used when trying to print a particular color. Undercolor removal is the process of reducing the amount of cyan, magenta, and yellow components to compensate for the amount of black that was added by the black generation. Flexibility in performing these functions is important for achieving good results under different printing conditions.

The **setblackgeneration** operator provides the functionality to generate extra black, no black, or a black value equal to all or a fraction of the minimum values of cyan, magenta, and yellow (see Section C.5). Its argument is a procedure that takes one numeric argument, the minimum value of *user cyan*, *magenta*, and *yellow* color components, and returns a single numeric result, the *user black* value (where a user color component value is that specified in the POSTSCRIPT language program before application of the corresponding transfer function). This procedure is automatically applied whenever **setrgbcolor**, the three-color case of **colorimage** (see Section C.3), or **sethsbcolor** specify a color. This *user black* value is then mapped to a *device black* value by applying the gray transfer function to its difference from 1.0 and subtracting the result from 1.0 (see **setcolortransfer** in Section C.5). The black generation function is not applied when **setgray**, **setcmykcolor** (see Section C.5), or the one- or four-color cases of **colorimage** specify colors. This computed black value is used only when outputting on four-color output devices.

The **setundercolorremoval** operator (see Section C.5) provides functionality to remove some amount of color from each of the cyan, magenta, and yellow components. This amount could be exactly the same amount as was generated to make the black component, zero (so no color is removed from the cyan, magenta and yellow components), some fraction of the black amount, or even a negative amount. Like **setblackgeneration**, this operator permits considerable flexibility in color correction.

The argument to **setundercolorremoval** is a procedure that takes one numeric argument, the minimum value of *user cyan*, *magenta*, and *yellow* color components, and returns a single numeric result that is subtracted from each of these original user color components. This procedure is applied whenever **setrgbcolor**, the three-color case of **colorimage** (see Section C.5), or **sethsbcolor** specify a color. After subtracting the value generated in the above mapping from the color components and resetting negative values to 0.0 and values greater than 1.0 to 1.0, each component is subtracted from 1.0 to yield *red*, *green*, and *blue* components. Each of these components is mapped into a *device color component* using its respective transfer function (see **setcolortransfer** in Section C.5). Undercolor removal is not applied when **setgray**, **setcmykcolor**, or the one- or four-color cases of **colorimage** specify a color. Undercolor removal is used only when outputting on four-color devices.

The following equations define the complete color transformation process from RGB to CMYK. The values red_u, grn_u, blu_u, gry_u, cyn_u, mag_u, yel_u, and blk_u are the (input) values of *user red*, *green*, *blue*, *gray*, *cyan*, *magenta*, *yellow*, and *black*, respectively. *UCR( )* is the undercolor removal function. *BG( )* is the black generation function. *RedT*, *GrnT*, *BluT*, and *GryT* are the red, green, blue and gray transfer functions (see **setcolortransfer** in Section C.5), respectively. The values red_d, grn_d, blu_d, gry_d, cyn_d, mag_d, yel_d, and blk_d are the (output) values of *device red*, *green*, *blue*, *gray*, *cyan*, *magenta*, *yellow*, and *black*, respectively.

$$
\begin{aligned}
cyn_u &= 1.0 - red_u \\
mag_u &= 1.0 - grn_u \\
yel_u &= 1.0 - blu_u \\
gry_u &= 1.0 \\
blk_u &= 0.0 \\
k &= Min(cyn_u, mag_u, yel_u) \\
u &= UCR(k) \\
red_d &= RedT(1.0 - Min(1.0, Max(0.0, cyn_u - u))) \\
grn_d &= GrnT(1.0 - Min(1.0, Max(0.0, mag_u - u))) \\
blu_d &= BluT(1.0 - Min(1.0, Max(0.0, yel_u - u))) \\
gry_d &= GryT(1.0 - BG(k)) \\
cyn_d &= 1.0 - red_d \\
mag_d &= 1.0 - grn_d \\
yel_d &= 1.0 - blu_d \\
blk_d &= 1.0 - gry_d
\end{aligned}
$$

C.4.3 Direct CMYK Color Specification

For the most demanding cases, color matching can require more complicated methods than those described above. The **setcmykcolor** operator and the four-color case of the **colorimage** operator bypass the black generation and undercolor removal operations, allowing the knowledgable user to specify the cyan, magenta, yellow, and black color components for a particular device. This operator provides no correction other than the transfer functions **setcolortransfer** specifies (see Section C.5); the results are device dependent.

The following equations define the complete color transformation process for the **setcmykcolor** operator and the four-color case of the **colorimage** operator. The values are as defined in Black Generation and Undercolor Removal Section C.4.2. These equations are equivalent to those given above except for the omission of the black generation and undercolor removal steps.

$$
\begin{aligned}
red_u &= 1.0 - cyn_u \\
grn_u &= 1.0 - mag_u \\
blu_u &= 1.0 - yel_u \\
gry_u &= 1.0 - blk_u \\
red_d &= RedT(red_u) \\
rn_d &= GrnT(grn_u) \\
blu_d &= BluT(blu_u) \\
gry_d &= GryT(gry_u) \\
cyn_d &= 1.0 - red_d \\
mag_d &= 1.0 - grn_d \\
yel_d &= 1.0 - blu_d \\
blk_d &= 1.0 - gry_d
\end{aligned}
$$

C.4.4 Color Screens, Transfer Functions, and Images

The operators **setcolorscreen**, **currentcolorscreen**, **setcolortransfer**, and **currentcolortransfer** provide an expansion of the operators **setscreen**, **currentscreen**, **settransfer**, and **currenttransfer**, respectively, by setting up a screen and a transfer function for each color component. The **colorimage** operator provides an expansion of the **image** operator to allow samples of one, three, or four color components. (See Section C.5 for more details.)

C.4.4.1 colorimage

colorimage is the logical expansion of **image** to handle sampled images whose samples are composed of more than one color component. The initial arguments to **colorimage** are the same as those for **image**. The final arguments differ according to the number of color components per sample and according to the encoding method.

The arguments to **colorimage** are as follows:

width height bits/component matrix $proc_0$ [...$proc_{ncolors-1}$] multiproc ncolors

(See Section C.5 for precise definitions of these arguments.)

ncolors describes the number of color components in each sample. Legal values for *ncolors* are 1 (gray-level samples only), 3 (RGB samples), or 4 (CMYK samples). *multiproc* distinguishes between encoding methods. The possibilities for *multiproc* are *false* (single procedure, color components are bunched together) or *true* (multiple procedures, one per color, components are separated into strings of like colors).

The legal variations of *ncolors* and *multiproc* allow the following possibilities (where proc subscripts have been changed to words to indicate the purpose of each proc):

w h b/c matrix $proc_{gray}$ false 1
w h b/c matrix $proc_{gray}$ true 1
w h b/c matrix $proc_{rgb}$ false 3
w h b/c matrix $proc_{red}$ $proc_{green}$$proc_{blue}$ true 3
w h b/c matrix $proc_{cmyk}$ false 4
w h b/c matrix $proc_{cyan}$ $proc_{magenta}$$proc_{yellow}$ $proc_{black}$ true 4

The first two variations here are both equivalent to

w h b/s matrix $proc_{gray}$ **image**

C.4.4.1.1 Data formats for colorimage operator

As indicated above, the **colorimage** operator has two forms, distinguished by its *multiproc* argument.

The single-procedure form is most useful if sample input is taken from a source that has already merged the color components. This form provides samples for which each RGB triple or CMYK quadruple is packed together in the string result of the procedure, using one of the following bit formats (where the high-order bit is shown on the left):

Bits/comp.	RGB Format				
1	RGBRGBRG	BRGBRGBR	GBRGBRGB	RGBRGBRG	....
2	RRGGBBRR	GGBBRRGG	BBRRGGBB	RRGGBBRR	....
4	RRRRGGGG	BBBBRRRR	GGGGBBBB	RRRRGGGG	....
8	RRRRRRRR	GGGGGGGG	BBBBBBBB	RRRRRRRR	....

or

Bits/comp.	CMYK Format				
1	CMYKCMYK	CMYKCMYK	CMYKCMYK	CMYKCMYK	....
2	CCMMYYKK	CCMMYYKK	CCMMYYKK	CCMMYYKK	....
4	CCCCMMMM	YYYYKKKK	CCCCMMMM	YYYYKKKK	....

Bits/comp.	CMYK Format				
8	CCCCCCCC	MMMMMMMM	YYYYYYYY	KKKKKKKK	....

The multiple-procedure form expects each procedure to return a string of values for only one color component per sample, using the same format as strings returned by the *proc* argument of the **image** operator. For a three-color image, $proc_0$ returns red values, $proc_1$ returns green values, and $proc_2$ returns blue values. For a four-color image, $proc_0$ returns cyan values, $proc_1$ returns magenta values, $proc_2$ returns yellow values, and $proc_3$ returns black values. The **colorimage** operator calls each of these procedures in turn, starting with $proc_0$ and continuing with $proc_1$, $proc_2$, and, if available, $proc_3$. When the **colorimage** operator needs more samples, it calls these procedures again in the same order. The color procedures must use separate strings for the three or four results of the three or four procedures, i.e., reusing the red string for the green values can result in some of the red values being lost. Also, the three or four procedures must return strings of identical lengths within each cycle of three or four calls.

The multiple-procedure form is most useful when color sample data are taken from separate color scanner passes. The **colorimage** operator requires the color data to be interleaved, as it must have all three or four components of any sample at the same time to do its work. The single-procedure form interleaves the data at the sample level; this may be convenient only if the data are already in that form when preparing the POSTSCRIPT language page description. The multiple-procedure form allows interleaving at a much coarser level. Typically, each procedure of the multiple-procedure form returns components for some number of scan lines of samples, where the number of components returned at each call is limited by the string storage available in the POSTSCRIPT interpreter.

C.4.4.1.2 Examples for colorimage operator

The following examples illustrate the use of the **colorimage** operator:

EXAMPLE 1:

```
/rgbstr 192 string def    % string to hold 256 two-bit samples
                          %   each of red, green, and blue data
45 140 translate          % locate lower left corner of image
132 132 scale             % map image to 132 point square
256 256 2                 % dimensions of source image
[256 0 0 -256 0 256]      % map unit square to source
{currentfile              % read image date from program file
rgbstr readhexstring pop}
false 3                   % single proc, 3 colors, bit format:
                          %   rrggbbrr ggbbrrgg bbrrggbb ...
colorimage
94a1bec8c0b371a3a5c4d281 ...   (98304 hex digits of image data)
```

The above is a one-procedure 2-bit RGB image. The base-4 representation of the hex data is

```
    2110 2201 2332 ...
```

which is composed of the following color samples:

```
r=2 g=1 b=1 r=0   g=2 b=2 r=0 g=1   b=2 r=3 g=3 b=2 ...
```

EXAMPLE 2:

```
/rstr 256 string def      % string to hold 256 8-bit red samples
/gstr 256 string def      % string to hold 256 8-bit green samples
                          %  (distinct from rstr)
/bstr 256 string def      % string to hold 256 8-bit blue samples
                          %  (distinct from rstr and bstr)
45 140 translate          % locate lower left corner of image
132 132 scale             % map image to 132 point square
256 256 8                 % dimensions of source image
[256 0 0 -256 0 256]      % map unit square to source
{currentfile rstr readhexstring pop}
                          % read red data from program file
{currentfile gstr readhexstring pop}
                          % read green data from program file
{currentfile bstr readhexstring pop}
                          % read blue data from program file
true 3                    % multiple proc, 3 colors
colorimage
7b5e606969615365556a6a66 ...   (512 hex digits of red data)
88868d848a92878578787a82 ...   (512 hex digits of green data)
62717c7b736e707d7b6a7c79 ...   (512 hex digits of blue data)
7dd8b8d8c837d8b8e9284878e ...  (512 hex digits of red data)
2788b838b8e8e86868988908 ...   (512 hex digits of green data)
81817d857f85858290949487 ...   (512 hex digits of blue data)
... (390144 more hex digits of RGB data, cycling as above)
```

The above is a three-procedure 8-bit RGB image. The initial samples for each color, in hex, are:

red:	7b	5e	60	69	...
green:	88	86	8d	84	...
blue:	62	71	7c	7b	...

EXAMPLE 3:

```
/cstr 128 string def      % string to hold 256 4-bit cyan samples
/mstr 128 string def      % string to hold 256 4-bit magenta samples
                          %  (distinct from cstr)
/ystr 128 string def      % string to hold 256 4-bit yellow samples
                          %  (distinct from cstr and mstr)
/kstr 128 string def      % string to hold 256 4-bit black samples
                          %  (distinct from cstr, mstr, and ystr)
45 140 translate          % locate lower left corner of image
132 132 scale             % map image to 132 point square
256 256 4                 % dimensions of source image
[256 0 0 -256 0 256]      % map unit square to source
{currentfile cstr readhexstring pop}
                          % read cyan data from program file
{currentfile mstr readhexstring pop}
                          % read magenta data from program file
{currentfile ystr readhexstring pop}
                          % read yellow data from program file
{currentfile kstr readhexstring pop}
                          % read black data from program file
true 4                    % multiple proc, 4 colors
colorimage
e1d8caa57b655b6779606b72 ...   (256 hex digits of cyan data)
6bdbb867b9fb6a4859569989 ...   (256 hex digits of magenta data)
996796e639cc0b29f94736c7 ...   (256 hex digits of yellow data)
c9c0cad0d3cad2b7c9e2d7d8 ...   (256 hex digits of black data)
5d2d6d7d4d3d1d4d6c9d4d9d ...   (256 hex digits of cyan data)
4cdcfd4d6d1d8d7d5d4d2d2d ...   (256 hex digits of magenta data)
d2b7c9e2d7d8d8cbbac2d9d8 ...   (256 hex digits of yellow data)
88ae96632a70f6f4d8d9d9d8 ...   (256 hex digits of black data)
... (260096 more hex digits of CMYK data, cycling as above)
```

The above is a four-procedure four-bit CMYK image. The initial sample for each color, in hex, are

cyan:	e 1 d 8 c a a 5 ...
magenta:	6 b d b b 8 6 7 ...
yellow:	9 9 6 7 9 6 e 6 ...
black:	c 9 c 0 c a d 0 ...

EXAMPLE 4:

```
/cstr 1024 string def     % string to hold 1024 8-bit cyan samples
/mstr 1024 string def     % string to hold 1024 8-bit magenta samples
                          %  (distinct from cstr)
/ystr 1024 string def     % string to hold 1024 8-bit yellow samples
                          %  (distinct from cstr and mstr)
/kstr 1024 string def     % string to hold 1024 8-bit black samples
                          %  (distinct from cstr, mstr, and ystr)
/cfile (img/smp.c) (r) file def   % binary file containing 1048576 8-bit
                                  %   cyan samples
/mfile (img/smp.m) (r) file def   % binary file containing 1048576 8-bit
                                  %   magenta samples
/yfile (img/smp.y) (r) file def   % binary file containing 1048576 8-bit
                                  %   yellow samples
/kfile (img/smp.k) (r) file def   % binary file containing 1048576 8-bit
                                  %   black samples
36 126 translate          % locate lower left corner of image
540 540 scale             % map image to 540 point square
1024 1024 8               % dimensions of source image
[1024 0 0 -1024 0 1024]   % map unit square to source
{cfile cstr readstring pop}       % read cyan data from imp/smp.c
{mfile mstr readstring pop}       % read magenta data from imp/smp.m
{yfile ystr readstring pop}       % read yellow data from imp/smp.y
{kfile kstr readstring pop}       % read black data from imp/smp.k
true 4                    % multiple proc, 4 colors
colorimage
```

The above is a four-procedure 8-bit CMYK image, with cyan, magenta, yellow, and black samples taken from the files *img/smp.c*, *img/smp.m*, *img/smp.y*, and *img/smp.k*, respectively. This example only applies to a POSTSCRIPT interpreter that has a file system.

C.4.5 Color Implementations

Each POSTSCRIPT interpreter contains default color output methods that correspond to the printer to which it is attached. If the attached printer is a direct-color binary device, the standard output method produces three- or four-color output. If the attached printer is a gray-scale color device, then the POSTSCRIPT interpreter uses a gray-scale three- or four-color output method. If the attached printer is a black-and-white device, then the default output method produces a single black-and-white rendition of each page described.

Black-and-white printers also have the capability of producing color separations. These are three or four black-and-white pages for each page described. Each output page corresponds to the output for one color component. These separations are only useful if they are used in

another printing process in which they are combined with color to form the intended full-color output.

The method by which a POSTSCRIPT language program informs the POSTSCRIPT interpreter that color separations are desired is still to be determined. This color separation capability will not be provided in the backward compatibility packages, as it requires considerable internal changes in the POSTSCRIPT interpreter.

C.5 Operators

Each color extension operator description shows syntax in the following format:

operand$_1$ *...operand*$_n$ ***operator*** *(result*$_1$ *...result*$_m$*)*

The operands precede and are input to the operator, with *operand*$_n$ being the top element on the operand stack. The operator pops these objects from the operand stack and consumes them. After executing, the operator leaves *result*$_1$ through *result*$_m$ on the operand stack, with *result*$_m$ being the top element.

A dash in an operand position indicates that the operator expects no operands. Likewise, a dash in a result position indicates that the operator returns no results, as follows:

— operator —

colorimage

Renders a sampled image, whose samples contain one, three, or four color components on the current page.

Use of the **colorimage** operator after a **setcachedevice** within the context of a **BuildChar** procedure is not permitted (an **undefined** error results).

Format

width height bits/component matrix $proc_0$ *[...*$proc_{ncolors-1}$*] multiproc ncolors* **colorimage** —

width
The width of the image, in samples. This argument has the same meaning as in the **image** operator.

height
The height of the image, in samples. This argument has the same meaning as in the **image** operator.

bits/component
The number of bits per each component of each sample. This value applies equally to all color components, and has the same meaning as in the **image** operator.

matrix
Transforms the samples into a unit square. This argument has the same meaning as in the **image** operator.

$proc_0$ [... $proc_{ncolors-1}$]
Procedures that return strings of samples to be imaged. **colorimage** permits its $proc_i$ arguments to return RGB or CMYK sample values rather than the single-color (gray) values as are returned by the *proc* argument of the **image** operator. For a detailed description of the data formats and how the $proc_i$ procedures are called, see Section C.4.4.1.1.

Use of **setcolorscreen**, **setcolortransfer**, **setscreen**, or **settransfer** by any of the $proc_i$ procedures causes unpredictable results.

multiproc
A boolean that distinguishes between two forms of the **colorimage** operator: *false* indicates the single-procedure form, argument ($proc_0$), and *true* indicates the multiple-procedure form, which requires one procedure argument per sample color ($proc_0$ ... $proc_{ncolors-1}$), i.e., three procedure arguments for RGB samples or four procedure arguments for CMYK samples. If the *ncolors* argument is 1, then there is only one procedure argument, $proc_0$, regardless of the value of the *multiproc* argument.

ncolors
The number of color components represented in the samples, either 1, 3, or 4. If *ncolors* is 1, the samples have only one component, a gray component, and the operation of **colorimage** is equivalent to that of **image** with the same five initial arguments. If the *ncolors* argument is 3, the **colorimage** operator takes RGB (light-high) samples. If the *ncolors* argument is 4, the **colorimage** operator takes CMYK (dark-high) samples. On a four-color (CMYK) machine, the POSTSCRIPT interpreter converts a three-color (RGB) image to CMYK using the black generation and undercolor removal functions, whereas a four-color (CMYK) image bypasses these operations.

Errors

limitcheck
A POSTSCRIPT implementation limit has been exceeded.

rangecheck
An operand value is not in the allowed range of values.

stackunderflow
The operand stack is empty when the operator expects a value.

typecheck
An operand value is not the type that the operator expects.

undefined
Attempted to use the **colorimage** operator after a **setcachedevice** in the context of a **BuildChar** procedure.

undefinedresult
A numeric computation produced a meaningless result.

currentblackgeneration

Returns the current black generation function in the graphics state (see **setblackgeneration**).

Format — currentblackgeneration *proc*

proc
The current black generation function.

Error

stackoverflow
The operand stack is full, preventing the operator from adding the returned values to the stack.

currentcmykcolor

Returns the four components of the current color in the graphics state according to the cyan-magenta-yellow-black color specification (see **setcmykcolor**).

The **currentgray** operator returns a weighted average of all four color components. Applying it is the equivalent of the following use of **currentcmykcolor**:

```
1.0 currentcmykcolor 4 1 roll 0.11 mul 3 1 roll 0.59 mul
exch 0.30 mul add add add sub dup 0.0 lt {pop 0.0} if
```

Format — currentcmykcolor *cyan magenta yellow black*

cyan
The cyan component of the current color.

magenta
The magenta component of the current color.

yellow
The yellow component of the current color.

black
The black component of the current color.

Error

stackoverflow
The operand stack is full, preventing the operator from adding the returned values to the stack.

currentcolorscreen

Returns all 12 current halftone screen parameters in the graphics state (see **setcolorscreen**).

The **currentcolorscreen** operator is the logical expansion of **currentscreen** to four color components. Applying the **currentscreen** operator returns the three paramters describing the gray/black screen. It is the equivalent of the following use of **currentcolorscreen**:

```
currentcolorscreen 12 3 roll 9 {pop} repeat
```

Format — currentcolorscreen *r/c-freq r/c-angle r/c-proc g/m-freq g/m-angle g/m-proc b/y-freq b/y-angle b/y-proc g/b-freq g/b-angle g/b-proc*

r/c-freq r/c-angle r/c-proc
The frequency, angle, and procedure for the red/cyan screen.

g/m-freq g/m-angle g/m-proc
The frequency, angle, and procedure for the green/magenta screen.

b/y-freq b/y-angle b/y-proc
The frequency, angle, and procedure for the blue/yellow screen.

g/b-freq g/b-angle g/b-proc
The frequency, angle, and procedure for the gray/black screen.

Error

stackoverflow
The operand stack is full, preventing the operator from adding the returned values to the stack.

currentcolortransfer

Returns the current transfer functions in the graphics state for each of the four color components (see **setcolortransfer**).

The **currentcolortransfer** operator is the logical expansion of **currenttransfer** to four color components. Applying the **currenttransfer** operator returns the gray transfer function. It is the equivalent of the following use of **currentcolortransfer**:

```
currentcolortransfer 4 1 roll pop pop pop
```

Format — currentcolortransfer *redproc greenproc blueproc grayproc*

redproc
The transfer function for the red component.

greenproc
The transfer function for the green component.

blueproc
The transfer function for the blue component.

grayproc
The transfer function for the gray component.

Error

stackoverflow
The operand stack is full, preventing the operator from adding the returned values to the stack.

currentundercolorremoval

Returns the current undercolor removal function in the graphics state (see **setundercolorremoval**).

Format — currentundercolorremoval *proc*

proc
The current undercolor removal function.

Error

stackoverflow
The operand stack is full, preventing the operator from adding the returned values to the stack.

setblackgeneration

Sets the current black generation function parameter in the graphics state.

The use of **setblackgeneration** after a **setcachedevice** operation within the scope of a **BuildChar** procedure is not permitted (an **undefined** error results).

Format *proc* setblackgeneration —

proc
A POSTSCRIPT language procedure that can be called with a number in the range 0.0 to 1.0 (inclusive) on the operand stack and which returns a number in the same range. This procedure maps the minimum of the *user cyan*, *magenta*, and *yellow* color components to *user black* values. For additional information, see Section C.4.

Example

This sets the black component to zero when the minimum of cyan, magenta, and yellow is less than or equal to 0.75, while minima greater than 0.75 produce a black component that increases linearly from 0.0 (at a minimum of 0.75) to 1.0 (when user cyan, magenta, and yellow all have values of 1.0).

```
{dup 0.75 le {pop 0.0}{0.75 sub 4.0 mul} ifelse} setblackgeneration
```

Errors

stackunderflow
The operand stack is empty when the operator expects a value.

typecheck
An operand value is not the type that the operator expects.

setcmykcolor

Sets the current color parameter in the graphics state to a color described in the cyan-magenta-yellow-black color specification.

The use of **setcmykcolor** after a **setcachedevice** operation within the scope of a **BuildChar** procedure is not permitted (an **undefined** error results).

Format

cyan magenta yellow black **setcmykcolor** —

cyan
The value of the cyan color component. Must be in the range 0.0 to 1.0.

magenta
The value of the magenta color component. Must be in the range 0.0 to 1.0.

yellow
The value of the yellow color component. Must be in the range 0.0 to 1.0.

black
The value of the black color component. Must be in the range 0.0 to 1.0.

Description

setcmykcolor establishes the color used subsequently to paint shapes such as lines, areas, and characters on the current page. This operator bypasses the black generation and undercolor removal operations.

For additional information, see Section C.4.

Note that applying the **setgray** operator sets the gray color component to its single argument and the red, green, and blue color components to 1.0. It is the equivalent of the following use of **setcmykcolor**:

```
0.0 0.0 0.0 1.0 5 -1 roll sub setcmykcolor
```

Errors

rangecheck
An operand value is not in the allowed range of values.

stackunderflow
The operand stack is empty when the operator expects a value.

typecheck
An operand value is not the type that the operator expects.

setcolorscreen

Sets the current halftone screen definitions for red/cyan (r/c), green/magenta (g/m), blue/yellow (b/y), and gray/black (g/b) output color components in the graphics state.

Format

r/c-freq r/c-angle r/c-proc g/m-freq g/m-angle g/m-proc b/y-freq b/y-angle b/y-proc g/b-freq g/b-angle g/b-proc **setcolorscreen** —

r/c-freq r/c-angle r/c-proc
The frequency, angle, and procedure for the red/cyan screen.

g/m-freq g/m-angle g/m-proc
The frequency, angle, and procedure for the green/magenta screen.

b/y-freq b/y-angle b/y-proc
The frequency, angle, and procedure for the blue/yellow screen.

g/b-freq g/b-angle g/b-proc
The frequency, angle, and procedure for the gray/black screen.

Description

Each of the *r/c-freq*, *g/m-freq*, *b/y-freq*, and *g/b-freq* operands is a number that specifies the screen frequency for one output color component, measured in halftone cells per inch in device space. The *r/c-angle*, *g/m-angle*, *b/y-angle*, and *g/b-angle* operands specify the number of degrees by which their respective halftone screens are rotated with respect to the device coordinate system. The *r/c-proc*, *g/m-proc*, *b/y-proc*, and *g/b-proc* operands are each a POSTSCRIPT language procedure (as for **setscreen**). They define one color component's spot function to determine the order in which pixels within a halftone cell of that color component are assigned a light value to produce any desired shade of that color component. As with **setscreen**, the pixels with the highest values returned by this function become dark soonest at the lightest shades of that color, and the pixels with the lowest values become dark latest at the darkest shades of that color. The red/cyan, green/magenta, and blue/yellow screens have no

effect on a black-and-white device, the gray/black screen has no effect on an RGB or CMY device, and no screens have any effect on a full (8-bits-per-pixel) gray-scale device.

Many binary color printers (those that print each dot at full intensity or not at all) require different angles for each color-component halftone to have attractive output. Each color printer containing a POSTSCRIPT interpreter has a default color screen chosen to look good on that printer.

The **setcolorscreen** operator is the logical expansion of **setscreen** to the four color components. It takes the same three types of arguments as **setscreen**, but repeated four times. Applying the **setscreen** operator sets all four screens equally. It is the equivalent of the following use of **setcolorscreen**:

```
3 copy 6 copy setcolorscreen
```

Example

This sets the halftone screen definitions for red/cyan (r/c), green/magenta (g/m), blue/yellow (b/y), and gray/black (g/b) output color components for standard color printing.

```
% 50 line dot screen with 75 degree cyan,
% 15 degree magenta,
% 0 degree yellow,
% and 45 degree black angled screens,
% which are standard for color printing
/sfreq 50 def          % 50 halftone cells per inch
/sproc { dup mul exch dup mul add 1 exch sub} def
                       % dot-screen spot function
sfreq 75 /sproc load   % 75 degree red (cyan) screen
sfreq 15 /sproc load   % 15 degree green (magenta) screen
sfreq 0 /sproc load    % 0 degree blue (yellow) screen
sfreq 45 /sproc load   % 45 degree gray (black) screen
setcolorscreen
```

Errors

limitcheck
A POSTSCRIPT implementation limit has been exceeded.

rangecheck
An operand value is not in the allowed range of values.

setcolorscreen

stackunderflow
The operand stack is empty when the operator expects a value.

typecheck
An operand value is not the type that the operator expects.

setcolortransfer

Sets the current transfer function parameters for red, green, blue, and gray in the graphics state.

Format

redproc greenproc blueproc grayproc
setcolortransfer —

redproc
The transfer function for the red component.

greenproc
The transfer function for the green component.

blueproc
The transfer function for the blue component.

grayproc
The transfer function for the gray component.

Description

Each operand must be a POSTSCRIPT language procedure that may be called with a number in the range 0.0 to 1.0 (inclusive) on the operand stack and which will return a number in the same range. These procedures map user values of the color components (e.g., those specified by **setrgbcolor** and adjusted by **setblackgeneration** and **setundercolorremoval**, or 1.0 minus those specified by **setcmykcolor**) to *device* color components (for halftones, a weighted average of the lightness of pixels in the halftone cell). Only those transfer functions corresponding to color components supported by a device will have an effect on that device's output. For example, *redproc*, *greenproc*, and *blueproc* will have no effect on a black-and-white device, while *grayproc* will have no effect on an RGB device.

The single-color **settransfer** operator takes a single procedure argument whose purpose is to provide gamma correction for a printer's halftoning response. That operator is useful for a variety of effects beyond its original intention as a gray response correction function, but it is useful only in the context of a single output color, as on

black-and-white printers. The **setcolortransfer** operator is the logical expansion of **settransfer** to four color components; it takes four function arguments, each similar in purpose to the function argument of **settransfer**, but each function separately controls the response for each of the red (1.0 minus cyan), green (1.0 minus magenta), blue (1.0 minus yellow), and gray (1.0 minus black) components, respectively.

Applying the **settransfer** operator sets all four transfer functions equally. It is the equivalent of the following use of **setcolortransfer**:

```
dup dup dup setcolortransfer
```

The use of **setcolortransfer** after a **setcachedevice** operation within the scope of a **BuildChar** procedure is not permitted (an **undefined** error results).

Example

This sets device blue as the square of user blue and leaves the other color components unchanged.

```
{} {} {dup mul} {} setcolortransfer
```

Errors

stackunderflow
The operand stack is empty when the operator expects a value.

typecheck
An operand value is not the type that the operator expects.

setundercolorremoval

Sets the current undercolor removal function parameter in the graphics state.

The use of **setundercolorremoval** after a **setcachedevice** operation within the scope of a **BuildChar** procedure is not permitted (an **undefined** error results).

Format *proc* setundercolorremoval —

proc
A POSTSCRIPT language procedure that may be called with a number in the range 0.0 to 1.0 (inclusive) on the operand stack and which will return a number in the range –1.0 (to *increase* the color components) to +1.0 (to *decrease* the color components). This procedure maps the minimum of the *user cyan*, *magenta*, and *yellow* components to a value to be subtracted from each of these same components.

For additional information, see Section C.4.

Example

This sets the undercolor removal to be half of the black component from black generation.

```
{currentblackgeneration exec 0.5 mul} setundercolorremoval
```

Errors

stackunderflow
The operand stack is empty when the operator expects a value.

typecheck
An operand value is not the type that the operator expects.

Appendix D

Troubleshooting Procedures

The following troubleshooting procedures are provided to diagnose error conditions encountered when using the DECprint Utility software.

D.1 Your DECprint Utility Job Does Not Print

If your DECprint Utility job completes successfully and does not print, check to ensure that the POSTSCRIPT file contains a **showpage** command. Some applications that generate figures for inclusion into other documents do not include a **showpage** command at the end of the POSTSCRIPT code.

D.2 Your Job Reports UNDEF Error Messages

If your DECprint Utility job reports an UNDEF error, your file may not contain the right data type, may contain nonstandard undefined operators, or may contain control characters. The following sections describe what you can do to print your file successfully.

D.2.1 Unsupported File Extensions

If you receive the following error message, check to see if the file you submitted with the DECprint Utility command is a POSTSCRIPT file.

```
%PSPRINT-W-UNDEF, undefined: Name not known - offending command
is string
```

D.2.2 Unsupported PostScript Operators

The DECprint Utility does not define device-specific POSTSCRIPT operators that it does not use. If your file contains a nonstandard operator, **setjobtimeout**, for example, then your job does not print and you receive the following messages:

```
%PSPRINT-W-UNDEF, undefined: Name not known - offending command
is setjobtimeout

%PSPRINT-E-FLUSING, rest of job will be ignored
```

To print your file containing the undefined operator, do the following:

1. Create a file (fix.PS) containing this line:

   ```
   /setjobtimeout {pop} def
   ```

2. Name this file on the DECprint Utility command line before your file:

   ```
   $PSPRINT/QUEUE=queue-name fix.PS,myfile.ps
   ```

Other nonstandard operators that are not defined by the DECprint Utility can cause the same problem. Follow the same procedure, substituting the name of the operator named in the UNDEF error message in place of the **setjobtimeout** operator.

D.2.3 Problems With Files Containing CTRL/D

If processing a file with the DECprint Utility results in an UNDEF error message reporting that the offending command is a standard operator, such as **showpage**, or is left blank, then your file may contain a control character. When receiving this type of error message, convert your file with /MESSAGES=FILE and look at the .LOG file, which captures control characters. CTRL/D is the most likely control character you may encounter.

To successfully print files containing CTRL/D with the DECprint Utility, do one of the following:

- Edit the file (or change the application) and remove the control character.
- Create a POSTSCRIPT file (fix2.PS) containing the line:

  ```
  (/004) cvn cvlit {} def
  ```

Name this file on the DECprint Utility command line before your file:

```
$ PSPRINT/QUEUE=queue-name fix2.PS,myfile.PS
```

The DECprint Utility software then ignores CTRL/D in your file.

D.3 Troubleshooting Color Printing Problems

The following sections describe problems you may encounter when printing a file that has color information.

D.3.1 Your Color Output is Black

If the DECprint Utility prints your color file in black-and-white, check to see that your file requests color conversion. The file may actually request black and white. For more information, check the documentation for your color application.

D.3.2 Thin Lines Don't Print

Depending on the color selection, very thin lines appear not to print. You can either choose a darker color or increase the width of the line.

D.3.3 Incorrect Colors

The color of your file printed on the LJ250 printer is dramatically wrong, such as, green looks like light blue. This may be your first warning that one color of ink is out or simply that a nozzle is clogged. For information on how to unclog the nozzle or replace an ink cartridge, refer to the documentation for your printer.

Colors appear faint when the LA324 MultiPrinter color ribbon requires replacement. See the printer documentation for information on how to replace the ribbon.

To check for true colors, you can print the IVP:

- **DECprint Utility** prints with red letters
- **Successfully** prints in green

- **Installed** prints in blue

Section 2.5 explains how to run the IVP after installation.

D.4 Your Setup Modules Do Not Work

DECprint Utility software does not support /FORM and /SETUP. If your POSTSCRIPT file requires setup modules to work properly, you can do either of the following:

- Rewrite your files to be independent of setup modules, which increases their portability.
- Extract the required modules into a file and include the name of the file on the command line.

```
$ PSPRINT/QUEUE=queue-name setupfile.PS,myfile.PS
```

D.5 /PARAMETERS=(NUMBER_UP=2) Does Not Work

The DECprint Utility does not support the /PARAMETERS qualifier. Instead, use the DECprint Utility /LAYUP qualifier for printing multilple pages on a sheet of paper. For information on the /LAYUP qualifier, see Section 4.3.2.

D.6 Unexpected Page Orientation

If you submitted your /LAYUP job to the DECprint Utility with /ORIENTATION=*landscape* and your pages print in portrait mode, then the file is a portrait file. Resubmit the job with /LAYUP=*n*/ORIENTATION=*portrait*.

Conversely, if you request a portrait orientation and your pages print in landscape mode, submit your job again with /ORIENTATION=*landscape*.

DECprint Utility software supports the use of the /ORIENTATION qualifier only with the /LAYUP qualifier to determine the order in which the page spots print. This order differs according to the option you select, LANDSCAPE or PORTRAIT.

Page orientation for the POSTSCRIPT file that you want to print using /LAYUP on your sixel printer is already determined. Use the /ORIENTATION qualifier to declare this orientation. See Sections 4.3.2 and 4.3.5 for more information.

D.7 DECprint Utility Jobs Occupy Your Terminal for Too Long

To keep your terminal free while using the DECprint Utility, you can do one of the following:

- Submit your DECprint Utility job to a batch queue using the command file Batch-PSPRINT.com provided in SYS$COMMON:[SYSHLP.EXAMPLES.PSPRINT]. For more information, see Section 4.6.
- Run dual sessions on your VT330/VT340 terminal. See the documentation for your video terminal for information.

D.8 Faint Diagonal Lines With a Write-White Engine

Printed output of diagonal lines one pixel wide from a write-white engine, such as the LN03 printer, is faint. The only solution is to increase the width of the line.

D.9 The Position of the Image is Wrong

If the image is not in the right place on the sheet of paper, check to make sure that the paper is properly aligned in the printer. To get the proper top of form, refer to the documentation for your printer.

Glossary

activity license An LMF license type that defines the number of concurrent uses allowed for a product at any one time. An activity can be defined as an interactive user, a running process, or a job. For example, when you register four activity licenses for the DECprint Utility software, LMF authorizes four users to access the DECprint Utility software at the same time.

A media size The United States media size of 8.5 × 11.0 in. (or 216 × 279 mm). Also known as letter size.

A4 media size The European media size of 210 × 297 mm (or 8.26 × 11.69 in.)

baud rate The speed (bits/second) at which the computer and the printer communicate.

bitmap A digitized image that can be stored, transmitted, and reproduced.

block The smallest logically-addressable unit of data that a specified device can transfer in an I/O operation (512 contiguous bytes for most disk devices).

document In the context of a DECprint Utility command, a document is an entity that signals the resetting of the POSTSCRIPT interpreter. In the command line, plus signs concatenate files into documents and commas separate documents.

dot In the context of the sixel protocol, the smallest displayable unit — an ink dot on paper. Dots can be round, oval, square, rectangular, and small or large.

driver The set of code in the operating system that handles the physical I/O to a device.

execution queue The queue with which the symbiont is associated.

font The representation of a typeface that describes some set of characters rendered in a particular point size, weight, and style.

global qualifier A qualifier that applies to all files in a job.

host A network node that performs services and enables start up and management of peripheral devices, such as printers.

image area The printable part of a page. On most printers, you cannot print to the physical edge of the page.

interpreter A program in the printer or on the host that converts data for a particular imaging data syntax into a bitmap.

landscape printing A method of printing characters parallel to the long edge of the paper.

name A descriptive identifier (ASCII string) that is associated with a subject or object in the system.

network A group of computers that are connected to each other by communication lines to share information and resources.

node A network-addressable component having a unique address.

number_up The term for printing multiple logical pages on one physical sheet of paper. For example, printing 4-up means that you are printing four pages on a single sheet.

page In the context of a printer, a page is the image you are printing; you usually print one page per sheet.

persistent parameters System parameters that remain unchanged across POSTSCRIPT jobs. Therefore, these parameters persist after the host-based POSTSCRIPT interpreter is reset.

pixel In the context of the sixel protocol, the logical rectangular image area (smallest piece of an image) defined by each bit of sixel data, as intended by the creator software. An aspect ratio defines the size of a pixel. The printer prints pixels as dots.

pixel aspect ratio Shape of the pixel represented as a ratio of the vertical side to the horizontal side.

portrait printing A method of printing characters parallel to the short edge of the paper. This is normal page orientation for printing. For example, this page is printed in a portrait orientation.

positional qualifier A print qualifier that affects the command differently depending on its position in the command line.

POSTSCRIPT interpreter A stored program that converts POSTSCRIPT commands into page images.

POSTSCRIPT language A programming language designed to convey a description of virtually any desired page to the printer. It can describe a page containing any combination of text, graphical shapes, and digitized images.

print symbiont In the context of the DECprint Utility, the software that transfers the queued DECprint Utility jobs to the sixel printer.

printer controller mode Terminal control sequences direct output written to the terminal to the printer port (attached printer) rather than to the terminal screen.

privilege The level of system access allowed to a user.

protocol The conventions or rules for the format and timing of messages sent and received.

resolution The number of dots in a defined area.

resource An identifiable entity, physical or conceptual, that can perform a service and can be named and accessed on a network. The sharable entity can be either hardware or software.

sheet A sheet is the piece of paper that is printed.

sixel A group of six vertical pixels represented by 6 bits in a character code of 7 or 8 bits.

system parameters Values maintained within the POSTSCRIPT interpreter and used by the interpreter to control various system-specific features. *See also* **persistent parameters** and **volatile parameters**.

timeout The expiration of the time limit in which a device is to complete an I/O transfer.

translator A stored program that changes the user's data syntax into a form that can be used by the printer.

user The person who initiates requests for service. These requests are handled by the host, which forwards them to the appropriate service.

volatile parameters System parameters that stay in effect for one POSTSCRIPT print job. At the end of the DECprint Utility job, volatile parameters revert to their default values.

Index

A

/AFTER qualifier • 4–12
Attached printers • 4–3, 4–5
    effect of control character during printing • 4–4
/ATTACHED qualifier • 4–6

B

Black generation • C–4

C

CMYK color • C–6
Color extensions
    colorimage operator • C–14
    currentblackgeneration operator • C–16
    currentcmykcolor operator • C–17
    currentcolorscreen operator • C–18
    currentcolortransfer operator • C–19
    currentundercolorremoval operator • C–20
    format • C–13
    setblackgeneration operator • C–21
    setcmykcolor operator • C–22
    setcolorscreen operator • C–24
    setcolortransfer operator • C–27
    setundercolorremoval operator • C–29
colorimage operator • C–7, C–14
    examples for • C–9
Color images • C–7
Color screens • C–7
Color transfer functions • C–7
Command syntax • 4–2
componentids extension • B–4, B–5
Control characters
    effect with attached printers • 4–4
currentblackgeneration operator • C–16
currentcmykcolor operator • C–17
currentcolorscreen operator • C–18
currentcolortransfer operator • C–19
currentundercolorremoval operator • C–20
CYMK color • C–3

D

DECprint Utility
    system messages • 5–1
DECprint Utility command line • 4–2
    attached printers • 4–3
    queued printers • 4–2
/DELETE qualifier • 4–14
Device control library • A–3, A–4
/DEVICE qualifier • 4–15
Device support • 4–5

F

/FORM qualifier • 4–14

H

Hardware requirements • 1–4

I

Installation
    hardware requirements • 1–4
    order with LN03 Image software • 1–5

Installation (Cont.)
procedure messages • 5–1
required disk space • 1–5
sample session • 2–8
software requirements • 1–4
time required • 1–5
Installation procedure questions • 2–2 to 2–7
Installation Verification Procedure (IVP)
output • 2–9

J

jobname operator • B–3, B–5

L

LA210 Letterprinter • 1–3, 3–3
LA324 MultiPrinter • 1–3, 3–3
LA75 Companion printer • 1–3, 3–3
/LAYUP qualifier • 4–7
LJ250 Companion Color printer • 1–3, 3–3
using transparencies with • 4–18
LN03 Image software
installation order with DECprint Utility • 1–5
LN03 PLUS Desktop Laser printer • 1–3, 3–3
Logical name
for queued printing • 3–1, 4–5

M

/MEDIUM qualifier • 4–8
Messages
descriptions • 5–2 to 5–11
format • 5–1
severity level • 5–2
/MESSAGES qualifier • 4–9

O

/ORIENTATION qualifier • 4–10

P

pagecount operator • B–3, B–6
pagesize operator • B–3
/PARAMETERS qualifier • 4–13
Persistent parameters • B–3
Postinstallation requirements • 3–1
for queued printing • 3–1
for VAXcluster systems • 3–3
POSTSCRIPT extensions • B–1 to B–6
componentids • B–5
portability • B–1
sheetcount • B–6
POSTSCRIPT operators
begin • B–3
end • B–3
jobname • B–5
pagecount • B–6
pagesize • B–3
product • B–4
revision • B–5
version • B–5
where • B–2
Printed output
IVP • 2–9
Printer startup file
setting up print queues • A–2
Printer support • 1–3
Print qualifiers
different from PRINT command • 4–12
effective at queue time • 4–3
not supported by DECprint Utility command • 4–13
specific to DECprint Utility command • 4–6
Print queues, creating • A–1
/PROCESSOR qualifer • A–5
product operator • B–4
PSPRINT$queue_DEVICE_TYPE • 3–1, 4–5
PSPRINT$STARTUP
editing of • A–2 to A–4
queue setup topics • A–4 to A–6
renaming • A–2

Q

Qualifiers
different from PRINT command • 4–12
effective at queue time • 4–3
not supported by DECprint Utility command • 4–13
specific to DECprint Utility command • 4–6
Queued printing • 4–5
Queues, creating • A–1

R

Release Notes • 2–2
revision operator • B–4, B–5

S

Separation pages • A–5
setblackgeneration operator • C–21
setcmykcolor operator • C–22
setcolorscreen operator • C–24
setcolortransfer operator • C–27
setundercolorremoval operator • C–29
Setup modules
  with DECprint Utility jobs • 4–14
/SETUP qualifier • 4–14
sheetcount operator • B–3, B–6
Software requirements • 1–4
statusdict • B–3
Supported devices
  device type keywords for • 3–3
  resolution of • 3–3
System messages • 5–1 to 5–11
System parameters • B–3
System startup file • A–4

T

Terminal device type • A–5
Terminals supported by DECprint Utility software • 1–3
Tranparency mode
  LJ250 printer • 4–18
Troubleshooting procedures • D–1 to D–5

U

UNDEF error message • B–1
Undefined name
  statusdict • B–3
  using **where** operator • B–2
Undercolor removal • C–4

V

version operator • B–5
VMSINSTAL procedure
  conventions • 2–1
  installation questions • 2–2 to 2–7
  release notes • 2–2
Volatile parameters • B–3