VAX/VMS Management/User's Guide: ScriptPrinters

Order Number: AA–JF62C–TE

This manual describes how to use the ScriptPrinter software on VAX/VMS and MicroVMS operating systems to print files on a ScriptPrinter serial line laser printer. It includes the management of the print queues and the submission of print requests.

Revision/Update Information:	This manual supersedes *VAX/VMS Management/User's Guide: LN03R ScriptPrinter*, AA–JF62B–TE.
Operating System and Version:	VAX/VMS, Version 4.7 or later
Software Version:	VAX ScriptPrinter Software 2.0

digital equipment corporation
maynard, massachusetts

First Printing, July 1987
Revised, May 1988
Revised, November 1988

ML–S1084

This document was prepared using VAX DOCUMENT, Version 1.1.

Contents

PART II SCRIPTPRINTER USER'S GUIDE

TABLES

Preface

Intended Audience

The *VAX / VMS Management / User's Guide: ScriptPrinters* is for two kinds of VAX/VMS users: the system manager and the general user.

For system managers, this manual covers how to install and manage the print queues that are associated with ScriptPrinters. This information requires a knowledge of VAX/VMS device and generic queues.

For general users this manual covers how to submit VAX/VMS print requests to the printer. This information requires a knowledge of the DIGITAL Command Language (DCL) as implemented in VAX/VMS systems.

Users need to interpret system messages, which are listed and explained in Chapter 9.

Document Structure

The manual contains the following chapters and appendixes:

- Chapter 1 provides an overview of the ScriptPrinter and its environment.
- Chapter 2 describes the DCL commands used to install and manage the print queues on the system.
- Chapter 3 describes the VMS Accounting for the ScriptPrinter.

- Chapter 4 describes flag and trailer pages for files, flag and trailer pages for jobs, and job log pages.
- Chapter 5 describes the ScriptPrinter device control library. It includes information on how to write a device control module.
- Chapter 6 describes how to create and use layup definition files.
- Chapter 7 explains how to use forms and setup modules with ScriptPrinter queues.
- Chapter 8 describes how to submit print requests to the printer.
- Chapter 9 contains a list of system messages. If a message is the result of an error condition, appropriate recovery procedures are included.
- Appendix A provides guidelines for writing POSTSCRIPT programs that run successfully in the ScriptPrinter environment.
- Appendix B describes the differences in using the DCL PRINT command with the ScriptPrinter and the PrintServer 40.
- The Glossary contains definitions of terms associated with ScriptPrinters and printing.

Conventions

This document uses the following conventions:

Convention	Meaning
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. Italic words in normal text indicate the first use of a new term that is defined in the glossary.
[ ]	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 (|).

Convention	Meaning
...	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.
RET	This symbol indicates that you should press the RETURN key.

Associated Documents

The following documents are associated with ScriptPrinters:

- *VAX/VMS Software Installation Guide: ScriptPrinters*
- *VMS DCL Dictionary*
- *Guide to VAX/VMS System Management and Daily Operations*
- *VMS Librarian Utility Manual*
- *VAX/VMS System Manager's Reference Manual*
- *Guide to Maintaining a VMS System*
- *VMS Utilities Reference Volume*
- *PostScript Language Reference Manual*
- *PostScript Language Tutorial and Cookbook*
- *PostScript Language Program Design*
- *LN03R ScriptPrinter Programmer's Supplement*
- *PostScript Translators Reference Manual*
- *PostScript Quick Reference Guide: ScriptPrinters*
- *LN03R ScriptPrinter Operator Guide*
- *LN03R ScriptPrinter Installation Guide*

Other reference books you may find helpful if your ScriptPrinter is on a DECserver in a local area Ethernet or uses Distributed Queuing Service (DQS) software:

- *DECserver 100 Terminal Server Operations Guide*
- *DECserver 200 Management Guide*
- *DECserver 200 Software Installation Guide (VMS/MicroVMS)*
- *DECserver 500 Management Guide*
- *Terminal Server Commands and Messages Reference*
- *LAT/VMS Management Guide*
- *VAX Distributed Queuing Service User's Guide*
- *VAX Distributed Queuing Service Management Guide*

Summary of Technical Changes

ScriptPrinter Software, Version 2.0, supports the following new features:

- Accounting
- ANSI setup files
- Device control library search list
- Layup
- Improved small ANSI job performance
- New DATA_TYPE synonyms
  - ANSI2, LINE, and TEXT: same as ANSI
  - PS: same as POSTSCRIPT
- Shared printer access (multihost)

Chapter 1

ScriptPrinter Overview

The LN03R ScriptPrinter is a high-quality laser printer that operates at speeds of up to eight pages a minute and supports Adobe's POSTSCRIPT® page description language. It can generate printed pages that integrate both graphics and text in a variety of fonts.

Installation of the ScriptPrinter software on the host is described in the *VAX/VMS Software Installation Guide: ScriptPrinters*. For more information about the unit's hardware features, see the *LN03R ScriptPrinter Operator Guide*. This chapter provides a brief description of the LN03R ScriptPrinter user panels and its software components.

1.1 LN03R ScriptPrinter User Panels

One button and a set of indicators are located on the front panel, and two switches are located on the rear panel. The button and indicators show you the status of the print engine and enable you to control its operation. The rear panel switches control paper size and the summary sheet. This section provides a summary description of the switches and buttons. For more information, see the *LN03R ScriptPrinter Operator Guide*.

1.1.1 On-Line/Off-Line Button

The on-line/off-line button on the front panel toggles the LN03R ScriptPrinter between on-line and off-line status. When the printer is on line, it can receive signals from the data communications line so it can print. When off line, it cannot receive the signals.

1.1.2 Graphic Indicators

The nine graphic indicators on the front panel indicate the following conditions: printer ready, printer communicating, power on, paper jam, service call necessary, add paper, add toner, full toner collection bottle, user maintenance necessary. Further information is supplied by the single-digit display.

1.1.3 Single-Digit Display

The single-digit display indicates status of the print engine. No number is displayed when print engine is in a ready state. When the graphic indicators are lit, a code indicates the problem. Flashing codes indicate an open cover and/or paper throughput errors. Nonflashing codes indicate problems with fusing, the optical system, the main motor, and OPC synchronous mark errors.

NOTE

A nonflashing "6" and a flashing "controller active" symbol (lightning bolt) indicates a PostScript program is running.

1.1.4 Page Count Indicator

A 6-digit counter on the right side of the printer displays the number of pages printed since the printer was built.

1.1.5 Paper Size

A switch on the rear panel indicates to the LN03R ScriptPrinter firmware the size of the paper in the input tray. The paper size switch setting should agree with the size of the paper in the input tray. To determine the setting of the paper size switch, generate a summary sheet.

Power off the printer before resetting the paper size switch. New settings of this switch have effect only at power-up time.

1.1.6 Summary Sheet Button

A white button, with the T in a circle, on the rear panel is used to generate a summary sheet.

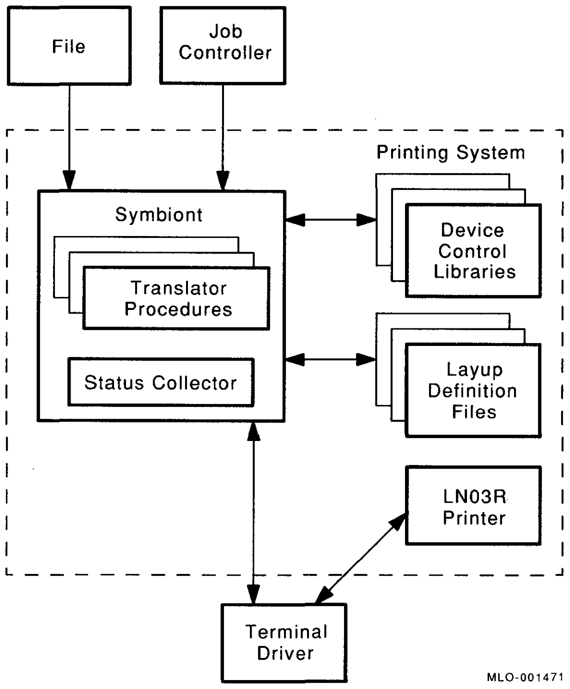
1.2 ScriptPrinter Software

The VAX ScriptPrinter software operates with a VMS or MicroVMS operating system and requires the job controller (JBC) for queue management to be running. The software components supplied with the VAX ScriptPrinter software are used in place of the PRTSMB symbiont supplied with the VMS operating system. The VAX ScriptPrinter software accepts user requests to print files. The software is distributed in the form of executable image files, text and help library files, and text files.

Chapter 8 describes the print symbiont user interface. Chapter 5 describes the device control library. Information on font metrics files can be found in the *LN03R ScriptPrinter Programmer's Supplement*. Data protocol translators are described in the *PostScript Translators Reference Manual.*

Figure 1–1 shows the system software structure.

Figure 1–1: Major Software Systems



1.2.1 System Files

The following system files are included in the software kit:

- Print symbiont
- Symbiont message
- Device control library
- Help library
- Command procedures to build startup and shutdown command files and run the installation verification procedure (IVP)
- Translators (3)
- Font metric library
- Layup definition
- Release notes
- Set POSTSCRIPT job timeout program
- Set printername POSTSCRIPT program

1.2.2 Software Services

The ScriptPrinter software provides the following services:

- Print symbiont software that processes print requests from VMS print queues assigned to the ScriptPrinter. Chapter 8 describes the print symbiont user interface.
- ScriptPrinter queue management services, as described in Chapter 2.
- Translators that convert data syntaxes into POSTSCRIPT. The data syntaxes supported are ANSI/SIXEL, ReGIS, and Tektronix 4010/4014. The translators are documented in the *PostScript Translators Reference Manual.*

The printer controller firmware of the ScriptPrinter includes an interpreter for Adobe's POSTSCRIPT page description language. The interpreter executes POSTSCRIPT commands from the data sent to it, to generate an image, which is printed by the print engine.

1.2.3 Application Programs

Application programs running on host systems produce files that can be printed on the LN03R. Types of applications are:

- Text formatters
- Computer aided design tools
- Spreadsheets
- Graphics editors

The output of these applications can be POSTSCRIPT, ReGIS, Tektronix 4010/4014, or ANSI. POSTSCRIPT is the preferred output, as it does not require translation.

1.3 VMS Operating System Support

The ScriptPrinter software depends on the following components of the VMS system for support:

- Command language: The ScriptPrinter software uses the DIGITAL Command Language (DCL) interpreter to parse commands and transfer them to the symbiont by way of the job controller.
- Terminal driver: The ScriptPrinter print symbiont communicates with the printer by means of the QIO interface to the VMS terminal driver or to the LAT driver.

Chapter 2

Managing Print Queues

The commands for managing ScriptPrinter print queues are a subset of the DCL commands that control print and batch queues on VMS systems. The procedures are specific to ScriptPrinters and serve the local printing environment.

This chapter describes commands used to set up and manage the ScriptPrinter print queues. Additional information about these commands can be found in the *Guide to VAX/VMS System Management and Daily Operations*. Reference information for these commands can be found in the *VMS DCL Dictionary*. This chapter assumes that you are familiar with the information in these reference books.

The queue-management commands are discussed under the following categories:

- Initialize queue operations
- Assign and deassign operations
- Define and delete operations
- Start and stop operations
- Set and show operations

2.1 Access Privileges

In order to use the queue management commands, you may need one of the following:

- Operator privileges
- Write or execute access to the queues
- Delete access to queue entries

The access privileges for commands are documented in the *VMS DCL Dictionary* and are included in the command descriptions for your convenience.

2.2 INITIALIZE/QUEUE Operations

The ScriptPrinter host software installation procedure includes establishing at least one print queue. You should set up a print execution queue for each ScriptPrinter attached to the host system. You should consider VAXclusters as a single system. The commands to set up ScriptPrinter print queues are in SYS$MANAGER:CPS$STARTUP.COM on VMS version 4.x systems, or in SYS$STARTUP:CPS$STARTUP.COM on VMS version 5.x systems. The qualifiers used with INITIALIZE/QUEUE for setting up ScriptPrinter queues are the following:

- /START
- /ON
- /LIBRARY
- /SEPARATE
- /SCHEDULE
- /DEFAULT
- /PROCESSOR

During the installation procedure, an execution print queue is established as follows:

```
$ INITIALIZE/QUEUE-
            /START-
            /ON=device-name-
            /LIBRARY=CPS$DEVCTL-
            /SEPARATE=(NORESET,BURST,TRAILER)-
            /SCHEDULE=NOSIZE-
            /DEFAULT=(FORMS=CPS$DEFAULT,NOFEED)-
            /FORMS=CPS$DEFAULT-
            /PROCESSOR=CPS$SMB-
            device-queue-name
```

where:

device-name	is the physical device to which the ScriptPrinter is connected.
device-queue-name	is the queue name.

You can change the characteristics of ScriptPrinter print queues, but you should not change the above parameters and conventions.

During the installation procedure, a generic print queue is established with the following commands:

```
$ INITIALIZE/QUEUE-
            /GENERIC=execution-queue-name-
            /START-
            /SCHEDULE=NOSIZE-
            generic-queue-name
```

where:

execution-queue-name	is the name of an execution queue into which the generic queue can place work.
generic-queue-name	is the name of the generic queue that can move jobs to the named execution queue.

The installation procedure creates print queues with the following qualifiers:

- All DATA TYPEs:
  /SEPARATE=(NORESET,BURST,TRAILER)
  /DEFAULT=(FORM=CPS$DEFAULT,NOFEED)
- ReGIS, POSTSCRIPT, and TEK4014 DATA TYPEs only:
  /SCHEDULE=NOSIZE
- ANSI DATA TYPE only:
  /SCHEDULE=SIZE

2.2.1 Device Control Library

The /LIBRARY qualifier specifies the name of the device control library supplied as part of the ScriptPrinter host software. The library is located in SYS$LIBRARY and has a file extension of TLB.

The library contains the following modules:

- LPS$$xxxxx — POSTSCRIPT printer control modules for DIGITAL use only. You must not delete or modify any of these modules.

- LPS$xxxxx — DIGITAL-supplied POSTSCRIPT printer control modules that are available for your use.

NOTE

Customers should not use the LPS$$ or LPS$ prefix in the names of their own device control modules. Reinstallation creates a new version of the device control library. (It does **not** delete the old version.) DIGITAL recommends that you create your own device control libraries for your customized modules. By doing this, your custom modules remain independent of future software installations. For information on creating and modifying libraries, refer to Section 7.1.3 in this manual and the *VMS Librarian Utility Manual*. Refer to Section 7.1.4 in this manual for how to set up a library search (order) list.

If the library or the LPS$$xxxxx modules are not available and are required for your print job, queue operations fail with one of the following error messages:

```
CPS-F-NODEVCTLLIB, No device control library specified for the queue

CPS-F-NOLIBR, Setup library library-name not found

CPS-F-SETUPNOTFOUND, Setup module module-name not found

CPS-F-SETUPREADERR, Setup module module-name read error
```

2.2.2 Job Separation Defaults

The /SEPARATE qualifier specifies job separation defaults for the printer. ScriptPrinter defaults, defined during installation of the host software, include printing the job burst and trailer pages. For more information, refer to Chapter 4.

2.2.3 Print Symbiont

The /PROCESSOR qualifier specifies the file name of the print symbiont image used to process a print request for the printer. The symbiont must be located in SYS$SYSTEM and must have a file extension of EXE.

2.2.4 Initialization Examples

The following INITIALIZE/QUEUE commands define queues for two LN03R printers with device names TTB4: and TTB7:

```
$ INITIALIZE/QUEUE-
            /START-
            /ON=TTB4:-
            /LIBRARY=CPS$DEVCTL-
            /SEPARATE=(FLAG,TRAILER)-
            /SCHEDULE=NOSIZE-
            /DEFAULT=(CPS$DEFAULT,NOFEED)-
            /PROCESSOR=CPS$SMB-
            LN03R$TTB4

$ INITIALIZE/QUEUE-
            /START-
            /ON=TTB7:-
            /LIBRARY=CPS$DEVCTL-
            /SEPARATE=(BURST,TRAILER)-
            /SCHEDULE=NOSIZE-
            /DEFAULT=(CPS$DEFAULT,NOFEED)-
            /PROCESSOR=CPS$SMB-
            LN03R$TTB7
```

During the installation, you can set up generic queues in the CPS$STARTUP.COM file to associate default data types with a print queue. In the following example, two generic queues are assigned to the execution queue for printer TTB4. Systemwide logical names based on the names of the generic queues are then defined to set up specific data types.

```
$ INITIALIZE/QUEUE-
            /START-
            /GENERIC=(LN03R$TTB4)-
            /SCHEDULE=SIZE-
            ANSI_TTB4

$ DEFINE/SYSTEM LPS$ANSI_TTB4_PARAMETER "DATA_TYPE=ANSI"

$ INITIALIZE/QUEUE-
            /START-
            /GENERIC=(LN03R$TTB4)-
            /SCHEDULE=NOSIZE-
            POST_TTB4

$ DEFINE/SYSTEM LPS$POST_TTB4_PARAMETER "DATA_TYPE=POSTSCRIPT"
```

A generic print queue can point to multiple execution queues. When a single generic print queue points to both a ScriptPrinter execution queue and a PrintServer execution queue, the default PARAMETER logical name associated with that generic queue (LPS$*queue-name_*PARAMETER) applies to print jobs executed on both devices. In previous versions of the software, the generic queue required two PARAMETER logical names, one for the ScriptPrinter and one for the PrintServer.

Additional information on generic queue logical names is in Section 2.7.1 to Section 2.7.3. See Chapter 8 for information on parameter defaults.

2.2.5 Queue Initialization in a Cluster

If the host installation is performed on a member of a cluster, the START parameter is replaced by a START_QUEUE symbol. The symbol START_QUEUE is defined during the execution of the CPS$STARTUP.COM procedure. If the system executing the procedure is the host system, the symbol is defined as START and the queue is initialized and started. If the system executing the procedure is not the host system, the symbol is defined as NOSTART and the queue is initialized but not started.

2.3 ASSIGN/DEASSIGN Commands

The ASSIGN and DEASSIGN commands for managing print queues include the following:

- ASSIGN/QUEUE
- DEASSIGN/QUEUE
- ASSIGN/MERGE

ASSIGN/QUEUE

Redirects jobs on a logical queue to an execution queue. At the time this command is issued, the logical queue must not be running. The logical queue must be initialized (INITIALIZE/QUEUE) before assigning it to an execution queue.

To move jobs from a logical queue to the execution queue, you must issue the START/QUEUE command to the logical queue.

Format

ASSIGN/QUEUE *execution-qname[:] logical-qname[:]*

execution-qname
The name of the execution queue to be associated with the logical queue. The execution queue cannot be a generic queue.

logical-qname
The name of the logical queue to be associated with the execution queue.

Related Commands

DEASSIGN/QUEUE
INITIALIZE/QUEUE
START/QUEUE
SHOW QUEUE

Restrictions

The ASSIGN/QUEUE command requires OPER (operator) privileges or E (execute) access to both of the specified queues.

DEASSIGN/QUEUE

Disassociates a logical queue from an execution queue. This command also stops the logical queue. Jobs can still be sent to the logical queue. However, new jobs, as well as any unprocessed jobs in the queue, are placed in a pending state. They remain pending until the logical queue is assigned to an execution queue (ASSIGN/QUEUE) and started (START/QUEUE).

Format

DEASSIGN/QUEUE *logical-qname[:]*

logical-qname
The name of the queue that is to be stopped and disassociated from an execution queue.

Related Commands

ASSIGN/QUEUE
SHOW QUEUE
START/QUEUE

Restrictions

The DEASSIGN/QUEUE command requires OPER (operator) privileges or E (execute) access to the queue.

ASSIGN/MERGE

Transfers pending jobs on one queue to another queue. Stop the source queue before performing the merge operation. The jobs executing on either queue are not affected.

Format

ASSIGN/MERGE *target-qname[:] source-qname[:]*

target-qname
The name of the queue to which the jobs will be transferred.

source-qname
The name of the queue from which the jobs will be transferred.

Related Commands

SHOW QUEUE
STOP/QUEUE/NEXT
STOP/QUEUE/REQUEUE

Restrictions

The ASSIGN/MERGE command requires OPER (operator) privileges or E (execute) access to both queues.

Comments

This command is useful in the event of a printer malfunction. You can reroute the jobs from the failing device to an operating device. To perform this transfer, use STOP/QUEUE/NEXT and STOP/QUEUE/REQUEUE. For example:

```
$ STOP/QUEUE/NEXT PRINTER$ABLE
$ STOP/QUEUE/REQUEUE=PRINTER$BAKER PRINTER$ABLE
$ ASSIGN/MERGE PRINTER$BAKER PRINTER$ABLE
```

The first command stops the PRINTER$ABLE queue from starting any new jobs. The next command requeues the current job on the PRINTER$ABLE queue to the PRINTER$BAKER queue. The last command requeues any pending jobs on the PRINTER$ABLE queue to the PRINTER$BAKER queue.

2.4 DEFINE/DELETE Commands

The define and delete operations for managing print queues include the following commands:

- DEFINE/CHARACTERISTIC
- DEFINE/FORM
- DELETE/CHARACTERISTIC
- DELETE/ENTRY
- DELETE/FORM
- DELETE/QUEUE

DEFINE/CHARACTERISTIC

Assigns a name and number to print-queue characteristics, such as location or speed. These characteristics can then be assigned to queues with the INITIALIZE/QUEUE or SET QUEUE commands. When the characteristics are specified in PRINT commands, you have control over which queues service your print requests.

Format

DEFINE/CHARACTERISTIC *char-name char-number*

char-name
A descriptive name assigned to the characteristic being defined. It can be 1 to 31 characters long and must contain at least one nonnumeric character.

char-number
The corresponding number to be assigned to the characteristic. Numbers can range from 0 to 127.

Related Commands

SHOW QUEUE/CHARACTERISTICS
SET QUEUE/CHARACTERISTICS
DELETE/CHARACTERISTICS

Restrictions

The DEFINE/CHARACTERISTIC command requires OPER (operator) privileges.

When you assign printer characteristics to a queue, a job can execute on that queue only if each characteristic that you explicitly specify for the job is also specified for the queue.

DEFINE/FORM

Assigns a name and number to a form to be used on the ScriptPrinter. For more information on using forms with the ScriptPrinter see Chapter 7.

Format

DEFINE/FORM *[/DESCRIPTION=string][/SETUP=(module-name,...)] form-name form-number*

/DESCRIPTION=string
1 to 255 characters describing the *form* being defined. The qualifier is optional and the default string is *form-name*. The string is displayed in response to the SHOW QUEUE/FORM command.

/SETUP=module-name
Specifies one or more modules that set up the device for the designated form. For ANSI jobs, the modules can be POSTSCRIPT or ANSI programs. All other print jobs can only use forms that include POSTSCRIPT setup modules. Setup modules must be located in SYS$LIBRARY:CPS$DEVCTL.TLB or in one of the libraries set up in a search list. If you want to use ANSI setup modules, the modules must be in an ANSI library which is named in your library search list. (Refer to Section 7.1.4 for information on setting up a library search list.) The default is NOSETUP.

form-name
A 1- to 31-character name assigned to the form being defined. The name must contain at least one nonnumeric character. A default *form-name* of DEFAULT is assigned when the system is started.

form-number
A number from 0 to 9999 assigned to correspond to the *form-name*. The DEFAULT form is assigned number 0.

DEFINE/FORM

Related Commands

SET QUEUE/FORM
SHOW QUEUE/FORM
DELETE/FORM

Restrictions

The DEFINE/FORM command requires OPER (operator) privileges.

The /STOCK qualifier to the DEFINE/FORM command can have any value, transparency or pink paper, for example. However, the STOCK qualifier for the print job and the queue must match for the job to print. Only use a different value for /STOCK when your form requires a different type of paper or media. If your job requires special stock, define a form, which does not use the default stock. Then, your print job waits in the queue until the stock defined for the queue is set to match the stock defined in your form.

DELETE/CHARACTERISTIC

Removes a characteristic from the system characteristic table. You must do a DELETE/CHARACTERISTIC followed by a DEFINE/CHARACTERISTIC to change the name of a characteristic. To change the number associated with a characteristic, use the DEFINE/CHARACTERISTIC command with the same name and a different number.

Format

DELETE/CHARACTERISTIC *char-name*

char-name
The name of the characteristic to be deleted. It must have been defined with the DEFINE/CHARACTERISTIC command.

Related Commands

DEFINE CHARACTERISTICS
SHOW QUEUE/CHARACTERISTICS

Restrictions

The DELETE/CHARACTERISTIC command requires OPER (operator) privileges.

DELETE/ENTRY

Removes one or more job entries from a queue. You can also stop a job that is printing with the command STOP/QUEUE/ABORT.

Format

DELETE/ENTRY *=(entry-number[,...]) queue-name[:]*[1]

entry-number[,...]
Specifies the entry numbers of one or more jobs to be deleted from a queue. If only one job is being deleted, you can omit the parentheses.

queue-name[:]
Specifies the name of the queue where the jobs are located.

Related Commands

SHOW QUEUE
STOP/QUEUE/ABORT

Restrictions

The DELETE/ENTRY command requires OPER (operator) privileges, E (execute) access to the specified queue, or D (delete) access to the specified job.

You do not need these privileges to delete your own print jobs from the print queue.

[1] In VMS, Version 5.0, *queue-name* is optional.

Comments

If the job is printing, it may take several seconds to stop. The ScriptPrinter prints data in its buffers, plus any required job trailer and/or job log page. During this time, a SHOW QUEUE command displays the job's status as ABORTING.

If you delete the job before the job burst or job flag page prints, you may not get the job burst or job flag page. The job trailer and job log pages print if you requested that they print.

Sometimes, when the serial communication cable between the printer and the host is disconnected and later reconnected, the active print job displays the status PRINTING when nothing is happening at the printer (no flashing "6" display). Use the DELETE/ENTRY command to clear the job from the print queue so that printing will resume. Then resubmit the lost print job.

NOTE

When reconnecting the LN03R ScriptPrinter to the host, turn off the printer. Turn on the printer after the connection to the host has been made. Turning the printer off and on insures proper synchronization between the print symbiont and the ScriptPrinter.

DELETE/FORM

Deletes a forms definition from the system forms table. Before using DELETE/FORM, execute the SHOW QUEUE/FULL/ALL command to be certain no references to the form exist, either as an attribute of an active print queue, or as a qualifier to a print request. If any references exist, delete them. If you try to delete a form with outstanding references, you are notified of the condition and the command is not executed.

Format

DELETE/FORM *form-name*

form-name
The name assigned to the form by a DEFINE/FORM command.

Related Commands

DEFINE/FORM

Restrictions

The DELETE/FORM command requires OPER (operator) privileges.

DELETE/QUEUE

Deletes a queue. First, execute the SHOW QUEUE/FULL command to make sure no generic queues reference the queue you are trying to delete. If any references exist, delete them. If you try to delete a queue with outstanding references, you are notified of the condition and the command is not executed.

Stop the queue with the STOP/QUEUE/NEXT command. When activity has ceased, issue the DELETE/QUEUE command. Any pending jobs on the queue are also deleted.

Format

DELETE/QUEUE *queue-name[:]*

queue-name
The name of the queue being deleted.

Related Commands

ASSIGN/QUEUE
INITIALIZE/QUEUE
SET QUEUE
START/QUEUE
STOP/QUEUE/NEXT

Restrictions

The DELETE/QUEUE command requires OPER (operator) privileges.

2.5 START/STOP QUEUE Commands

The start and stop operations used to manage print queues include the following commands:

- START/QUEUE
- START/QUEUE/MANAGER
- STOP/QUEUE
- STOP/QUEUE/ABORT
- STOP/QUEUE/MANAGER
- STOP/QUEUE/NEXT
- STOP/QUEUE/REQUEUE
- STOP/QUEUE/RESET

START/QUEUE

Starts a queue that has been initialized without the start qualifier or restarts a queue that has been stopped.

The qualifiers used with the INITIALIZE command can also be used with this command, except for qualifiers that position a file on the restart of a paused queue (ALIGN, BACKWARD, FORWARD, SEARCH, START, and TOP_OF_FILE). With each of these qualifiers, the printer resumes printing the job at the point where it stopped when the queue was paused.

Format

START/QUEUE *queue-name[:]*

queue-name
The name of the queue to be started or restarted.

Related Commands

SET QUEUE
STOP/QUEUE
STOP/QUEUE/NEXT
STOP/QUEUE/RESET

Restrictions

The START/QUEUE command requires OPER (operator) privileges or E (execute) access to the specified queue.

START/QUEUE/MANAGER

Starts the system job queue manager that controls the scheduling of job requests. The command is generally included in the system startup procedure. The queue manager must be started before any queue-management or job-submission operations. You need to issue this command only if you are setting up queues for the first time or a new queue file has to be created. Refer to the *VMS DCL Dictionary* and the *Guide to VAX/VMS System Management and Daily Operations* for more information on this command.[1]

Format

START/QUEUE/MANAGER *[file-spec]*

file-spec
Specifies the file to contain the queue-management information. The default file specification is SYS$SYSTEM:JBCSYSQUE.DAT.

Related Command

STOP/QUEUE/MANAGER

Restrictions

The START/QUEUE/MANAGER command requires both OPER (operator) and SYSNAM (system name) privileges.

If the printer is part of a cluster, the queue file should be created and maintained in the system common area, SYS$COMMON:[SYSEXE].

[1] For additional information refer to the *VAX/VMS System Manager's Reference Manual* or the *Guide to Maintaining a VMS System.*

STOP/QUEUE

Pauses an execution queue, suspends the current job, and suspends processing of new jobs entered into the queue.

Format

STOP/QUEUE *queue-name[:]*

queue-name
The name of the queue you want to pause.

Related Commands

START/QUEUE
STOP/QUEUE/ABORT
STOP/QUEUE/MANAGER
STOP/QUEUE/NEXT
STOP/QUEUE/REQUEUE
STOP/QUEUE/RESET

Restrictions

The STOP/QUEUE command requires OPER (operator) privilege or E (execute) access to the specified queue.

If you have not reset the timeout value in LPS$SET_TIMEOUT.PS to 0, then STOP/QUEUE may cause the printer to timeout. This may result in the loss of the current job.

STOP/QUEUE/ABORT

Aborts a job executing in the print queue, deletes the job entry from the queue, and continues processing other jobs in the queue.

Format

STOP/QUEUE/ABORT *queue-name[:]*

queue-name
The name of the queue that contains the job you want to stop.

Related Command

STOP/QUEUE/REQUEUE

Restrictions

The STOP/QUEUE/ABORT command requires OPER (operator) privilege, E (execute) access to the specified queue, or D (delete) access to the job.

Comments

A job that is printing may take several seconds to stop. The ScriptPrinter prints data left in its buffers, plus a job trailer and/or job log page. During this time, a SHOW QUEUE command displays the job's status as ABORTING.

If you abort the job before the job burst or job flag page prints, you may not get the job burst or job flag page. The job trailer and job log pages print if you requested that they print.

STOP/QUEUE/MANAGER

Stops output-execution queues on the host after executing jobs have completed. The STOP/QUEUE/MANAGER command is generally included in the system shutdown command procedure. Pending jobs remain pending. When activity in the queues ceases, the job queue manager file is closed. Refer to the *VMS DCL Dictionary* and the *Guide to VAX/VMS System Management and Daily Operations* for more information.[1]

Format **STOP/QUEUE/MANAGER**

Related Command

START/QUEUE/MANAGER

Restrictions

The STOP/QUEUE/MANAGER command requires both OPER (operator) and SYSNAM (system name) privileges.

[1] For additional information refer to the *VAX/VMS System Manager's Reference Manual* or the *Guide to Maintaining a VMS System.*

STOP/QUEUE/NEXT

Stops a queue on your system after the current job has completed. To restart the queue and resume printing, use the START/QUEUE command.

Format **STOP/QUEUE/NEXT** *queue-name[:]*

queue-name
The name of the queue you want to stop.

Related Commands

STOP/QUEUE
START/QUEUE

Restrictions

The STOP/QUEUE/NEXT command requires OPER (operator) privilege or E (execute) access to the specified queue.

Comments

The STOP/QUEUE/NEXT command is in contrast to the STOP/QUEUE/RESET command, which immediately stops the current job. Both commands prevent printing of new print jobs.

STOP/QUEUE/REQUEUE

Stops an executing job and requeues the job to the same or another queue. The queue is not stopped and processing continues for other jobs in the queue. If you specify a queue name as a parameter to the REQUEUE qualifier, the job is requeued to the specified queue.

If you also specify the HOLD qualifier, the job is placed in a hold state in addition to being requeued. The job can be released later with the SET QUEUE/ENTRY/RELEASE or SET QUEUE/ENTRY/NOHOLD command. The job can also be removed from the queue with the DELETE/ENTRY command.

Format

STOP/QUEUE/REQUEUE *[=new-qname]*
queue-name[:]

new-qname
The name of a queue to which the stopped job will be requeued. If this parameter is not present, the job is requeued on the same queue.

queue-name
The name of the queue containing the job to be stopped and requeued.

Related Command

STOP/QUEUE/ABORT

Restrictions

The STOP/QUEUE/REQUEUE command requires OPER (operator) privilege, E (execute) access to the queue, or D (delete) access to the job.

Comments

If the job is printing, it may take several seconds to stop it. The ScriptPrinter prints data left in its buffers, plus a job trailer and/or job log page. During this time, a SHOW QUEUE command displays the job's status as ABORTING.

STOP/QUEUE/RESET

Stops a queue. If a job is printing when you issue this command, the printer restarts this job at the beginning of the file when you restore the queue.

To restart the queue, use the START/QUEUE command.

Format

STOP/QUEUE/RESET *queue-name[:]*

queue-name
The name of the queue you want to reset.

Related Command

START/QUEUE

Restrictions

The STOP/QUEUE/RESET command requires OPER (operator) privilege or E (execute) access to the specified queue.

Comments

If the job is printing, it may take several seconds to stop it. The ScriptPrinter prints data left in its buffers, plus a job trailer and/or job log page. During this time, a SHOW QUEUE command displays the job's status as ABORTING.

2.6 SET/SHOW QUEUE Commands

The set and show operations used to manage print queues include the following commands:

- SET ENTRY
- SET QUEUE
- SET QUEUE/ENTRY
- SHOW QUEUE
- SHOW QUEUE/CHARACTERISTICS
- SHOW QUEUE/FORM

2.6.1 /RETAIN Qualifier

Distributed Queuing Services (DQS) software adds the /RETAIN=ALL qualifier to a SET/QUEUE command. Then, the ScriptPrinter Software retains print jobs upon completion and DQS returns log files upon request. Once DQS returns the log file, the log file is deleted from SYS$LOGIN:[DQS$SERVER].

SET ENTRY[1]

Changes or deletes attributes of a print job that is in a queue but not executing. The print job is identified by a job entry number assigned when the print request was issued. To recall a job entry number, use the SHOW QUEUE command.

You can use qualifiers specified under the SET ENTRY command in the *VMS DCL Dictionary*. If you are changing an attribute for a multifile job, the change affects all files for that job.

Format

SET ENTRY *=entry-number [/qualifiers]*

entry-number
The entry number of the job whose attributes you want to change.

Related Commands

SHOW QUEUE
PRINT

Restrictions

The SET ENTRY command requires OPER (operator) privilege or E (execute) access to the queue. You may also change the attributes of any job for which you have D (delete) access.

[1] VMS Version 5.0 or later supports the SET ENTRY command.

SET QUEUE

Changes the attributes of a queue. You can use qualifiers specified under the SET QUEUE command in the *VMS DCL Dictionary*. However, the qualifiers used must be compatible with the attributes of ScriptPrinter print queues. Attributes for ScriptPrinter print queues are documented in Section 2.2.

If an attribute to be changed requires the queue to be stopped, use the STOP/QUEUE/NEXT command to stop the queue after the printing job has completed. Make your change with the SET QUEUE command and restart the queue with the START/QUEUE command.

Format

SET QUEUE *queue-name[:]*

queue-name
The name of the execution or generic queue whose attributes you want to modify.

Related Commands

INITIALIZE/QUEUE
STOP/QUEUE/NEXT
START/QUEUE
SHOW QUEUE

Restrictions

The SET QUEUE command requires OPER (operator) privilege or E (execute) access to the queue. Changes to the OWNER_UIC and PROTECTION qualifiers require OPER privilege.

SET QUEUE/ENTRY

Changes or deletes attributes of a print job that is in a queue but not executing. The print job is identified by a job entry number assigned when the print request was issued. To recall a job entry number, use the SHOW QUEUE command.

You can use qualifiers specified under the SET QUEUE/ENTRY command in the *VMS DCL Dictionary.* If you are changing an attribute for a multifile job, the change affects all files for that job.

You can release a print job on hold with the SET QUEUE/ENTRY or SET ENTRY command, as follows:

```
$ SET QUEUE/ENTRY=nnnn/RELEASE queue_name
```

where:

nnnn	is the print job queue entry number.
queue_name	is the name of the queue.

Alternatively, you can remove a print job from the queue with the following DCL command:

```
$ DELETE/ENTRY=nnnn queue_name
```

Format

SET QUEUE/ENTRY *=entry-number [/qualifiers][queue-name[:]]*

entry-number
The entry number of the job whose attributes you want to change.

queue-name
The name of the queue in which the specified job is entered. If you do not specify a queue name, the system assumes the default queue SYS$PRINT.

Related Commands

SHOW QUEUE
PRINT

Restrictions

The SET QUEUE/ENTRY command requires OPER (operator) privilege or E (execute) access to the queue. You may also change the attributes of any job for which you have D (delete) access.

SHOW QUEUE

Displays information about the initialized queues and about the jobs in those queues. If you do not specify a queue name, the system displays information about all queues. Information about jobs that are running or pending will also be displayed.

Format

SHOW QUEUE *[queue-name[:]][/ALL][/FULL]*

queue-name
The name of the queue. The default is to display information about all queues.

ALL
Displays information about all jobs in the queue.

FULL
Displays information about the queue and any job in the queue owned by the process.

Related Commands

SHOW QUEUE/CHARACTERISTICS
SHOW QUEUE/FORM

Comments

Occasionally you may encounter a print queue status of DEVICE UNAVAILABLE. This is normal. When the symbiont synchronizes with the printer during execution of a print job, the symbiont sets the print queue status to DEVICE UNAVAILABLE.

Usually the synchronization period is short, so a repeat of the SHOW QUEUE command should report a print queue status other than DEVICE UNAVAILABLE. If the DEVICE UNAVAILABLE status persists for more than several seconds, power has been removed from the ScriptPrinter or the serial communication cable between the ScriptPrinter and the host has been disconnected.

NOTE

When reconnecting the ScriptPrinter to the host, turn off the printer. Turn on the printer after the connection to the host has been made. This action ensures proper synchronization between the print symbiont and the ScriptPrinter.

SHOW QUEUE/CHARACTERISTICS

Displays the queue characteristics available on your system. If you specify a characteristic name in the command line, the system returns the corresponding characteristic number.

Format

SHOW QUEUE/CHARACTERISTICS *[char-name]*

char-name
The name of a characteristic previously defined with the DEFINE/CHARACTERISTIC command. Wildcard characters are allowed, and the default is asterisk (*), which displays all characteristics.

To display the characteristics of a particular queue, use the SHOW QUEUE/FULL command.

Related Commands

SHOW QUEUE
DEFINE/CHARACTERISTIC

SHOW QUEUE/FORM

Displays the names and numbers of the forms for your system. If you specify a form name in the command line, the system returns the corresponding form number.

Format

SHOW QUEUE/FORM *[form-name][/FULL]*

form-name
The name of a form defined with the DEFINE/FORM command. Wildcard characters are allowed and the default is asterisk (*), which means display all forms on the system.

FULL
Displays page-formatting information about the form, in addition to the default name, number, stock, and description.

Related Commands

SHOW QUEUE
DEFINE/FORM

2.7 Changing the Default Parameters for a Queue

A number of ScriptPrinter parameters can be set from the PRINT command or associated with a queue. A system manager can define a systemwide logical to associate a parameter with a queue. These logical names are in the form LPS$*queue-name*_PARAMETER, where *queue-name* is the name of the queue whose jobs are to take the defaults.

Installation of the ScriptPrinter software generates the file named CPS$STARTUP.COM (located in SYS$MANAGER: on VMS, Version 4.x, or in SYS$STARTUP: on VMS Version 5.x). This file contains the systemwide logical names for each generic queue that has a default DATA_TYPE.

Only a default DATA_TYPE is defined during installation. You can change the default parameters by defining or redefining the systemwide logical names in CPS$STARTUP.COM. For example, if you requested a generic queue for ReGIS data during the installation of LN03R TTB4, the following systemwide logical name is defined:

```
$ DEFINE/SYS/EXEC LPS$REGIS_TTB4_PARAMETER "DATA_TYPE=REGIS"
```

To add default page and sheet sizes, redefine the logical name as follows:

```
$ DEFINE/SYS/EXEC LPS$REGIS_TTB4_PARAMETER -
"DATA_TYPE=REGIS,PAGE_SIZE=B,SHEET_SIZE=B"
```

2.7.1 Syntax for Parameters Associated with a Queue

Parameters specified by the LPS$*queue-name*_PARAMETER logical must follow the same rules as those entered with the PRINT/PARAMETERS command. However, parameters containing special characters or delimiters are enclosed in double quotation marks (" ") in the PRINT command and are not enclosed in extra quotation marks in the LPS$*queue-name*_PARAMETER systemwide logical.

For example, in the following PRINT command, you must enclose the PAGE_LIMIT = (4,10) in double quotation marks:

```
$ PRINT/PARAMETER=(DATA_TYPE=ANSI,"PAGE_LIMIT=(4,10)",SHEET_COUNT=3)
```

However, in the following logical name definition, you must **not** enclose the PAGE_LIMIT = (4,10) in extra double quotation marks:

```
$ DEFINE/SYS/EXEC LPS$SCRIPT_PARAMETER -
  "DATA_TYPE=ANSI,PAGE_LIMIT=(4,10),SHEET_COUNT=3"
```

2.7.2 Syntax Errors in Parameters Associated with a Queue

If an error is generated parsing the equivalence name string, the print symbiont ignores the parameter causing the error, as well as all the remaining parameters.

In the following example, there is an extra "L" in the argument to the PAGE_ORIENTATION parameter. Therefore, the print symbiont ignores the second and third parameters, PAGE_ORIENTATION and SHEET_COUNT. Only the value DATA_TYPE is retained by the symbiont.

```
$ DEFINE/SYS/EXEC LPS$generic_queue_PARAMETER-
  "DATA_TYPE=TEK4014,PAGE_ORIENTATION=LLANDSCAPE,SHEET_COUNT=5"
```

2.7.3 Order of Defaulting for Parameters

The LPS$*queue-name*_PARAMETER logical name for a generic queue supersedes the logical name defined for the execution queue. However, if there is a syntax error in the translation of the generic queue's logical name, the execution queue's logical name is translated and overrides the values set by the generic queue. The symbiont also uses the execution queue logical when the print job was submitted directly to the execution queue or to a generic queue that has no logical name associated with it.

Parameters set by the /PARAMETERS qualifier to the PRINT command override any defaults set for the queue. The symbiont selects default values for parameters, from highest to lowest priority, as follows:

1. PARAMETERS specified in the PRINT command
2. Defaults established by the LPS$*queue-name*_PARAMETER logical(s)
3. Built-in defaults
   - DATA_TYPE=ANSI
   - PAGE_SIZE=A or A4 (Same as SHEET_SIZE)

- PAGE_ORIENTATION=portrait
- SHEET_COUNT=one
- MESSAGES=nomessages
- PAGE_LIMIT=complete job
- NUMBER_UP=none
- LAYUP_DEFINITION=none
- SHEET_SIZE=A or A4 (Depends on the switch setting on the back of the printer)

In addition, PAGE_SIZE and SHEET_SIZE follow these defaulting rules:

1. SHEET_SIZE and PAGE_SIZE values come from one source. If you supply either of these parameters in the PRINT command, then the symbiont ignores the queue logical values for both parameters. If you do not supply either value on the PRINT line, the symbiont takes the values defined in the queue logical for both parameters.
2. If only PAGE_SIZE or SHEET_SIZE is supplied, then the symbiont gives the omitted parameter the same value as the one specified. This is regardless of whether the PRINT line or the queue logical supplies the value.
3. If you did not supply a PAGE_SIZE or SHEET_SIZE parameter and no logical name is associated with the queue, the symbiont uses the PAGE_SIZE set by the switch on the back of the printer.

Chapter 3

Accounting for the ScriptPrinter

This chapter describes the VMS Accounting Utility on the ScriptPrinter.

The VMS Accounting Utility records information about the use of system resources in the file SYS$MANAGER:ACCOUNTING.DAT. The print symbiont for the ScriptPrinter adds information about ScriptPrinter use to this file. For information about the VAX/VMS Accounting Utility, see the *VAX/VMS Utilities Reference Volume*.

3.1 Accounting Fields Filled in by the ScriptPrinter Print Symbiont

The print symbiont supplies information for the following fields in the accounting file for each ScriptPrinter print job:

- Pages printed
  Number of sheets printed. When you are using the NUMBER_UP parameter to print multiple pages on a sheet, this still records the physical sheets printed, not the number of logical pages printed. The number of sheets printed includes the separation (flag and trailer) pages.
- Gets from source
  Number of RMS "gets" to all files in the print job.
- QIO puts
  Number of QIO "writes" to the ScriptPrinter per job.

When a print job completes successfully, the print symbiont sends a request to the job controller to insert certain accounting data fields into the accounting log file. For successful jobs, the accounting file also includes the following message:

```
%SYSTEM-S-NORMAL, normal successful completion
```

If the print job terminates abnormally, the information the print symbiont supplies in the accounting log may be incomplete. To filter out such jobs, count only the jobs that receive the successful completion status message. Any of the following may cause the job to terminate abnormally:

- DELETE/ENTRY command
- STOP/QUEUE/RESET command
- STOP/QUEUE/ABORT command
- Operator aborted job
- Symbiont crash
- ScriptPrinter crash
- POSTSCRIPT syntax error
- POSTSCRIPT interpreter error

3.2 Accounting Fields Supplied by the Job Controller

The job controller supplies information for the following fields in the accounting file for ScriptPrinter print jobs:

- Username
  VMS username of the user who submitted the print job.
- Account
  Account name of the user who submitted the print job.
- UIC
  UIC of username of the user who submitted the print job.
- ProcessID
  VMS Process ID of the user who submitted the print job.
- Start Time
  When the print job reached the top of the device queue; **not necessarily** when the print job actually began to print.
- Finish Time
  When the job finished printing.

- Symbiont Time
  The wall-clock time it took for the job to print (finish time minus start time).
- Priority
  Queue priority of the print job.
- Final Status Code
  Exit status of the print job. The status code for success is "00040001".
- Queue Entry
  Queue entry number. This is the number VMS assigns to every print job. In this example, 222 is the queue entry number:

  ```
  $ PRINT/QUEUE=LNO3R$SCRIPT4/NOTIFY test.mem
    Job TEST (queue LN03R$SCRIPT4, entry 222) started on queue
    LN03R$SCRIPT4
  ```

- Queue Name
  Name of the print device queue.
- Queue Job
  Print job name, as printed on the job flag page.
- Final Status test = F$MESSAGE
  The text for the Final Status code. The text for completion of a normal job is:

  ```
  %JBC-S-NORMAL,  Normal successful completion
  ```

3.3 Accounting Fields Left Blank by the Job Controller

The job controller fills the following fields with blanks in the accounting file for ScriptPrinter print jobs:

- Owner ID
- Terminal name
- REMOTE node name
- Remote ID

3.4 Accounting Fields Filled with Zeros by the Job Controller

The job controller fills the following fields with zeros in the accounting file for ScriptPrinter print jobs:

- Privilege <31–00>
- Privilege <63–32>
- Processor Time

Chapter 4

Flag, Burst, Trailer, and Log Pages

The ScriptPrinter can generate flag, burst, trailer, and log pages for each job printed. It can also generate flag, burst, log, and trailer pages for each file printed.

4.1 Job Flag and Trailer Pages

The printing of job separation pages depends on a corresponding attribute of the job's print queue. These attributes are initially defined in the CPS$STARTUP.COM with the SEPARATE qualifier in the INITIALIZE/QUEUE or SET QUEUE command. The default, as defined during the installation of the host software, is to print the job burst and job trailer pages, but not the job flag page.

```
$ INITIALIZE/QUEUE-
          /SEPARATE=(BURST,TRAILER)
```

The system manager can change the BURST and TRAILER attributes for a queue, by using the /SEPARATE qualifier options [NO]BURST and [NO]TRAILER with the SET QUEUE command.

```
$ SET QUEUE/SEPARATE=(noburst,notrailer)
```

Job flag, job burst, job trailer, and the job log pages print in the default POSTSCRIPT context of the ScriptPrinter, not in the context of the user's job. That is they are not affected by parameters set with the PRINT/PARAMETER command or by the POSTSCRIPT file being printed.

For examples of the job separation pages, see the *VAX/VMS Software Installation Guide: ScriptPrinters*.

NOTE

Even if you choose to have no separation pages, the symbiont always prints a job trailer page if an error occurs in the print job.

When the symbiont deletes a job before the job burst or job flag page prints, you may not get the job burst or job flag pages. However, the job log and job trailer pages print if you requested them.

4.1.1 Job Flag Page Items

The job flag page contains the following items:

- Optional company name (defined by system logical name PSM$ANNOUNCE)
- Optional note (from /NOTE=qualifier on the PRINT command)
- Client node name and user's name
- Job number
- Job name (from /NAME qualifier on the PRINT command or name of first file in the job)
- Owner User Identification Code (UIC)
- Account name
- Process priority
- Submit queue (queue where job was originally submitted)
- Time submitted
- Print queue (name of execution queue)
- Printer (terminal) device name
- Time started

4.1.2 Job Trailer Page Items

The job trailer page contains the following items:

- End of job banner
- Client node name and user's name
- Job number
- Job name (from /NAME qualifier in PRINT command or name of first file in the job)
- Owner User Identification Code (UIC)
- Account name
- Process priority
- Submit queue (queue where job was originally submitted)
- Time submitted
- Print queue (name of execution queue)
- Printer (terminal) device name
- Time started
- Time finished
- Qualifiers (the PRINT command qualifiers, other than /PARAMETERS, in effect for this job)
- Parameters (the parameters in effect for the job, both those specified by the user in the PARAMETERS qualifier of the PRINT command and those defaulted from system logicals)
- Number of sheets printed (includes all flag, burst, and trailer pages)
- Two messages (first two "interesting" messages received from the printer)

NOTE

Printing two interesting messages on the job trailer page is a new feature with the ScriptPrinter, Version 2.0. "Interesting" messages are those that would help the user determine what went wrong with his print job. An example of the messages that may appear on the job trailer page is as follows:

```
4-AUG-1988 11:03  %JBC-F-JOBABORT, job aborted during execution
4-AUG-1988 11:03  %LPS-W-SYNERR, syntaxerror: Input ended in string
or procedure body - offending command is --nostringval--
```

Messages that would not be included in this category are those that accompany every job: start and end messages; hardware printer error messages, like "out of paper"; messages related to communications; and messages that the symbiont uses to manage resources in the printer.

If the print job contains errors, the symbiont prints these two messages on the job trailer page regardless of the MESSAGES option selected.

4.2 File Flag and Trailer Pages

The printing of a file flag or file trailer page depends on a corresponding attribute of the job's print queue. These attributes are initially defined with the DEFAULT qualifier in the INITIALIZE/QUEUE or SET QUEUE command. The default, as defined during the installation of the host software, is to print neither file flag nor file trailer pages.

```
$ INITIALIZE/QUEUE-
          /DEFAULT=(NOBURST,NOFLAG,NOTRAILER,FORM=CPS$DEFAULT)
```

```
$ SET QUEUE/DEFAULT=(NOBURST,NOFLAG,NOTRAILER,FORM=CPS$DEFAULT)
```

To override the file flag and file trailer attributes for a file, use the /[NO]FLAG and /[NO]TRAILER qualifiers on the PRINT command. In the following example, each file in the job will be preceded by a flag page and followed by a trailer page:

```
$ PRINT/FLAG=ALL/TRAILER=ALL file1.ps,file2.ps,file3.ps
```

File flag, burst, and trailer pages print in the same POSTSCRIPT context as the user's print job. They are affected by the parameters set in the PRINT/PARAMETER command and by the POSTSCRIPT file being printed.

NOTE

If a job is aborted, you may not get a file flag or file trailer for the file that was printing.

4.2.1 File Flag Page Items

The file flag page contains the following items:

- Optional company name (defined by the system logical name PSM$ANNOUNCE)
- Optional note (from /NOTE=qualifier on the PRINT command)
- Client node name and user's name
- Job number
- File identification (file name, extension, and version only)
- File specification (full file specification)
- Date last modified
- Owner User Identification Code (UIC)
- Length of file
- Longest record
- Process priority
- Submit queue (print queue where job was originally submitted)
- Time submitted
- Print queue (name of execution queue)
- Printer (terminal) device name

4.2.2 File Trailer Page Items

The file trailer page contains the following items:

- End of file banner
- Client node name and user's name
- Job number
- File identification (file name, extension, and version only)
- File specification (full file specification)
- Date last modified
- Owner User Identification Code (UIC)
- Length of file
- Longest record
- Process priority

- Submit queue (print queue where job was originally submitted)
- Time submitted
- Print queue (name of execution queue)
- Printer (terminal) device name
- Qualifiers (the PRINT command qualifiers, other than /PARAMETERS, in effect for this file)
- Parameters (the parameters in effect for the job, both those specified by the user in the PARAMETERS qualifier of the PRINT command and those defaulted from system logicals)

4.3 Job and File Burst Pages

Job and file burst pages are identical to their respective job and file flag pages, with the addition of a gray border printed about 1/4 inch from each edge of the paper. The default, as defined during the installation of the host software, is to print a job burst page (SEPARATE=BURST). For both a burst and a flag page to print, the system manager must modify the ScriptPrinter queue parameters to include SEPARATE=(BURST,FLAG).

To get file burst pages, include the BURST qualifier in your PRINT command. When you specify PRINT/BURST, you get only a file burst page. If you want both file burst and file flag pages, you must use both the BURST and FLAG qualifiers in the command, that is PRINT/BURST/FLAG.

4.4 Job Log Page

Use MESSAGES=PRINT in the PARAMETERS qualifier of the PRINT command to print a job log page. When printed, the job log page precedes the job trailer page. If you are not printing trailer pages, the job log page is the last page of a job.

NOTE

If your print job produced an error, the symbiont automatically generates a trailer page. The first two interesting messages print on the trailer page (see Section 4.1.2).

If you are expecting message output in response to POSTSCRIPT **user-data** commands, use MESSAGES=KEEP to record the messages in the log file. This ensures you against losing any data. The job log page has a limited message area.

4.4.1 Job Log Page Items

The job log page contains the following items:

- Job log page banner
- Client node name and user's name
- Job number
- Job name (from NAME qualifier in PRINT command or name of first file in the job)
- The messages area (If there are more than 40 message lines, this area contains the first 36 message lines, an ellipsis, and as much of the last three messages as space allows on the job log page.)
- The total number of messages (may be different from the number printed)

4.5 File Error Page

The symbiont generates a file error page if any error occurred during the printing of your file. If the print symbiont cannot access the requested file at print time, the file error page is the only file page printed.

NOTE

When you submit a job with /COPIES=n and the file to be printed is deleted before it is printed, the printer still prints **n** copies of the file error page.

4.5.1 File Error Page Items

The file error (log) page contains the following items:

- File log page banner
- Client node name and user's name
- Job number
- File identification (file name, extension, and version only)
- File specification (full file specification)
- The messages area

Chapter 5

The Device Control Library

To implement PRINT qualifiers that affect the output appearance, the symbiont uses modules from a **device control library**. The device control library is a VMS text library file, which contains modules consisting of POSTSCRIPT procedures. The symbiont extracts the appropriate modules and inserts them into the POSTSCRIPT data stream being sent to the printer during the print job.

The device control library of the ScriptPrinter software is named SYS$LIBRARY:CPS$DEVCTL.TLB. The names of the modules supported by DIGITAL in the library begin with LPS$$ or LPS$.

This chapter describes the ScriptPrinter device control library and explains how to include device control modules in a print job.

5.1 Device Control Modules

Device control modules routinely perform the following functions:

- Control device dependent functions
- Provide commonly used POSTSCRIPT procedures
- Transmit information about the print job back to the symbiont for control purposes

Some modules implement the following parameters to the PRINT command:

NUMBER_UP	PAGE_SIZE
OUTPUT_TRAY	SHEET_COUNT
PAGE_LIMIT	SHEET_SIZE
PAGE_ORIENTATION	

5.1.1 Order of Device Control Modules in a Print Job

A device control library module can be included in a print job by including it in the setup of a device queue or by specifying it on the command line. The common ways to include a device control library module in a print job are listed here:

- DEFINE/FORM/SETUP=(module[,...])
- INIT/QUEUE/SEPARATE=RESET=([module])
- PRINT/SETUP=(module[,...])
- PRINT/PARAMETER=(parameter[,...])

Table 5–1 shows the data that can be included in a print job and the source of the request to send that data. The table entry "Print symbiont" indicates that the print symbiont sent the data in response to a request other than the ones listed above, such as PRINT/FLAG. The double horizontal lines indicate the point at which the default POSTSCRIPT environment is restored in the printer.

NOTE

Future implementations may not require the same modules in the same order.

Table 5–1: Order of Data Sent to a Print Job

Data Sent to Printer	Origin of Request
Persistant prologue checking	Print symbiont
LPS$$GetSheetCount	Print symbiont
LPS$$InitPSDevice	Print symbiont
LPS$$SetOutputTray	Print/Parameter
Job flag page	Init/Queue/Separate=Flag

Table 5–1 (Cont.): Order of Data Sent to a Print Job

	Data Sent to Printer	Origin of Request
	Job burst page	Init/Queue/Separate=Burst
	POSTSCRIPT form setup modules	Define/Form/Setup
	LPS$$LoadDict	Print symbiont
	Any /PARAM modules	Print/Parameter
	Layup definition translator output	Print/Param=Layup
	LPS$$SetContext	Print symbiont
This section is repeated for each subsequent file in the job	POSTSCRIPT file setup modules	Print/Setup or File/Setup
	File burst page	Print symbiont
	File flag page	Print symbiont
	LPS$$VMSTATUS (xlated jobs)	Print symbiont
	File n	User
	File error page[1]	Print symbiont
	File trailer page	Print symbiont
	LPS$$FlushPages	Print/Param=Nup or Layup
	Reset sequence	Init/Queue/Separate=Reset
	CTRL/D	Print symbiont
	LPS$$GetSheetCount	Print symbiont
	LPS$$InitPSDevice	Print symbiont
	LPS$$SetOutputTray	Print/Parameter
	Job log page	Print
	Job trailer page	Init/Queue/Separate=Trailer

[1]The symbiont only generates a file error page if an error occurs in the processing of the file.

5.1.1.1 Including Modules with Commands

Some device control modules are sent only when specified by the user or system manager. These modules are invoked by the PRINT, INITIALIZE QUEUE, or START QUEUE commands with the following qualifiers: PARAMETERS, SETUP, FLAG, TRAILER, FORM, SEPARATE. Not all qualifiers are available for each command. Refer to Chapters 2 and 8 for more information about these commands and qualifiers.

NOTE

For information on defining forms, including setup modules in forms, creating new device control libraries, inserting modules into the new libraries, and setting up a library search list, see Chapter 7.

5.1.2 Creating Device Control Modules

You can add modules to the ScriptPrinter device control library. These modules must contain POSTSCRIPT code, because the contents of the modules are sent directly to the ScriptPrinter, bypassing translation.

DIGITAL recommends that you take advantage of multiple device control libraries. Creating a separate library for your own POSTSCRIPT modules allows you to use the library with other POSTSCRIPT printers and safeguards your custom-designed modules from future software updates. For more information on using setup modules, creating libraries, and setting up library search orders, see Chapter 7.

New module names should **not** begin with LPS$$ or LPS$.

When creating a new device control module, follow these steps:

1. Design the POSTSCRIPT code for the desired function.
2. Test the new function in a controlled environment.
3. Add the code to your device control library. See Section 7.1.3 for how to create and add your module to a library.
4. Perform any necessary queue or form setup. For example:
   - If the function should be associated with a form, specify the module in the DEFINE/FORM command.
   - If the function should be performed between jobs, specify the module as a reset module in the INIT/QUEUE command.

5. Reset and restart the queue and test the function in the full VMS environment.

Also see Appendix A and the *PostScript Language Tutorial and Cookbook* for information on creating POSTSCRIPT programs for the ScriptPrinter.

5.2 Error Handler

The device control library includes an error handler to help debug POSTSCRIPT programs. The error handler prints the last partial page of output, as well as information to help identify the error.

5.2.1 Including the Error Handler in a Print Job

The error handler is not automatically included each time a job prints (unless your system manager has changed this default). Therefore, you must explicitly invoke it, as follows:

```
$ PRINT/SETUP=LPS$ERRORHANDLER filename
```

The error handler returns USERDATA messages. You can send these messages to a file or printer by using a /PARAMETERS=MESSAGES parameter, or to your terminal with the PRINT/NOTIFY command (see Sections 8.2.3 and 8.1.7, respectively).

5.2.2 Error Handler Output

When an error occurs, the error handler executes a **showpage** command to print the last partial page of output (see Example 5–1). It also gives the following information:

1. The name of the error
2. The POSTSCRIPT operator that encountered the error
3. The contents of the operand stack

   The error handler displays the value of each object on the stack, with numbers in decimal. All elements of arrays, strings, and procedures are displayed recursively. Indicators describe other objects, for example,
   –savelevel– for a save object.

   The first item displayed is the object on the top of the stack.

4. The contents of the execution stack

   The execution stack contains partial procedures that are being executed. The top object is a procedure containing the operators and operands still to be executed. The second object is the unexecuted part of the calling procedure.

   The error handler does not display the entire execution stack, but displays only that part of the execution stack that represents the user's job.

5. Information about the graphics state:
   - Current transformation matrix
   - Color (a setgray value)
   - Current position
   - Line width
   - Line cap
   - Line join
   - Flatness
   - Miter limit
   - Dash pattern

5.2.2.1 Output Format

Each POSTSCRIPT data type is easily identifiable, usually in the way that data type appears in a POSTSCRIPT source file. This section describes how the error handler represents each data type.

- Arrays are displayed recursively, so that each element in an array is fully expanded, even if it is another array. Objects in an array are expanded only to a depth of two, to prevent indefinite recursion when displaying an array that contains itself.

  Arrays are executable and nonexecutable. Executable arrays are procedures displayed in braces ({ }) and nonexecutable arrays are displayed as several objects in brackets ([ ]). If the array has no read access or if the recursion depth has been exceeded, the array is represented by one of the following:

  –array– for normal arrays
  –proc– for executable arrays
  –packedarray– for packed arrays
  –packedproc– for packed executable arrays

- A boolean object is represented by true or false, depending on its value.
- A dictionary object is represented by –dictionary–.
- A file object is represented by –filestream–.
- A font object is represented by –fontid–.
- An integer is represented by a decimal number.
- A mark object is represented by –mark–.
- A name object is represented by the literal name of the object, preceded by a slash for literal names.
- A null object, for example, the initial value of each element of an uninitialized array, is represented by –null–.
- An operator is represented by the operator name, preceded by two slashes.
- A real object is represented by a decimal number, with a decimal point and at least one digit after the decimal point.
- A save object is represented by –savelevel–.
- A string object is represented by the ASCII text of the string in parentheses (string), just as the string would be entered in a POSTSCRIPT file.
- The POSTSCRIPT language may be extended to include new data types that are unknown to the error handler. Objects with unknown types are represented as two question marks followed by the name of the unknown type.

5.2.3 POSTSCRIPT Environment

The error handler references operators from the dictionary **systemdict**, rather than using definitions that may have been modified by the user program.

In some cases, a program can behave differently when the error handler is loaded. For example, executing the **exit** operator outside a looping context causes an **invalidexit** error if the error handler is not loaded. However, if the error handler is loaded, the program exits without generating an error.

5.2.4 Error Handler Example

The sample log file in this section is for the following POSTSCRIPT program:

```
3 array dup dup 0 exch put
/myproc { [ 8 8 ] 0 0 div setdash } def
100 200 moveto
myproc
```

The following command includes the error handler and causes a log file to be generated:

```
$ PRINT/PARAMETER="MESSAGES=KEEP"/SETUP=LPS$ERRORHANDLER  filename
```

Example 5–1 shows the error handler output that is appended to the log file when the program executes.

Example 5–1: Sample Error Handler Log File

```
ERROR: undefinedresult
OFFENDING COMMAND: div

OPERAND STACK:

0
0
[ 8 8 ]
[ [ [ -array- -null- -null- ] -null- -null- ] -null- -null-  ]

EXECUTION STACK:

{setdash }

GRAPHICS STATE:
Current Matrix: [ 0.0 4.16667 4.16667 0.0 0.0 0.0 ]
Color: 0
Current position: x = 100.0, y = 200.0
Line width: 1.0
Line cap: 0
Line join: 0
Flatness: 1.0
Miter limit: 10.0
Dash pattern: [] 0.0
```

The array defined at the start of the example file includes a pointer to itself as its first element. It is expanded two levels deep. The innermost version of the array is represented simply as –array–.

5.3 Additional Character Encodings

POSTSCRIPT provides a character set encoded using the Adobe Standard Encoding vector. When a POSTSCRIPT program calls a font, the **findfont** operator uses this vector to identify characters.[1]

POSTSCRIPT Version 47.2 also provides the ISOLatin1 encoding vector, named ISOLatin1Encoding, which is the encoding scheme used by the ISO Latin Alphabet Number 1 Standard (ISO 8859/1). In addition, POSTSCRIPT Version 47.2 provides all the characters used in the ISO Latin Alphabet Number 1 in all the ScriptPrinter fonts (except the Symbol font). See Appendix B of the *LN03R ScriptPrinter Programmer's Supplement* for a list of the characters included in the ISO Latin Alphabet Number 1 Standard (ISO 8859/1).

The ScriptPrinter device control library provides another encoding vector, as well as Adobe Standard Encoding and ISOLatin1. This is the DECMCS (DIGITAL Multinational Character Set) encoding. See Appendix B of the *LN03R ScriptPrinter Programmer's Supplement* for a list of the characters included in the DIGITAL Multinational Character Set. All the characters encoded in all these vectors are present in all the POSTSCRIPT fonts except Symbol.

Characters in the ISOLatin1 and DECMCS encodings do not have separate font metric files (.AFM files).[2] If you need to look up character information in a font metric file, look up the character by its name, not by its encoding.

[1] The Symbol font has its own encoding vector, which is used when the Symbol font is called.

[2] For more information on .AFM files, refer to the chapter on font metrics in the *LN03R ScriptPrinter Programmer's Supplement.*

5.3.1 Using the Additional Encodings

To use a character encoding other than the Adobe Standard Encoding, you must follow these steps:

1. Define a procedure, **encodefont** for instance, in the prologue of your POSTSCRIPT program that applies an encoding vector to a font to create a new font. Before invoking **findfont**, your program calls **encodefont** to encode the font with the desired encoding vector. The code for **encodefont** is as follows:

```
/encodefont {
    findfont dup                    % Get the old font dict
    maxlength dict begin            % Make a new one just as big
    {   1 index /FID ne             % Copy everything but FID
        {   def }
        {   pop pop }
        ifelse }
    forall
    /Encoding exch def              % Install the new encoding
    dup /FontName exch def          % new font dict is still current.
    currentdict definefont          % Create the new font.
    end
} bind def
```

2. Create your new font using the **encodefont** procedure. The procedure takes three arguments:

   - Name of the new font
   - The encoding vector
   - Name of the old font

   encodefont creates a new font with the new name and encoding, and returns the new font. Choose any name for your new font. You might choose a name that includes the name of the old font and the encoding vector.

   You can create and immediately use your new font or you can create the new font and use it later in your program. For example, to create and use a Times-Roman font encoded with ISO Latin 1, use the following code:

```
/Times-Roman-ISOLatin1 ISOLatin1Encoding
        /Times-Roman encodefont
12 scalefont setfont
```

To create and use the font separately in your program, use this code:

```
/Times-Roman-ISOLatin1 ISOLatin1Encoding
        /Times-Roman encodefont pop
.
.
.
/Times-Roman-ISOLatin1 findfont 12 scalefont setfont
```

To use the DECMCS encoding, you must also include the device control library module (LPS$DECMCSENCODING) that defines this encoding vector. The LPS$DECMCSENCODING module only provides encoding vectors. The module does not define characters that were undefined in versions of POSTSCRIPT earlier than Version 40.

Example 5–2 at the end of this chapter shows how the DECMCS encoding vector is defined.

5.3.2 Including the DECMCS Encoding Module in a Print Job

To use the DECMCS character set, specify the following:

```
$ PRINT/SETUP=(LPS$DECMCSENCODING) filename.ps
```

5.3.3 Defining ISOLatin1 for Non-DIGITAL Printers

POSTSCRIPT printers not made by DIGITAL that use a version of POSTSCRIPT earlier than Version 40 may need to define an ISOLATIN1 encoding vector. For example, you have a POSTSCRIPT program that requires the ISOLatin1 encoding vector to print on a typesetter that uses an early version of POSTSCRIPT.

In these cases, you can use the **encodefont** procedure, but you also have to define the ISOLatin1Encoding in the prologue of your POSTSCRIPT program. Example 5–3 at the end of this chapter shows how to define the ISOLatin1 encoding vector. Aternatively, extract the LPS$ISOLATIN1ENCODING module from the device control library:

```
$LIB -
   /Extract=LPS$ISOLatinEncoding -
   /Out=ISOLatinEncoding.ps -
   Sys$Library:CPS$DEVCLT.TLB
```

The LPS$ISOLATIN1ENCODING module only provides encoding vectors. The module does not define characters that were undefined in versions of POSTSCRIPT earlier than Version 40.

5.3.4 Examples of Encoding Vectors

The examples in this section demonstrate how the ISOLatin1 and DEC MCS encoding vectors are defined. See the *LN03R ScriptPrinter Programmer's Supplement* for more information about how these encoding vectors are defined. The encoding vector DECMCSEncoding is defined by changing the vector ISOLatin1Encoding.

Example 5–2: Defining the DECMCS Encoding Vector

```
%! DECMCSEncoding.PS
%
% Create DEC Multinational Character Set (MCS) encoding vector.

/DECMCSEncoding ISOLatin1Encoding 256 array copy def

mark
  8#177 8#240 8#244 8#246 8#254 8#255 8#256 8#257
  8#264 8#270 8#276 8#320 8#336 8#360 8#376 8#377
counttomark
{DECMCSEncoding exch /questionmirror put}
repeat
% stack now contains   mark
  8#250 /currency
  8#327 /OE
  8#335 /Ydieresis
  8#367 /oe
  8#375 /ydieresis
counttomark -1 bitshift % divide by 2
{DECMCSEncoding 3 1 roll put}
repeat
% stack now contains   mark
cleartomark
```

Example 5–3: Defining the ISOLatin1 Encoding Vector

```
%! ISOLatin1Encoding.PS
%
% Create ISO Latin 1 encoding vector, if it does not already exist.

mark
/ISOLatin1Encoding
8#000 1 8#054 {StandardEncoding exch get} for
  /minus % at position 8#055, was hyphen
8#056 1  8#217 {StandardEncoding exch get} for
/dotlessi % at position 8#220
8#301 1 8#317 {StandardEncoding exch get} for % all the accents
/space  /exclamdown /cent  /sterling
/currency /yen  /brokenbar /section
/dieresis /copyright /ordfeminine /guillemotleft
/logicalnot /hyphen  /registered /macron
/degree  /plusminus /twosuperior /threesuperior
/acute  /mu  /paragraph /periodcentered
/cedilla /onesuperior /ordmasculine /guillemotright
/onequarter /onehalf /threequarters  /questiondown
/Agrave  /Aacute  /Acircumflex /Atilde
/Adieresis /Aring  /AE  /Ccedilla
/Egrave  /Eacute  /Ecircumflex /Edieresis
/Igrave  /Iacute  /Icircumflex /Idieresis
/Eth  /Ntilde  /Ograve  /Oacute
/Ocircumflex /Otilde  /Odieresis /multiply
/Oslash  /Ugrave  /Uacute  /Ucircumflex
/Udieresis /Yacute  /Thorn  /germandbls
/agrave  /aacute  /acircumflex /atilde
/adieresis /aring  /ae  /ccedilla
/egrave  /eacute  /ecircumflex /edieresis
/igrave  /iacute  /icircumflex /idieresis
/eth  /ntilde  /ograve  /oacute
/ocircumflex /otilde  /odieresis /divide
/oslash  /ugrave  /uacute  /ucircumflex
/udieresis /yacute  /thorn  /ydieresis
/ISOLatin1Encoding where not {256 array astore def} if
cleartomark
```

Chapter 6

Layup Definition Files

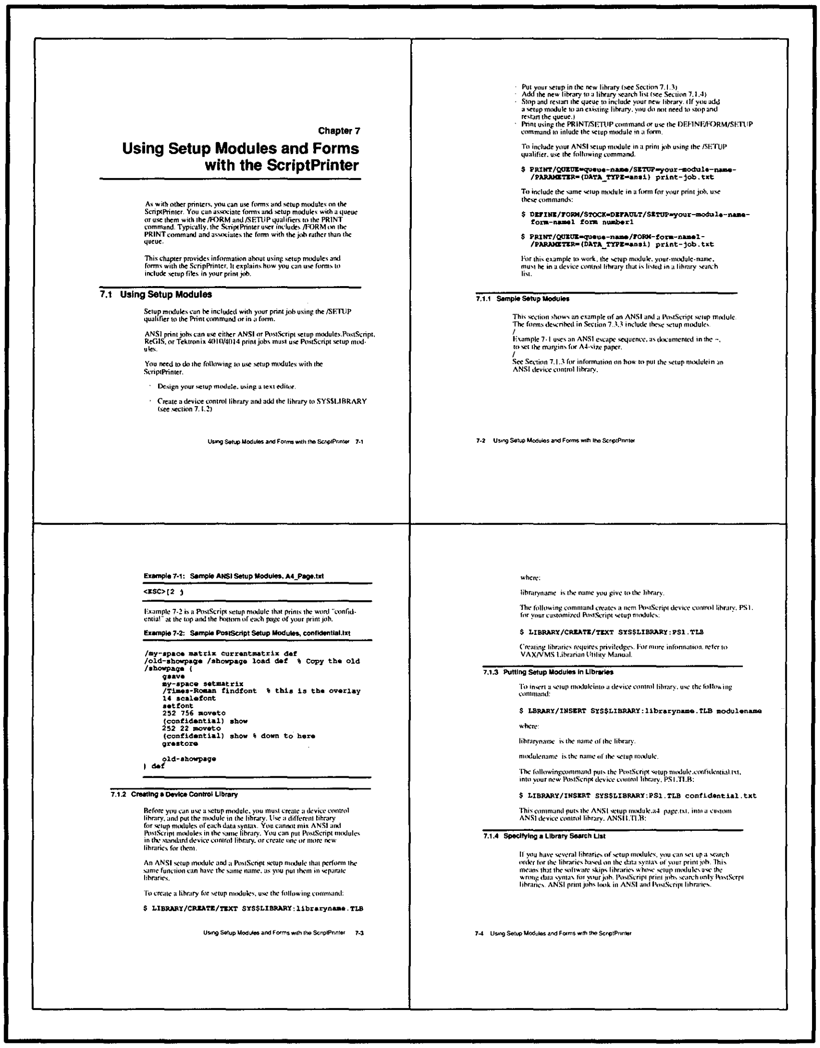
Layup is a ScriptPrinter feature that can be associated with a queue or used with the PRINT/PARAMETERS command. To understand layup, it is important to remember the distinction between a sheet and a page. A sheet is a piece of paper. A page (sometimes called a logical page) is an image that you want to print, generally on a single sheet. For example, on this sheet, all the text from the top line to the page number is one page.

Using layup, you can control the mapping of pages to sheets. This lets you specify where on a sheet a page is printed. Layup also lets you set the number of pages printed on a sheet. Layup can be used with any file that can be printed on a ScriptPrinter, including translated files. It is implemented through the NUMBER_UP and LAYUP_DEFINITION parameters to the PRINT/PARAMETER command. (See Sections 8.2.2 and 8.2.4 for more information on using these parameters.)

The graphic arts term for printing multiple pages per sheet is **number-up**. The NUMBER_UP parameter determines the maximum number of **page spots** on a sheet. These are the locations on a sheet where a page could be printed. For example, if NUMBER_UP=4, you can print four pages on a single sheet. This means there are four page spots on the sheet, even if your print job actually has fewer than four pages. Figure 6–1 shows an example of a sheet printed with NUMBER_UP=4.

The LAYUP_DEFINITION parameter calls a layup definition file. You can use layup definition files to set sheet margins, page borders, a page grid, and a number of other options. This chapter describes how to create a layup definition file.

Figure 6–1: Example of Printing Four Pages to a Sheet (4-UP) with Borders



Chapter 7

Using Setup Modules and Forms with the ScriptPrinter

As with other printers, you can use forms and setup modules on the ScriptPrinter. You can associate forms and setup modules with a queue or use them with the /FORM and /SETUP qualifiers to the PRINT command. Typically, the ScriptPrinter user includes /FORM on the PRINT command and associates the form with the job rather than the queue.

This chapter provides information about using setup modules and forms with the ScripPrinter. It explains how you can use forms to include setup files in your print job.

7.1 Using Setup Modules

Setup modules can be included with your print job using the /SETUP qualifier to the Print command or in a form.

ANSI print jobs can use either ANSI or PostScript setup modules.PostScript, ReGIS, or Tektronix 4010/4014 print jobs must use PostScript setup mod-ules.

You need to do the following to use setup modules with the ScriptPrinter.

- Design your setup module, using a text editor.
- Create a device control library and add the library to SYS$LIBRARY (see section 7.1.2)

Using Setup Modules and Forms with the ScriptPrinter 7-1

- Put your setup in the new library (see Section 7.1.3)
- Add the new library to a library search list (see Section 7.1.4)
- Stop and restart the queue to include your new library. (If you add a setup module to an existing library, you do not need to stop and restart the queue.)
- Print using the PRINT/SETUP command or use the DEFINE/FORM/SETUP command to inlude the setup module in a form.

To include your ANSI setup module in a print job using the /SETUP qualifier, use the following command.

```
$ PRINT/QUEUE=queue-name/SETUP=your-module-name-
  /PARAMETER=(DATA_TYPE=ansi) print-job.txt
```

To include the same setup module in a form for your print job, use these commands:

```
$ DEFINE/FORM/STOCK=DEFAULT/SETUP=your-module-name-
  form-name1 form number1

$ PRINT/QUEUE=queue-name/FORM-form-name1-
  /PARAMETER=(DATA_TYPE=ansi) print-job.txt
```

For this example to work, the setup module, your-module-name, must be in a device control library that is listed in a library search list.

7.1.1 Sample Setup Modules

This section shows an example of an ANSI and a PostScript setup module. The forms described in Section 7.3.3 include these setup modules.
/
Example 7-1 uses an ANSI escape sequence, as documented in the ~, to set the margins for A4-size paper.
/
See Section 7.1.3 for information on how to put the setup modulein an ANSI device control library.

7-2 Using Setup Modules and Forms with the ScriptPrinter

Example 7-1: Sample ANSI Setup Modules, A4_Page.txt

```
<ESC>[2 }
```

Example 7-2 is a PostScript setup module that prints the word "confid-ential" at the top and the bottom of each page of your print job.

Example 7-2: Sample PostScript Setup Modules, confidential.txt

```
/my-space matrix currentmatrix def
/old-showpage /showpage load def  % Copy the old
/showpage {
    gsave
    my-space setmatrix
    /Times-Roman findfont  % this is the overlay
    14 scalefont
    setfont
    252 756 moveto
    (confidential) show
    252 22 moveto
    (confidential) show % down to here
    grestore

    old-showpage
} def
```

7.1.2 Creating a Device Control Library

Before you can use a setup module, you must create a device control library, and put the module in the library. Use a different library for setup modules of each data syntax. You cannot mix ANSI and PostScript modules in the same library. You can put PostScript modules in the standard device control library, or create one or more new libraries for them.

An ANSI setup module and a PostScript setup module that perform the same function can have the same name, as you put them in separate libraries.

To create a library for setup modules, use the following command:

```
$ LIBRARY/CREATE/TEXT SYS$LIBRARY:libraryname.TLB
```

Using Setup Modules and Forms with the ScriptPrinter 7-3

where:

libraryname is the name you give to the library.

The following command creates a nem PostScript device control library, PS1, for your customized PostScript setup modules:

```
$ LIBRARY/CREATE/TEXT SYS$LIBRARY:PS1.TLB
```

Creating libraries requires priviledges. For more information, refer to VAX/VMS Librarian Utility Manual.

7.1.3 Putting Setup Modules in Libraries

To insert a setup moduleinto a device control library, use the following command:

```
$ LBRARY/INSERT SYS$LIBRARY:libraryname.TLB modulename
```

where:

libraryname is the name of the library.

modulename is the name of the setup module.

The followingcommand puts the PostScript setup module.confidential.txt, into your new PostScript device control library, PS1.TLB:

```
$ LIBRARY/INSERT SYS$LIBRARY:PS1.TLB confidential.txt
```

This command puts the ANSI setup module.a4 page.txt, into a custom ANSI device control library, ANSI1.TLB:

7.1.4 Specifying a Library Search List

If you have several libraries of setup modules, you can set up a search order for the libraries based on the data syntax of your print job. This means that the software skips libraries whose setup modules use the wrong data syntax for your job. PostScript print jobs search only PostScrpt libraries. ANSI print jobs look in ANSI and PostScript libraries.

7-4 Using Setup Modules and Forms with the ScriptPrinter

MLO-001472

6.1 Creating a Layup Definition File

Layup definition files must be located in a directory that is pointed to by the systemwide logical name LPS$LAYUP. At installation, the protection for this directory is set so that only a privileged user can add files to the directory.

When creating a layup definition file with selected layup options, follow these rules:

- A layup definition file must have the file type LUP. The file name can use letters, numbers, underscore (_), and hyphen (-) as long as the hyphen is not the first character.
- Start each layup option on a new line. Blank lines and white space within lines are ignored.
- Lines beginning with an exclamation point (!) are considered comments and are ignored.
- Do not abbreviate layup options.
- Layup options can use either lowercase or uppercase letters as layup is not case sensitive.
- Some layup options take values, indicated by an equal sign (=). You must use at least one value after an equal sign. Separate multiple values with commas.
- If you specify a layup option more than once in a layup definition file, the last setting is the one that layup uses.

The following sections describe the layup options that are set in a layup definition file and show some sample layup definition files. Section 8.2.2 shows how to include a layup definition file in a print job.

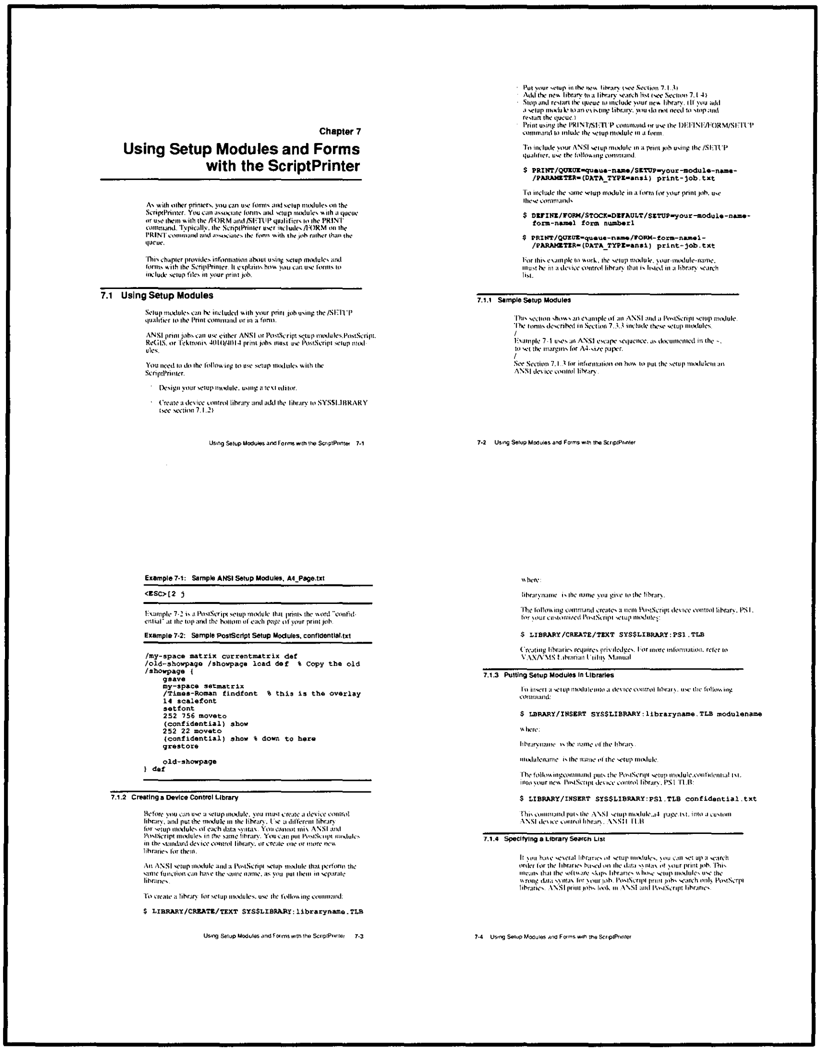
6.1.1 Borders

The BORDERS option draws a border around each page. This is often useful to differentiate pages when you are printing a job using NUMBER_UP. Borders are only drawn around actual pages, not around all page spots. This allows you to distinguish between blank pages in a document and nonexistent pages. Figure 6–1 shows pages printed using the BORDERS option. Figure 6–2 shows the same pages printed without the borders option.

You can turn off the borders, using the option NOBORDERS.

If you use NUMBER_UP, the default is to draw borders.

Figure 6–2: Example of Printing Four Pages to a Sheet (4-UP) without Borders



Chapter 7

Using Setup Modules and Forms with the ScriptPrinter

As with other printers, you can use forms and setup modules on the ScriptPrinter. You can associate forms and setup modules with a queue or use them with the /FORM and /SETUP qualifiers to the PRINT command. Typically, the ScriptPrinter user includes /FORM on the PRINT command and associates the form with the job rather than the queue.

This chapter provides information about using setup modules and forms with the ScripPrinter. It explains how you can use forms to include setup files in your print job.

7.1 Using Setup Modules

Setup modules can be included with your print job using the /SETUP qualifier to the Print command or in a form.

ANSI print jobs can use either ANSI or PostScript setup modules.PostScript, ReGIS, or Tektronix 4010/4014 print jobs must use PostScript setup modules.

You need to do the following to use setup modules with the ScriptPrinter.

- Design your setup module, using a text editor.
- Create a device control library and add the library to SYS$LIBRARY (see section 7.1.2)

Using Setup Modules and Forms with the ScriptPrinter 7-1

- Put your setup in the new library (see Section 7.1.3)
- Add the new library to a library search list (see Section 7.1.4)
- Stop and restart the queue to include your new library. (If you add a setup module to an existing library, you do not need to stop and restart the queue.)
- Print using the PRINT/SETUP command or use the DEFINE/FORM/SETUP command to inlude the setup module in a form.

To include your ANSI setup module in a print job using the /SETUP qualifier, use the following command.

```
$ PRINT/QUEUE=queue-name/SETUP=your-module-name-
  /PARAMETER=(DATA_TYPE=ansi) print-job.txt
```

To include the same setup module in a form for your print job, use these commands

```
$ DEFINE/FORM/STOCK=DEFAULT/SETUP=your-module-name-
  form-name1 form number1
```

```
$ PRINT/QUEUE=queue-name/FORM-form-name1-
  /PARAMETER=(DATA_TYPE=ansi) print-job.txt
```

For this example to work, the setup module, your-module-name, must be in a device control library that is listed in a library search list.

7.1.1 Sample Setup Modules

This section shows an example of an ANSI and a PostScript setup module. The forms described in Section 7.3.3 include these setup modules.
/
Example 7-1 uses an ANSI escape sequence, as documented in the ~, to set the margins for A4-size paper.
/
See Section 7.1.3 for information on how to put the setup modulein an ANSI device control library.

7-2 Using Setup Modules and Forms with the ScriptPrinter

Example 7-1: Sample ANSI Setup Modules, A4_Page.txt

```
<ESC>[2 j
```

Example 7-2 is a PostScript setup module that prints the word "confidential" at the top and the bottom of each page of your print job.

Example 7-2: Sample PostScript Setup Modules, confidential.txt

```
/my-space matrix currentmatrix def
/old-showpage /showpage load def  % Copy the old
/showpage {
    gsave
    my-space setmatrix
    /Times-Roman findfont  % this is the overlay
    14 scalefont
    setfont
    252 756 moveto
    (confidential) show
    252 22 moveto
    (confidential) show % down to here
    grestore

    old-showpage
} def
```

7.1.2 Creating a Device Control Library

Before you can use a setup module, you must create a device control library, and put the module in the library. Use a different library for setup modules of each data syntax. You cannot mix ANSI and PostScript modules in the same library. You can put PostScript modules in the standard device control library, or create one or more new libraries for them.

An ANSI setup module and a PostScript setup module that perform the same function can have the same name, as you put them in separate libraries.

To create a library for setup modules, use the following command:

```
$ LIBRARY/CREATE/TEXT SYS$LIBRARY:libraryname.TLB
```

Using Setup Modules and Forms with the ScriptPrinter 7-3

where:

libraryname is the name you give to the library.

The following command creates a nem PostScript device control library, PS1, for your customized PostScript setup modules:

```
$ LIBRARY/CREATE/TEXT SYS$LIBRARY:PS1.TLB
```

Creating libraries requires priviledges. For more information, refer to VAX/VMS Librarian Utility Manual

7.1.3 Putting Setup Modules in Libraries

To insert a setup moduleinto a device control library, use the following command:

```
$ LBRARY/INSERT SYS$LIBRARY:libraryname.TLB modulename
```

where:

libraryname is the name of the library.

modulename is the name of the setup module.

The followingcommand puts the PostScript setup module,confidential.txt, into your new PostScript device control library, PS1.TLB:

```
$ LIBRARY/INSERT SYS$LIBRARY:PS1.TLB confidential.txt
```

This command puts the ANSI setup module,a4_page.txt, into a custom ANSI device control library, ANSI1.TLB

7.1.4 Specifying a Library Search List

If you have several libraries of setup modules, you can set up a search order for the libraries based on the data syntax of your print job. This means that the software skips libraries whose setup modules use the wrong data syntax for your job. PostScript print jobs search only PostScrpt libraries. ANSI print jobs look in ANSI and PostScript libraries.

7-4 Using Setup Modules and Forms with the ScriptPrinter

MLO-001473

6.1.2 Sheet Margins

The MARGINS option sets the margins of the sheet on which you are printing. This option takes four values. The values set the top, bottom, left, and right margins for a sheet. The numbers are interpreted as printer's points. (There are 72 points to an inch.)

You always specify the values for sheet margins in the order top, bottom, left, and right, because the sheet margins are independent of whether the page is portrait or landscape. Positive values start at the edge of the page and move toward the center; negative values start at the edge of the page and move away from the center of the page.

When you print using a layup definition file that uses the MARGINS option, the page is scaled to fit on the area of the sheet that is left when margins are subtracted. (See Figure 6–3) You can use negative margin values to move the page on the sheet without scaling down the size of the page. (See Example 4 in Section 6.2.)

If you specify the MARGINS option but do not specify any values, layup uses a value of 36 for all four margins. If you specify NOMARGINS, the translator uses the value 0 for all four margins.

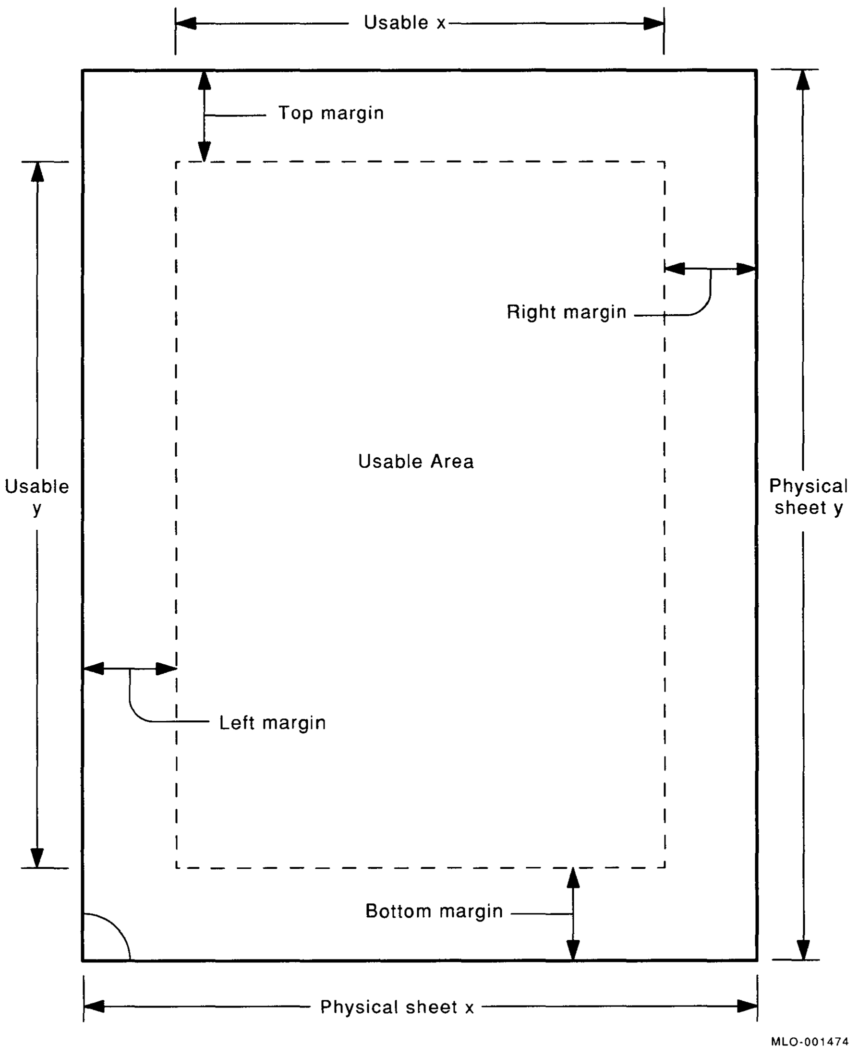
The default margin setting is a 36-point margin (a half inch) on all sides.

Example

The following example lets you create a wide left margin. It sets the top and bottom margins to 10 points, the left margin to 60 points, and the right margin to 10 points.

```
Margins = 10, 10, 60, 10
```

Figure 6–3: Usable Sheet Area with Margins Option



6.1.3 Margin Alternation

The ALTERNATE option lets you exchange a pair of margins on alternate sheets. This option is useful to prepare a job for double-sided copying.

The following list describes the values that can be supplied to the ALTERNATE option.

Value	Result
LEFT	Left and right margins alternate
RIGHT	Left and right margins alternate
none	Left and right margins alternate
TOP	Top and bottom margins alternate
BOTTOM	Top and bottom margins alternate

If you specify NOALTERNATE or do not specify this option, neither pair of margins alternates.

6.1.4 Pages per Sheet

The NUMBER_UP parameter to the PRINT/PARAMETER command determines the maximum number of pages that you can print for each sheet. The PAGESPERSHEET option in a layup definition file specifies how many pages are actually printed on a sheet.

Because NUMBER_UP determines the number of page spots, it also determines the layout of the page spots on the sheet. PAGESPERSHEET determines only the range of page spots to use, not their layout.

You must supply a value to PAGESPERSHEET, and the value must be less than the value supplied to NUMBER_UP. You cannot specify NOPAGESPERSHEET.

The default is to use all the page spots on a sheet.

Example

Assume you have a layup definition file called SIXPAGES.LUP that contains the following line:

```
PagesPerSheet = 6
```

You print a file using the following command:

```
$ PRINT/PARAMETER=(LAYUP_DEFINITION=SIXPAGES, NUMBER_UP=8) filename.typ
```

Even though this command provides eight page spots for each sheet, the job prints with six pages on a sheet leaving the other two page spots blank.

6.1.5 First Page

The FIRSTPAGE option specifies the first page spot on a sheet where a page will be printed.

You must specify a value to FIRSTPAGE. The value is the number of the page spot on which to print the first page. Page spots are numbered starting at 1. The value supplied to FIRSTPAGE must be less than the value supplied to NUMBER_UP. You cannot specify NOFIRSTPAGE.

The default is to use the first page spot on a sheet.

The FIRSTPAGE option affects only the first sheet of the print job. All other sheets begin printing at page spot 1.

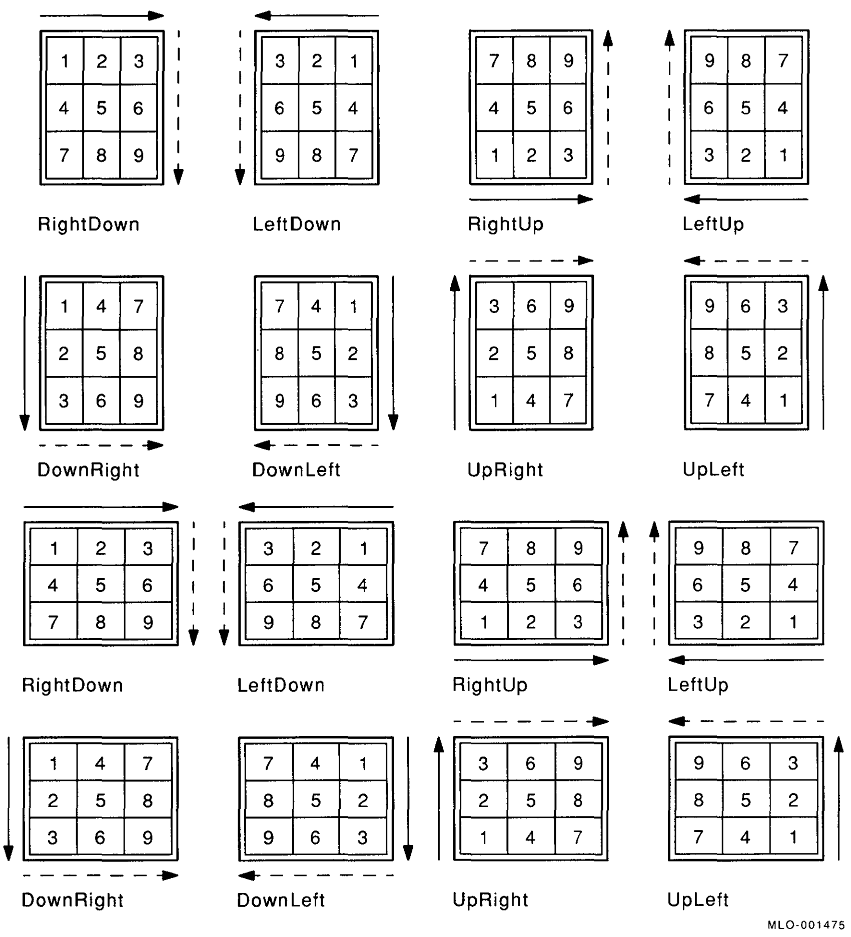
6.1.6 Page Order

The PAGEORDER option sets the order in which the pages appear on the sheet. This can be thought of as the reading order of the pages. (See Figure 6–4.)

You must supply a value to PAGEORDER. The following list describes the eight values that can be used. The default page order is RightDown.

Value	Description
RightDown	Pages are ordered left to right, top row first, then moving down
RightUp	Pages are ordered left to right, bottom row first, then moving up
LeftDown	Pages are ordered right to left, top row first, then moving down
LeftUp	Pages are ordered right to left, bottom row first, then moving up
DownRight	Pages are ordered top to bottom, left column first, then moving right
UpRight	Pages are ordered bottom to top, left column first, then moving right
DownLeft	Pages are ordered top to bottom, right column first, then moving left
UpLeft	Pages are ordered bottom to top, right column first, then moving left

Figure 6–4: Examples of Page Order Option



6.1.7 Page Grid

Instead of using the NUMBER_UP parameter to specify the maximum number of page spots on a sheet, you can use the GRID option. The GRID option lets you specify the number of page spots on a sheet in columns and rows.

GRID takes two values. The first value is the number of columns in the grid, the second is the number of rows. The product of the two values cannot exceed 100.

The grid is interpreted in relation to the pages, not the sheet. As with NUMBER_UP, the orientation of the pages is independent of the sheet orientation. This means that you may print portrait-shaped pages, but hold the sheet in landscape to read them.

Example

This example specifies that the pages will print 6-up, in a grid of two columns across and three rows down.

```
Grid = 2, 3
```

6.2 Sample Layup Definition Files

The first three layup definition files in this section are included as part of the ScriptPrinter software. They are already located in the directory LPS$LAYUP.

1. The following sample layup definition file is for a print job that will be printed one page to a sheet. It sets a wide left margin, so the pages can be punched. It does not alternate the margins, and it turns off borders.

   ```
   ! LPS$SingleHoles.LUP specifies a larger left margin to allow for
   ! hole punching.  This file is for single sided printing.

   no borders
   margins = 19, 19, 60, 19
   no alternate
   ```

2. The following sample layup definition file is for a print job that is going to be copied double-sided and then punched. It alternates the larger margin. It turns off the borders, as it is intended for jobs that are printed one page to a sheet.

   ```
   ! LPS$DoubleHoles.lup specifies a larger left margin to allow for
   ! hole punching. This file is for double sided printing.

   no borders
   margins = 19, 19, 60, 19
   alternate = left
   ```

3. The following sample layup definition file is for jobs that are printed with more than one page to a sheet, for example, a job printed with NUMBER_UP=6. This file uses borders to differentiate among pages. Again, it leaves a wide left margin but does not alternate margins.

   ```
   ! LPS$NUP.LUP specifies a variation for 2 up printing. A larger
   ! left margin is specified to allow for hole punching. This file
   ! is for single sided printing.

   borders
   margins = 19, 19, 60, 19
   ```

4. The following sample layup definition file is for jobs that print one page per sheet. It maintains the full size of the page image but allows a wide margin for hole-punching. It does this by using a negative right margin. The negative right margin moves some of the page image off the physical sheet, so this layup definition file is only good for pages that have page margins.

   ```
   ! LPS$ShiftForHoles.LUP specifies a wide left margin for hole-
   ! punching without scaling down the size of the page image.  The
   ! negative margin means some of the page image is off the physical
   ! sheet.  It is meant for printing one page per sheet.

   no borders
   margins = 0, 0, 60, -60
   ```

6.3 Error Messages

Errors in layup definition files are returned with the identification BADLAYDEF. To display the error messages on your terminal, use the /NOTIFY qualifier to the PRINT command. To print or keep them, use the MESSAGES parameter to PRINT/PARAMETER. Error messages use the following format:

```
%CPS_W_BADLAYDEF, <condition> on line <line-number>
in layup definition
```

The <condition> describes the problem and <line-number> is the line number in the layup definition file on which the error occurred. See Section 9.2.1 for the text and explanation of the messages generated by errors in a layup definition file.

Chapter 7

Using Setup Modules and Forms with the ScriptPrinter

As with other printers, you can use forms and setup modules on the ScriptPrinter. You can associate forms and setup modules with a queue or use them with the /FORM and /SETUP qualifiers to the PRINT command. Typically, the ScriptPrinter user includes /FORM on the PRINT command and associates the form with the job rather than the queue.

This chapter provides information about using setup modules and forms with the ScriptPrinter. It explains how you can use forms to include setup files in your print job.

7.1 Using Setup Modules

Setup modules can be included with your print job using the /SETUP qualifier to the PRINT command or in a form.

ANSI print jobs can use either ANSI or POSTSCRIPT setup modules. POSTSCRIPT, ReGIS, or Tektronix 4010/4014 print jobs must use POSTSCRIPT setup modules.

You need to do the following to use setup modules with the ScriptPrinter:

- Design your setup module, using a text editor.
- Create a device control library and add the library to SYS$LIBRARY (see Section 7.1.2).
- Put your setup module in the new library (see Section 7.1.3).

- Add the new library to a library search list (see Section 7.1.4).
- Stop and restart the queue to include your new library. (If you add a setup module to an existing library, you do not need to stop and restart the queue.)
- Print using the PRINT/SETUP command or use the DEFINE/FORM/SETUP command to include the setup module in a form.

  To include your ANSI setup module in a print job using the /SETUP qualifier, use the following command:

  ```
  $ PRINT/QUEUE=queue-name/SETUP=your-module-name-
    /PARAMETER=(DATA_TYPE=ansi) print-job.txt
  ```

  To include the same setup module in a form for your print job, use these commands:

  ```
  $ DEFINE/FORM/STOCK=DEFAULT/SETUP=your-module-name -
    form-name1 form-number1

  $ PRINT/QUEUE=queue-name/FORM=form-name1-
    /PARAMETER=(DATA_TYPE=ansi) print-job.txt
  ```

  For this example to work, the setup module, *your-module-name*, must be in a device control library that is listed in a library search list.

7.1.1 Sample Setup Modules

This section shows an example of an ANSI and a POSTSCRIPT setup module. The forms described in Section 7.3.3 include these setup modules.

Example 7–1 uses an ANSI escape sequence, as documented in the *PostScript Translators Reference Manual*, to set the margins for A4-size paper.

See Section 7.1.3 for information on how to put the setup module in an ANSI device control library.

Example 7–1: Sample ANSI Setup Module, A4_Page.txt

```
<ESC>[2 J
```

Example 7–2 is a POSTSCRIPT setup module that prints the word "confidential" at the top and the bottom of each page of your print job.

Example 7–2: Sample POSTSCRIPT Setup Module, confidential.txt

```
/my-space matrix currentmatrix def
/old-showpage /showpage load def    % Copy the old definition.
/showpage {
    gsave
    my-space setmatrix
    /Times-Roman findfont % this is the overlay
    14 scalefont
    setfont
    252 756 moveto
    (confidential) show
    252 22 moveto
    (confidential) show  % down to here
    grestore

    old-showpage
} def
```

7.1.2 Creating a Device Control Library

Before you can use a setup module, you must create a device control library, and put the module in the library. Use a different library for setup modules of each data syntax. You cannot mix ANSI and POSTSCRIPT modules in the same library. You can put POSTSCRIPT modules in the standard device control library, or create one or more new libraries for them.

An ANSI setup module and a POSTSCRIPT setup module that perform the same function can have the same name, as you put them in separate libraries.

To create a library for setup modules, use the following command:

```
$ LIBRARY/CREATE/TEXT SYS$LIBRARY:libraryname.TLB
```

where:

libraryname	is the name you give to the library.

The following command creates a new POSTSCRIPT device control library, PS1, for your customized POSTSCRIPT setup modules:

```
$ LIBRARY/CREATE/TEXT SYS$LIBRARY:PS1.TLB
```

Creating libraries requires privileges. For more information, refer to *VMS Librarian Utility Manual.*

7.1.3 Putting Setup Modules in Libraries

To insert a setup module into a device control library, use the following command:

```
$ LIBRARY/INSERT SYS$LIBRARY:libraryname.TLB modulename
```

where:

libraryname	is the name of the library.
modulename	is the name of the setup module.

The following command puts the POSTSCRIPT setup module, confidential.txt, into your new POSTSCRIPT device control library, PS1.TLB:

```
$ LIBRARY/INSERT SYS$LIBRARY:PS1.TLB confidential.txt
```

This command puts the ANSI setup module, a4_page.txt, into a custom ANSI device control library, ANSI1.TLB:

```
$ LIBRARY/INSERT SYS$LIBRARY:ANSI1.TLB a4_page.txt
```

Adding modules to libraries requires privileges. For more information, refer to *VMS Librarian Utility Manual.*

7.1.4 Specifying a Library Search List

If you have several libraries of setup modules, you can set up a search order for the libraries based on the data syntax of your print job. This means that the software skips libraries whose setup modules use the wrong data syntax for your job. POSTSCRIPT print jobs search only POSTSCRIPT libraries. ANSI print jobs look in ANSI and POSTSCRIPT libraries.

NOTE

> When you create a library search list order, you must always include the supplied ScriptPrinter device control library first in the list.

At installation, the INITIALIZE/QUEUE command in CPS$STARTUP.COM includes the qualifier /LIBRARY=CPS$DEVCTL. To specify a library search order, edit the INITIALIZE/QUEUE command in CPS$STARTUP.COM, as shown here:

```
$ INITIALIZE/QUEUE-
    .
    .
    .
/LIBRARY=logical-name-
    .
    .
    .
```

where:

logical-name	is a logical name that expands to a list of library names. All the library names in the list must be in SYS$LIBRARY and have the file type TLB.

Add the definition of the logical name to be expanded to CPS$STARTUP.COM. The command to define the logical name is shown here:

```
$ DEFINE logical-name search-list
```

where:

logical-name	is the name that you assign to the search-list and that you use with the INITIALIZE/QUEUE/LIBRARY command.
search-list	is the list of libraries to be searched. For each library that includes the /DATA_TYPE qualifier, enclose the library name and the qualifier in quotation marks. If you do not use the /DATA_TYPE qualifier, you do not need to enclose the library name in quotation marks. The default data type for a library is POSTSCRIPT.

NOTE

> You must stop and start a queue to use a new definition of the logical device control library. If you only change the contents of a library, you do not need to stop and start the queue.

In this Example 7–3, there are three POSTSCRIPT libraries, named CPS$DEVCLT, PS1 and PS2, and one ANSI library, named ANSI1. CPS$DEVCLT is the standard POSTSCRIPT device control library. PS1 and PS2 contain POSTSCRIPT setup modules. ANSI1 contains ANSI setup modules. The example creates the search order CPS$DEVCTL, PS1, ANSI1, PS2.

Example 7–3: Creating a Search Order for Device Control Libraries

```
$ DEFINE ALL_DEVCTL-
         CPS$DEVCTL,-
         "PS1/DATA_TYPE=POST",-
         "ANSI1/DATA_TYPE=ANSI",-
         PS2
$ INITIALIZE/QUEUE-
            /LIBRARY=ALL_DEVCTL-
    .
    .
    .
```

Using the libraries and search order in Example 7–3, all print jobs would look first in the standard device control library for the required setup modules and for any setup modules that had been explicitly specified with /SETUP or /FORM. All print jobs would then look in PS1 for any specified setup modules not found in the device control library. ANSI jobs would then look for ANSI setup modules in ANSI1. All other jobs would skip the ANSI library and look for POSTSCRIPT modules in PS2.

If you have an ANSI module and a POSTSCRIPT module that perform the same function and have the same name, determine which one you want as the default for ANSI jobs. In Example 7–3, you would place the POSTSCRIPT module in PS1 to use a POSTSCRIPT default for ANSI jobs, or in PS2 to use the ANSI module as the ANSI job default.

7.2 Using Forms

In general, ScriptPrinter users include forms in a print job by using the /FORM qualifier to the PRINT command.

Forms can be used to associate colored paper or transparencies with a queue. You can also use forms that include setup modules, which provide special details usually provided with preprinted forms, such as logos, letterheads, and electronic forms.

On DIGITAL's POSTSCRIPT printers, some of the typical uses of forms can be accomplished through other means. For example, you can set the page size and orientation by using parameters to the VMS PRINT command, by associating print parameters with a queue, or by using POSTSCRIPT device control library modules.

To use setup modules with the ScriptPrinter, do the following:

- Create any setup module that you want to include in a form. Put the setup modules in the appropriate library.
- Use the DEFINE/FORM command to create the form.
- Use the PRINT/FORM command to include the form in a print job.

You can associate the form with a queue, by creating a new execution queue and specfying the new form name using the /FORM qualifier.

7.3 Defining Forms

Use the DEFINE/FORM command to assign a name and number to a form to be used on the ScriptPrinter. The DEFINE/FORM command requires OPER (operator) privileges.

The format of the DEFINE/FORM command is:

```
$ DEFINE/FORM/STOCK=default form-name form-number
```

where:

form-name	is a 1-to 31-character name assigned to the form being defined. The name must contain at least one nonnumeric character. A default name of DEFAULT is assigned when the system is started.

form-number	is a number from 1 to 127 assigned to correspond to form-name. The DEFAULT form is assigned the number zero.

Use the /SETUP qualifier to DEFINE/FORM to include a setup file in a form (see Section 7.3.3).

NOTE

For earlier detection of spelling errors in device control library modules, define a form to include a setup module rather than using the /SETUP qualifier to the PRINT command. The /FORM qualifier is verified when you enter the PRINT command. /SETUP is verified at print time.

7.3.1 Supported FORM Qualifiers

In creating forms for the ScriptPrinter, you can only use the following DEFINE/FORM qualifiers:

- /DESCRIPTION=*string*
  Use /DESCRIPTION to define the form type more specifically.
- /SETUP=*module name*
  ANSI print jobs can use forms that include POSTSCRIPT or ANSI setup modules. All other print jobs can only use POSTSCRIPT setup modules.
- /STOCK=*type of paper stock*
  The /STOCK qualifier to the DEFINE/FORM command can have any value, transparency or pink paper, for example. However, the /STOCK qualifier for the print job and the queue must match for the job to print. Only use a differenct value for /STOCK when your form requires a different type of paper or media. If your job requires special stock, define a form that does not use the default stock. Then, your print job waits in the queue until the stock defined for the queue is set to match the stock defined in your form.

7.3.2 Unsupported FORM Qualifiers

ScriptPrinters ignore the following FORM qualifiers:

- /LENGTH
- /MARGIN
- /PAGE_SETUP
- /SHEET_FEED
- /TRUNCATE
- /WIDTH
- /WRAP

7.3.3 Sample ScriptPrinter Forms

ScriptPrinters ignore the page setup qualifiers to DEFINE/FORM, therefore the only way for a form to change the page setup is to include a setup module. Forms for ANSI print jobs can use either ANSI or POSTSCRIPT setup modules. Forms for POSTSCRIPT, ReGIS, and Tektronix 4010/4014 print jobs must use POSTSCRIPT setup modules.

Example 7–4 is a sample form that uses an ANSI setup module. The setup module, A4_page.txt, is shown in Example 7–1 in Section 7.1.1.

Example 7–4: Sample Definition of an ANSI Form

```
DEFINE /FORM A4PAPER 100
       /DESCRIPTION="set margins for 8.27 x 11.29 paper "
       /STOCK=DEFAULT
       /SETUP=A4_PAGE
```

Example 7–5 shows a sample form that uses a POSTSCRIPT setup module. The setup module, confidential.txt, is shown in Example 7–2 in Section 7.1.1.

Example 7–5: Sample of a POSTSCRIPT Form

```
DEFINE/FORM confidential 101-
      /DESCRIPTION="prints the word confidential on each page "-
      /STOCK=DEFAULT-
      /SETUP=confidential
```

7.3.4 Deleting a Form

Use the DELETE/FORM command to delete a forms definition from the systems forms table. The DELETE/FORM command requires OPER (operator) privileges. The format of this command is as follows:

```
$ DELETE/FORM form-name
```

where:

form-name	is the name assigned to the form you are using

Before you use the DELETE/FORM command, execute the SHOW QUEUE/FULL/ALL command. Make sure no outstanding references exist to the form you want to delete. References to a form can be as an attribute of an active print queue or as a qualifier to a print request. If you try to delete a form with outstanding references, you are notified of the condition, and the form is not deleted.

Chapter 8

Submitting Print Requests

To print a file on a ScriptPrinter, you use the VMS PRINT command. PRINT has the following syntax:

$ PRINT/qualifier file-spec[,...]

Most of the PRINT command qualifiers function with no change from VMS. Some, however, have either no effect or a different effect when applied to a ScriptPrinter.

This chapter contains several sections. Section 8.1 documents the PRINT command qualifiers that behave differently in a ScriptPrinter environment. For a complete list of PRINT command qualifiers, refer to the *VMS DCL Dictionary*. Section 8.2 documents the ScriptPrinter-specific parameters that are passed to the print symbiont by using the /PARAMETERS qualifier.

Additional sections describe the parameters defaulted by the print symbiont, the use of the SHOW QUEUE command to interrogate a ScriptPrinter print queue, and how to recover from a paper jam.

8.1 Print Command Qualifiers

The PRINT command qualifiers that function differently on the ScriptPrinter are described in this section. (Some commonly-used qualifiers that do not function differently are also documented here for convenience.) A qualifier may have varying results, depending on the type of data being processed. POSTSCRIPT data is passed directly to the POSTSCRIPT interpreter in the ScriptPrinter. Non-POSTSCRIPT data is passed to a data syntax translator for conversion to POSTSCRIPT format. The data syntax translator can be specified in the PRINT command or can be defined by the systemwide logical name for the default parameters for that queue (see Section 2.7). Translators for ANSI, ReGIS, and Tektronix 4010/4014 data syntaxes are supplied with the ScriptPrinter.

8.1.1 /COPIES

/COPIES gives you multiple copies of the file you are printing as described in the *VMS DCL Dictionary*. However, when you use the /COPIES=n qualifier on a ScriptPrinter, the software must translate your file and transmit it to the printer **n** times. This increases the load on the CPU on which the ScriptPrinter symbiont runs and can slow down your print job.

When you use the command PRINT/COPIES=n for POSTSCRIPT jobs, the POSTSCRIPT interpreter does not reset to its default job start state between each copy of the file in the print job. This causes each copy to inherit the POSTSCRIPT state of the previous copy. Files that fail to clean up the POSTSCRIPT state, when printed with the /COPIES qualifier, can incur POSTSCRIPT errors such as **dictfull**, **limitcheck**, or **VMerror**. Refer to the *LN03R ScriptPrinter Programmer's Supplement*, for a description of POSTSCRIPT programming techniques that help avoid this problem.

If uncollated output is acceptable, use the /PARAMETERS=SHEET_COUNT=n qualifier (see Section 8.2.9).

8.1.2 /DELETE

/DELETE determines whether the job controller deletes a file after printing. On the ScriptPrinter, with the DELETE qualifier, the job controller deletes the print files **regardless** of whether the print job completed successfully or not. For example, a job queued as follows aborts due to a syntax error. The job controller still deletes the user's print files.

```
$ PRINT/DELETE/PARAMETERS=(SHEET_SIZE=GREEN) file_name
```

8.1.3 /FEED

The FEED qualifier is ignored by the POSTSCRIPT interpreter and by the supplied data syntax translators.

8.1.4 /FORM

/FORM specifies attributes of the print job, such as size and/or type of paper. If the form stock type does not match the form's stock for the job's execution queue, the symbiont places the job in a pending state. The job is not released until either the PRINT command or the queue attributes are changed so that the two stock types match.

The ScriptPrinter supports only the /SETUP option of the DEFINE/FORM qualifier. For information on defining forms and setup modules, see Chapter 7.

Note that on a ScriptPrinter, some of the typical uses of forms are accomplished through other means. For example, to specify the orientation of a print job, you use the command PRINT/PARAMETERS=PAGE_ORIENTATION.

8.1.5 /HEADER

/HEADER is ignored by the POSTSCRIPT interpreter and by the supplied data syntax translators.

8.1.6 /JOB_COUNT

/JOB_COUNT gives you multiple copies of the job you are printing. A job consists of all files queued by a single PRINT command.

When you use the command PRINT/JOB_COUNT=n for POSTSCRIPT jobs, the POSTSCRIPT interpreter does not reset to its default job start state between each iteration of the print job. This causes each copy of the print job to inherit the POSTSCRIPT state of the previous copy of the print job. Print jobs that fail to clean up the POSTSCRIPT state, when printed with the /JOB_COUNT qualifier, can incur POSTSCRIPT errors such as **dictfull**, **limitcheck**, or **VMerror**. Refer to the *LN03R ScriptPrinter Programmer's Supplement*, for a description of POSTSCRIPT programming techniques that help avoid this problem.

/COPIES gives you multiple copies of each file, or if uncollated output is acceptable, use the /PARAMETERS=SHEET_COUNT=n qualifier, which uses resources more efficiently.

8.1.7 /NOTIFY

/NOTIFY causes you to be notified when the job starts and again when it finishes.

Errors occurring during the printing process, as well as messages sent back by POSTSCRIPT operators (such as **print**, **pstack**, **=**, and **==**), are also displayed on your terminal if /NOTIFY is included in the print request.

8.1.8 /PAGES

/PAGES is ignored in print requests for a ScriptPrinter. To achieve the same result, use the PAGE_LIMIT parameter in the PARAMETERS qualifier of the PRINT command.

8.1.9 /PASSALL

Although the default for the PRINT command is NOPASSALL, the print symbiont for the ScriptPrinter treats the data as PASSALL. /PASSALL specifies that the host system pass data to the printer without analyzing the file for control data, such as escape sequences.

8.1.10 /QUEUE

The default print output queue is SYS$PRINT. Use /QUEUE to direct your print job to a ScriptPrinter-specific print queue, unless SYS$PRINT has been established as the queue for that printer.

This qualifier acts in the same manner as described in the *VMS DCL Dictionary*.

8.1.11 /REMOTE

/REMOTE queues a print job on a remote node whose default print queue (SYS$PRINT) is serviced by a ScriptPrinter. The file to be printed must exist on the remote node. The file specification must contain the host's node name, in the following syntax:

```
$ PRINT/REMOTE nodename::filename
```

The file is queued on the default print queue (SYS$PRINT) of the other system and takes the default ScriptPrinter characteristics defined for that system. Not all qualifiers, including QUEUE or PARAMETERS, may be used with REMOTE.

This qualifier acts in the same manner as described in the *VMS DCL Dictionary*.

8.1.12 /RESTART

Normally, /RESTART restarts a job after a crash or after a STOP/QUEUE/REQUEUE command. It does this by enabling checkpointing. Since ScriptPrinters are POSTSCRIPT printers, it is impractical to reestablish the state in effect at the time of checkpoint. Therefore, use /RESTART with caution. POSTSCRIPT jobs that depend on passing POSTSCRIPT state from one file to the next do not produce the expected result.

Translated jobs and POSTSCRIPT jobs that do not depend on passing POSTSCRIPT state from one file to the next print as expected with /RESTART.

8.1.13 /SETUP

/SETUP sends modules from the device control library to the printer before the data for each file is printed. These modules may be encoded in POSTSCRIPT or in ANSI, if the file to be printed is an ANSI file. When necessary the symbiont directs the module through the ANSI translator. The standard POSTSCRIPT device control library for a ScriptPrinter queue is SYS$LIBRARY:CPS$DEVCTL.TLB. You may define other libraries to contain your POSTSCRIPT and ANSI setup modules, and then set up a library search list to make them available (see Sections 7.1.3 and 7.1.4).

Examples

The following example shows how to include the device control library module for the error handler in a print job.

```
$ PRINT/SETUP=LPS$ERRORHANDLER filename.typ
```

The following examples show the effect of the position of /SETUP in the command line. In this example, the setup module goes to the printer before each file:

```
$ PRINT/SETUP=modulename file1.typ,file2.typ,file3.typ
```

Here, the setup module affects only file2:

```
$ PRINT file1.typ,file2.typ/SETUP=modulename,file3.typ
```

For earlier detection of spelling errors in device control library modules, use forms that include setup modules. The /FORM qualifier is verified when you enter the PRINT command. /SETUP is verified at print time.

8.1.14 /SPACE

/SPACE is ignored by the POSTSCRIPT interpreter and by the supplied data syntax translators.

8.2 /PARAMETERS Qualifier

Use the /PARAMETERS qualifier with the PRINT command to pass printer-specific parameters to the print symbiont for processing. The following restrictions apply:

- All files submitted with the same PRINT command are considered to be one print job. Therefore, each file is subject to the parameters passed in the /PARAMETERS qualifier.
- Parameters are checked for syntax and value limits at printing time, **not when the PRINT command is first issued at the host**. When an error is detected by the print symbiont at print time, the job is aborted and an error message is generated. A syntax error generates a message on your screen regardless of the /NOTIFY qualifier.

The /PARAMETERS qualifier adheres to the following rules:

- Parameters consist of keywords and associated values.

- A maximum of eight parameters is allowed.[1] If you are using DQS, a maximum of seven parameters is allowed.
- If you specify two or more parameters, separate them by commas and enclose the set in parentheses.
- If the value associated with a parameter contains delimiters, such as commas or parentheses, enclose the value (or the entire parameter) in double quotation marks.

The following examples are all valid notations:

```
$ PRINT/PARAMETERS=SHEET_COUNT=4

$ PRINT/PARAMETERS=(SHEET_COUNT=4,DATA_TYPE=REGIS,MESSAGES)

$ PRINT/PARAMETERS=(SHEET_COUNT=4,"PAGE_LIMIT=(1,4)",MESSAGES)

$ PRINT/PARAMETERS=(SHEET_COUNT=4,PAGE_LIMIT="(1,4)",MESSAGES)
$ PRINT/PARAMETERS="SHEET_COUNT=4,PAGE_LIMIT=(1,4),MESSAGES"
```

In this version of the ScriptPrinter software, the /PARAMETERS qualifier supports the following parameters :

- DATA_TYPE=*data-type-name*
- LAYUP_DEFINITION=*layup-definition-filename*
- [NO]MESSAGES=*[([PRINT][,][KEEP])]*
- NUMBER_UP=*n*
- PAGE_ORIENTATION=*logical-orientation*
- PAGE_LIMIT=*([lower-limit,][upper-limit])*
- PAGE_SIZE=*logical-size*
- SHEET_COUNT=*number*
- SHEET_SIZE=*physical-size*

This version of the ScriptPrinter also supports the following qualifier with limits (see the description of the qualifier for specific information):

- OUTPUT_TRAY=*tray-name*

[1] A series of parameters enclosed in quotation marks is considered to be a single parameter.

8.2.1 DATA_TYPE

Format: DATA_TYPE=*data-type-name*

Determines if the data to be printed must be translated into the POSTSCRIPT language. It specifies which translator, if any, the print symbiont invokes.

If *data-type-name* is POSTSCRIPT, the data is sent to the ScriptPrinter without translation. Otherwise, the print symbiont calls the data syntax translator to convert the data before sending it to the ScriptPrinter. The translators support the data types shown in Table 8–1.

Without the DATA_TYPE parameter, the system uses the data type associated with the generic queue that is handling the print request. (Print queues for data types can be set up during or after software installation. See Sections 2.2.4 and 2.7.)

If a DATA_TYPE is not associated with the generic queue, the symbiont uses the DATA_TYPE associated with the execution queue.

If the DATA_TYPE parameter is not present, and no data type is associated with the print queues, the data type defaults to ANSI.

An invalid DATA_TYPE causes the print job to abort.

Table 8–1: Recognized Data Types

Data Type	Data Translation
ANSI	ANSI data converted by the ANSI translator (see *PostScript Translators Reference Manual*).
ANSI2	ANSI Level 2—ANSI subset for LA100/LA210 (currently treated as ANSI).
ASCII	Printing characters plus CR, LF, BS, HT, VT, and FF control characters (currently treated as ANSI).
LINE	Printing characters plus CR, LF, HT, and FF control characters (currently treated as ANSI).
POSTSCRIPT	POSTSCRIPT program data processed by the POSTSCRIPT interpreter without conversion.
PS	Same as POSTSCRIPT.
REGIS	ReGIS commands data converted by the ReGIS translator (see *PostScript Translators Reference Manual*).

Table 8–1 (Cont.): Recognized Data Types

Data Type	Data Translation
TEK4014	Tektronix 4010/4014 graphics commands data converted by the Tektronix 4010/4014 translator (see *PostScript Translators Reference Manual*).
TEXT	Printing characters plus CR and LF control characters (currently treated as ANSI).

You can abbreviate the data types. Simply use enough letters to make the abbreviation unique. Abbreviations for TEK4014 that worked in previous versions may conflict with the TEXT data type.

8.2.2 LAYUP_DEFINITION

Format: LAYUP_DEFINITION=*layup-definition-filename*

Calls a file that sets all of the layup features, except NUMBER_UP. The symbiont looks for the file in LPS$LAYUP:*layup-definition-filename.LUP*.

The file type for layup definition files is LUP. However, you do **not** use the file type on the command line. You can call a layup definition file on the print command line, or a layup definition file can be associated with a queue. See Chapter 6 for information about creating a layup definition file.

If you specify a layup definition file without specifying NUMBER_UP, the default is NUMBER_UP=1, even if NUMBER_UP is associated with the queue. Similarly, if you specify NUMBER_UP without specifying a layup definition file, your print job does not use a layup definition file, even if there is one associated with the queue. So, to use a layup definition file and NUMBER_UP together, you must specify both on the command line or associate both with a logical queue.

If you specify both a layup definition file and NUMBER_UP and the value for NUMBER_UP is 0, the job will print without the layup definition file and with a warning message. If the requested layup definition file name does not exist or is not accessible, the job aborts with an error message.

Example

This example uses the layup definition file LPS$HOLES.LUP. The print job uses the default NUMBER_UP=1, which means you are printing one page per sheet. (See Section 6.2 for examples of layup definition files.)

```
$ PRINT/PARAM=(LAYUP_DEFINITION=lps$holes) filename.typ
```

8.2.3 MESSAGES

Format: [NO]MESSAGES[=*([PRINT][,][KEEP])*]

Specifies the destination of messages and user data returned by PostScript. The default for the parameter is NOMESSAGES.

The following are the allowable combinations of parameter values:

- MESSAGES=PRINT — Specifies that messages are to be recorded on a job log page, printed before the job trailer. The message file is then deleted. With the MESSAGES=PRINT parameter, the ScriptPrinter prints up to 40 lines of message text on the job log page. To capture messages in excess of 40 lines, use MESSAGES=KEEP.
- MESSAGES — Same as MESSAGES=PRINT.
- MESSAGES=KEEP — Specifies that messages are to be recorded in the default file SYS$LOGIN:CPS$JOB_nnn.LOG, where "nnn" is the queue entry number assigned to that job.

  If you print your jobs from a remote node using DQS, the log file CPS$JOB_nnn.LOG is located in DQS$LOG_AREA:*system logical*. With the MESSAGES=KEEP parameter option, DQS returns the log file through VAX mail and then deletes the file.
- MESSAGES="(PRINT,KEEP)" — Specifies that messages are recorded on the job log page and in the default file.

NOTE

When the value associated with a parameter contains commas or parentheses, enclose the value or the entire parameter in double quotation marks. For examples, refer to Section 8.2.

- NOMESSAGES — Specifies that messages are not to be recorded. The message file is never created. NOMESSAGES is the default.

NOTE

Regardless of the MESSAGES option selected, with errors present, the trailer page contains the first two "interesting" messages (see Section 4.1.2).

8.2.4 NUMBER_UP

Format: NUMBER_UP=*n*

The NUMBER_UP parameter sets the maximum number of pages that can be printed on a sheet. Your job prints using the number set by NUMBER_UP, unless you use a layup definition file that contains the **PagesPerSheet** option or the **Grid** option. The default is to print one page on a sheet (NUMBER_UP=1), but you can set a default NUMBER_UP for a queue.

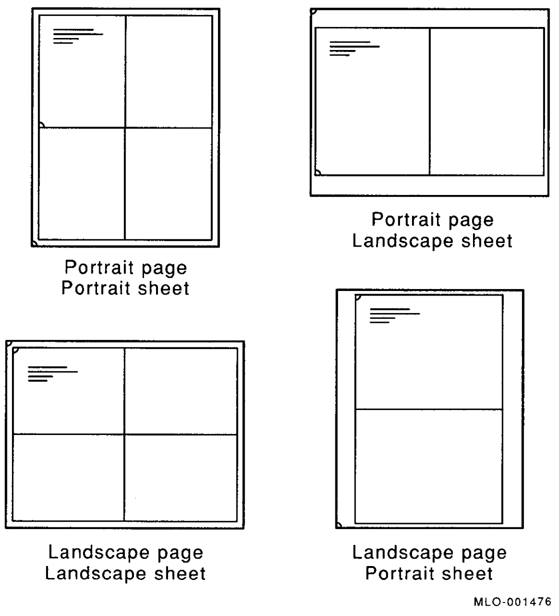
You can use any integer from 1 to 100 as a value for NUMBER_UP.

To disable layup, specify NUMBER_UP=0. This overrides any layup defaults that are set for a queue.

When you print using NUMBER_UP, the aspect ratio of the page is preserved. This means that, for example, a portrait-shaped page retains its portrait orientation. However, the page and sheet may have different orientations. For example, if you print portrait pages 2-up, the pages remain portrait-shaped, but you hold the sheet in landscape orientation to read them. If you specify that the pages are to print as landscape pages with 2-up, you would hold the sheet in portrait orientation to read them.

Figure 8–1 shows the four different cases of page and sheet orientation with NUMBER_UP. 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 8–1: Page Orientation with NUMBER_UP



8.2.4.1 NUMBER_UP Parameter with Paper Selection Parameters

NUMBER_UP sizes the page image according to the page size in effect at the time the print job is submitted.

Examples

1. This example prints the file eight pages to a sheet.

```
$ PRINT/PARAM=(NUMBER_UP=8) filename.typ
```

2. This example prints the file four pages to a sheet and uses the sample layup definition file LPS$SINGLEHOLES.LUP.

```
$ PRINT/PARAM=(LAYUP_DEF=LPS$SINGLEHOLES,NUMBER_UP=4) filename.typ
```

3. This example prints the file 2-up (two pages to a sheet) and gives the pages landscape orientation. Remember that the sheet and the pages do not always have the same orientation.

   ```
   PRINT/PARAM=(NUMBER_UP=2,PAGE_ORIENT=LANDSCAPE) filename.typ
   ```

8.2.5 OUTPUT_TRAY

Format: OUTPUT_TRAY=*tray-name*

Some POSTSCRIPT printers have multiple paper output trays. The OUTPUT_TRAY parameter selects a paper output tray for the print job. This parameter is not useful for the ScriptPrinter, since it has just one output tray that stacks paper face down. The acceptable entries for *tray-name* are TOP (the default) or SIDE, both of which specify the tray on the top of the ScriptPrinter.

NOTE

If you select OUTPUT_TRAY=FACE_UP, the job aborts as the ScriptPrinter only stacks paper face down.

8.2.6 PAGE_LIMIT[1]

Format: PAGE_LIMIT=*([lower-limit,][upper-limit])*

The PAGE_LIMIT parameter specifies the numbers of the first and last pages to be printed for the job. The maximum value you can specify is 10000. If you abbreviate PAGE_LIMIT, you must type at least PAGE_ L.

If this parameter is omitted, the default is to print the entire job. If *lower-limit* is omitted, printing starts on the first page of the job. If *upper-limit* is omitted, printing continues to the end of the job. If either the lower or upper limit is zero, the parameter generates an error. These limits apply to the entire job and not to each file of a multifile job. Therefore, if you intend to print a selected portion of a file, issue a separate print request for that file alone.

[1] This parameter was called SHEET_LIMIT in Version 1.1 of the ScriptPrinter.

You can omit the parentheses if you are specifying a value only for *upper-limit*. For example, PAGE_LIMIT=10 prints the first 10 sheets of a job. PAGE_LIMIT="(5,10)" prints pages 5 through 10 of a job. PAGE_LIMIT="(5,)" prints page 5 through the end of the job.

When you use the PAGE_LIMIT parameter, your print job aborts when the *upper-limit* is reached. This means that you should not use PAGE_LIMIT with the /COPIES qualifier, because the print job may abort before the second copy begins printing. If you want to print multiple copies of job for which you specify PAGE_LIMIT, use the SHEET_COUNT parameter. Flag, trailer, burst, and log pages are not included in the count.

NOTE

If you specify a PAGE_LIMIT and do not print to the end of the data, you will not get a file trailer page for the file that was cut off. The rest of the data for that file and data for any subsequent files for that job is ignored.

Examples

1. This example prints the first ten pages of a job:

```
$ PRINT/PARAM=(PAGE_LIMIT=10) filename.typ
```

2. This example prints pages 10 through 13 on one sheet of paper:

```
$ PRINT/PARAM=(NUMBER_UP=4,"PAGE_LIMIT=(10,13)") filename.typ
```

3. This example prints two uncollated copies of the first three pages of a job:

```
$ PRINT/PARAM=(SHEET_COUNT=2, PAGE_LIMIT=3) filename.typ
```

8.2.7 PAGE_ORIENTATION

Format: PAGE_ORIENTATION=*logical-orientation*

Specifies the orientation of the printed output on the page. If the DATA_TYPE is POSTSCRIPT, this parameter is ignored except for layup. The entries for *logical-orientation* are as follows:

- PORTRAIT, which specifies the lines of text to be parallel to the short side of the logical page.

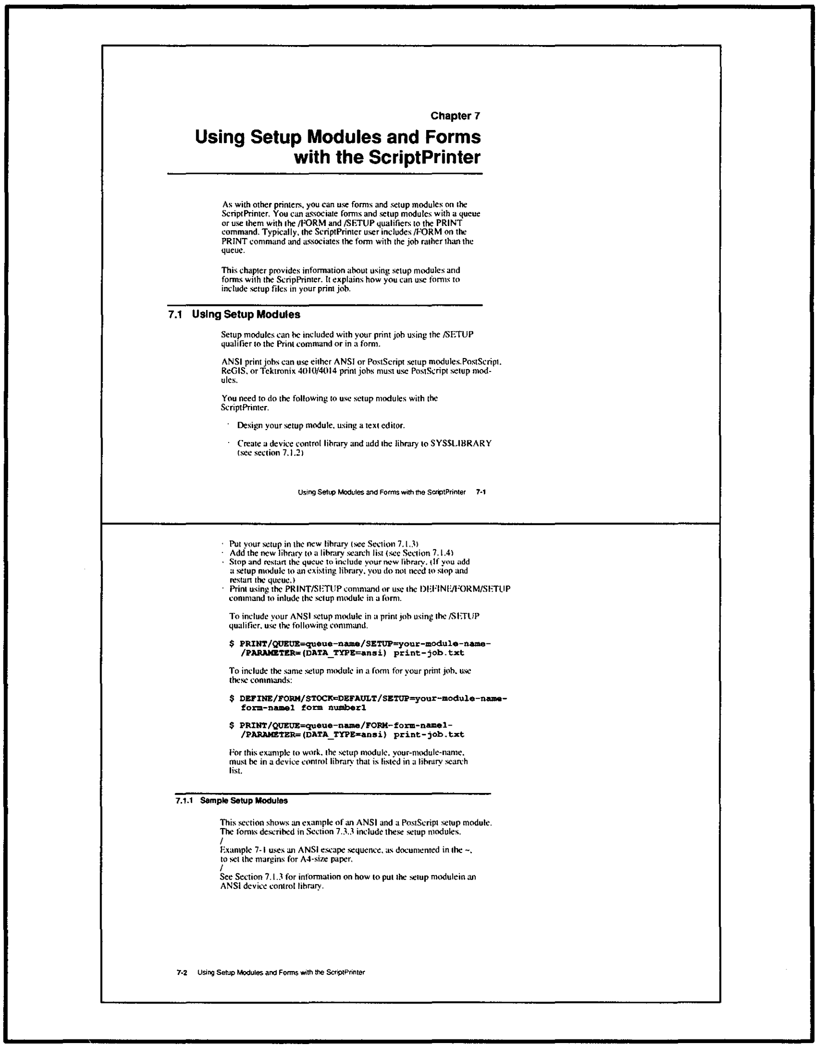
- LANDSCAPE, which specifies the lines of text to be parallel to the long side of the page. LANDSCAPE mode allows for 132-column format with ANSI files.

Refer to the *PostScript Translators Reference Manual* to determine the complete effect of this qualifier for a particular translator.

8.2.7.1 PAGE_ORIENTATION Parameter with NUMBER_UP Parameter

PAGE_ORIENTATION affects the page, not the sheet. When you print using NUMBER_UP, the page orientation and the sheet orientation are not always the same. For example, if you print portrait pages 2-up, the pages remain portrait-shaped, but you may hold the sheet in landscape orientation to read them. If you change the PAGE_ORIENTATION to *landscape*, the pages are printed as landscape pages, but you hold the sheet in portrait orientation to read them. (See Figure 8–2.)

Figure 8–2: Landscape Pages on a Portrait Sheet



Chapter 7

Using Setup Modules and Forms with the ScriptPrinter

As with other printers, you can use forms and setup modules on the ScriptPrinter. You can associate forms and setup modules with a queue or use them with the /FORM and /SETUP qualifiers to the PRINT command. Typically, the ScriptPrinter user includes /FORM on the PRINT command and associates the form with the job rather than the queue.

This chapter provides information about using setup modules and forms with the ScripPrinter. It explains how you can use forms to include setup files in your print job.

7.1 Using Setup Modules

Setup modules can be included with your print job using the /SETUP qualifier to the Print command or in a form.

ANSI print jobs can use either ANSI or PostScript setup modules.PostScript. ReGIS, or Tektronix 4010/4014 print jobs must use PostScript setup modules.

You need to do the following to use setup modules with the ScriptPrinter.

- Design your setup module, using a text editor.
- Create a device control library and add the library to SYS$LIBRARY (see section 7.1.2)

Using Setup Modules and Forms with the ScriptPrinter 7-1

- Put your setup in the new library (see Section 7.1.3)
- Add the new library to a library search list (see Section 7.1.4)
- Stop and restart the queue to include your new library. (If you add a setup module to an existing library, you do not need to stop and restart the queue.)
- Print using the PRINT/SETUP command or use the DEFINE/FORM/SETUP command to inlude the setup module in a form.

To include your ANSI setup module in a print job using the /SETUP qualifier, use the following command.

```
$ PRINT/QUEUE=queue-name/SETUP=your-module-name-
   /PARAMETER=(DATA_TYPE=ansi) print-job.txt
```

To include the same setup module in a form for your print job, use these commands:

```
$ DEFINE/FORM/STOCK=DEFAULT/SETUP=your-module-name-
   form-name1 form number1

$ PRINT/QUEUE=queue-name/FORM-form-name1-
   /PARAMETER=(DATA_TYPE=ansi) print-job.txt
```

For this example to work, the setup module, your-module-name, must be in a device control library that is listed in a library search list.

7.1.1 Sample Setup Modules

This section shows an example of an ANSI and a PostScript setup module. The forms described in Section 7.3.3 include these setup modules.
/
Example 7-1 uses an ANSI escape sequence, as documented in the ~, to set the margins for A4-size paper.
/
See Section 7.1.3 for information on how to put the setup modulein an ANSI device control library.

7-2 Using Setup Modules and Forms with the ScriptPrinter

MLO-001477

8.2.8 PAGE_SIZE

Format: PAGE_SIZE=*logical-size*

The PAGE_SIZE parameter specifies the size of the logical pages being printed. If you abbreviate PAGE_SIZE, you must type at least PAGE_S.

If you specify a page size that the ScriptPrinter does not implement, your job may not print, and, if you have specified /NOTIFY, an error message is displayed on your terminal.

The acceptable entries for *logical-size* are shown in Table 8–2. Although the LN03R ScriptPrinter only supports two physical sheet sizes (A and A4), the layup feature supports the logical page sizes described in Table 8–2 for the LN03R.

Table 8–2: Page Sizes

Entry	Abbreviation	Page Size
LETTER or A	LET[1]	8.5 x 11 in. (216 x 279 mm)
LEDGER or B	LED[1]	11 x 17 in. (279 x 432 mm)
LEGAL	LEG[1]	8.5 x 14 in. (216 x 356 mm)
EXECUTIVE	E	7.5 x 10.5 in. (191 x 267 mm)
A5		5.8 x 8.3 in. (148 x 210 mm)
A4		8.3 x 11.7 in.(210 x 297 mm)
A3		11.7 x 16.5 in. (297 x 420 mm)
B5		7.2 x 10.1 in.(176 x 250 mm)
B4		10.1 x 14.3 in.(250 x 353 mm)

[1]If you specify L or LE as the argument to PAGE_SIZE, the printer uses LEGAL paper.

If you do not specify this parameter, PAGE_SIZE defaults to SHEET_SIZE.

If you also do not specify SHEET_SIZE, PAGE_SIZE defaults to the value specified for the queue.

If PAGE_SIZE is not specified for the queue, it defaults to LETTER (or A) or A4, depending on the switch setting on the back of the printer at power up.

When DATA_TYPE = POSTSCRIPT, the PAGE_SIZE parameter is ignored, unless NUMBER_UP is in effect. When NUMBER_UP is in effect, PostScript files use the PAGE_SIZE parameter to determine the size of the logical page. When there is no NUMBER_UP parameter, the PostScript file determines the size of the logical page.

When the DATA_TYPE is ANSI or ASCII, the only recognized page sizes are A, A4, B, and LEGAL. Any other value is treated as A.

8.2.9 SHEET_COUNT

Format: SHEET_COUNT=*number*

Specifies the number of times each sheet is to be printed. This parameter produces uncollated output (copies of the first sheet, followed by copies of the second sheet, and so on). If you abbreviate SHEET_COUNT, you must type at least SHEET_C.

The default is SHEET_COUNT=1, and the range of values for *number* is 1 through 10000.

For collated output, use the COPIES or JOB_COUNT qualifier.

8.2.10 SHEET_SIZE

Format: SHEET_SIZE=*physical-size*

On PostScript printers with multiple paper input trays, the SHEET_SIZE parameter specifies the physical size of the sheets being printed. Since the LN03R has only one paper input tray, you can specify any sheet size but only A and A4 are useful. SHEET_SIZE defaults to PAGE_SIZE. If you do not specify PAGE_SIZE, SHEET_SIZE defaults to the size specified by the paper size switch on the back of the printer.

8.3 Multiple Files and Multiple Copies

The PostScript state is initialized at the beginning of each print job. Regardless of the number of files specified in a job, the PostScript state is not reinitialized until the next job begins.

If you are printing multiple files or if you are printing multiple collated copies with the PRINT/COPIES command, the POSTSCRIPT state is initialized at the beginning of the first file of the job. The next file starts with the POSTSCRIPT state that existed at the end of the first file. The third file starts with the ending state of the second, and so forth for the rest of the job. If the POSTSCRIPT program does not leave the ending state the same as the beginning state, subsequent files may not process correctly.

If the ending state causes problems with multifile jobs, print each file as a separate job.

If the ending state causes problems when printing multiple copies of a file, use the /PARAMETERS=SHEET_COUNT=n qualifier or print each copy as a separate job. With /PARAMETERS=SHEET_COUNT, the copies are not collated.

When you print multiple copies of your print job with the PRINT/JOB_COUNT command, the POSTSCRIPT state is not reset between each repetition of the print job. Each instance of the print job inherits the POSTSCRIPT state from the previous print job.

Print jobs that fail to clean up the POSTSCRIPT state can cause POSTSCRIPT errors, such as **dictfull**, **limitcheck**, or **VMerror**, when used with the /JOB_COUNT qualifier. Refer to *LN03R ScriptPrinter Programmer's Supplement* for POSTSCRIPT programming techniques to help avoid this problem.

8.4 Using the SHOW QUEUE Command

The SHOW QUEUE command displays the status of each job on the LN03R print queue. Interpret the status of each job as follows:

Status	**Meaning**
Pending	The job is waiting to become the current job on the queue.
Starting	The job is in the process of making a connection to the printer.
Printing	The job is on the queue and is either printing or is waiting to be printed on the printer.
Aborting	The job is being aborted by a DELETE/ENTRY or STOP/QUEUE/ABORT command.

Occasionally you encounter a print queue status of DEVICE UNAVAILABLE. This is normal. When the symbiont synchronizes with the printer during execution of a print job, the symbiont sets the print queue status to DEVICE UNAVAILABLE.

Repeating the SHOW QUEUE command should report a status other than DEVICE UNAVAILABLE as the synchronization period is quite short.

If the DEVICE UNAVAILABLE status persists for more than several seconds, check the power to the ScriptPrinter and/or the serial communication cable connections between the ScriptPrinter and the host.

NOTE

When reconnecting the ScriptPrinter to the host, you must turn off the printer. Turn on the printer after you complete the connection to the host. Turning the printer off and on ensures proper synchronization between the print symbiont and the ScriptPrinter.

8.5 Recovering from a Paper Jam

When the printer detects a paper jam, it places the associated print job on HOLD in the print queue. You can release the print job on hold and recover from paper jams with the SET ENTRY (SET QUEUE/ENTRY) command. The ScriptPrinter restarts the printing of the released job from the beginning.

Alternatively, you can remove a print job from the queue with the following DCL command:

```
$ DELETE/ENTRY=nnnn queue_name
```

8.6 Using the ScriptPrinter with DQS

You must set up the host node of the ScriptPrinter with the form CPS$DEFAULT for the host system to act as a Distributed Queuing Services (DQS) server node. DQS enables you to print files on output devices that are attached to remote VMS nodes in your network.

For information on using and managing DQS, refer to the *VAX Distributed Queuing Service User's Guide* and the *VAX Distributed Queuing Service Management Guide.*

Chapter 9

ScriptPrinter System Messages

The system messages in this chapter are issued by the ScriptPrinter software. These are identified by a facility code of CPS or LPS. Refer to the *VAX/VMS System Messages and Recovery Procedures Reference Manual* for descriptions of other messages issued by VAX/VMS.

9.1 System Message Overview

ScriptPrinter system messages are classified into two categories, based on their destination: the host system operator or to the user.

To receive user messages, use the NOTIFY qualifier with the PRINT command. The messages are in response to either your print request or a system condition that affects your print request.

To receive operator messages, you need OPERATOR privileges and must execute the REPLY/ENABLE=PRINTER command. You then receive the operator messages that apply to your host system.

NOTE

To ensure that you receive operator messages, make sure OPCOM is running. If not, start OPCOM by issuing the command:

```
$ @SYS$SYSTEM:STARTUP OPCOM
```

Start OPCOM before executing the REPLY/ENABLE=PRINTER command.

9.1.1 Message Format

ScriptPrinter system messages have one of the following formats:

```
%fac-s-ident, text
-fac-s-ident, text
```

Code	Meaning
fac	The facility code; either CPS or LPS
%	The prefix to all primary messages
–	The prefix to all continuation messages
s	The severity level of the message
ident	An abbreviation of the message text
text	The expanded text of the message

9.1.2 Severity Level

The severity levels of ScriptPrinter 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 which may be recoverable.
F	Fatal — system encountered a fatal error and cannot continue processing this request.

9.2 Message Descriptions

The following message descriptions are alphabetized by the *ident* portion of the messages code. The message prefix, facility designation, and severity code are not shown. If the facility designation is other than

CPS or LPS, refer to the *VAX/VMS System Messages and Recovery Procedures Reference Manual* for the message description.

If the **User Action** calls for submitting an SPR (Software Performance Report), refer to the *Guide to VAX/VMS System Management and Daily Operations* for instructions.

BAD_DEVCTL, Bad library device control specification — *string*.

Explanation: You see this message when an error occurs in the specification of a component, indicated by *string*, in the logical device control library.

Action: Change the specification and restart the queue.

BADOPC, OPC belt is bad

Explanation: OPC belt requires replacement.

User Action: Replace the OPC cartridge. Refer to replacement instructions in the *LN03R ScriptPrinter Operator Guide*.

Do not remove the OPC cartridge from its black bag until you are ready to install it. The cartridge degrades if exposed to light for more than 5 minutes.

BADOPCTONERUFL, OPC belt is bad and or toner empty

Explanation: Toner cartridge is empty and/or the OPC belt requires replacement.

User Action: Replace both the toner cartridge and the OPC cartridge. Follow the directions in the respective replacement kits or refer to the procedures described in the *LN03R ScriptPrinter Operator Guide*.

BADLAYDEF, *condition* on line *line number* in layup definition

Explanation: An error appears in the layup definition file included with your print job.

User Action: See Section 9.2.1 for a description of the text that can appear as the *condition* in a BADLAYDEF layup definition error message. Check the line in the layup definition file that is indicated in the error message.

CANTCHECKPNT, Checkpointed job *job number* is requeued

Explanation: A print job was stopped and has been requeued. The job will print from the beginning.

User Action: Check the printed output. If it is not complete, resubmit the job.

CANTUSETRN, Translator from *data-type* to POSTSCRIPT is unusable

Explanation: The translator generated a severe error and has been marked unusable. Subsequent jobs with data type *data-type* also incur this message and are placed in a HOLD status by the symbiont.

User Action: Ask your system manager to restart the print queue. This loads a new copy of the translator. When the queue is restarted, release the jobs that were placed on HOLD. Submit a Software Performance Report.

CMEMERR, internal controller memory error

Explanation: The LN03R has a memory failure in the printer controller.

User Action: Call DIGITAL Field Service. Refer to the *LN03R ScriptPrinter Operator Guide.*

CONAPPLICATION, Connection request is not to a LAT applications port

Explanation: You did not request a connection to a recognized LAT application port.

User Action: Specify the correct applications port name in the LTLOAD.COM and restart the queue.

CONTERMINATED, Connection abnormally terminated

Explanation: Your connection to the local area Ethernet terminated abnormally.

User Action: Make sure that the DECserver has power and that the setup of the DECserver port serving the ScriptPrinter agrees with the characteristics listed in the *LN03R ScriptPrinter Installation Guide.* Restart the queue.

CONTIMEOUT, Connection timed out, server not available, or incorrect server name specified

> **Explanation:** The connection timed out. You selected a server that was not available or you provided an incorrect server name. The timeout period is 5 seconds.
>
> **User Action:** Specify the correct server name of an available server. Restart the queue.

DATAOVERUN, Data overrun

> **Explanation:** This indicates a communication error.
>
> **User Action:** Try setting a lower baud rate for the ScriptPrinter.

DEVSYNERR, Syntax error in ON qualifier. *Queue-name* not valid printer name

> **Explanation:** The symbiont detected an error when parsing the name value of the ON qualifier of the INITIALIZE/QUEUE command. It determined that the specified name is not a ScriptPrinter queue.
>
> **User Action:** Stop the queue, delete it, and reinitialize it with a valid LN03R queue name.

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

> **Explanation:** The POSTSCRIPT interpreter sensed this error while trying to execute the POSTSCRIPT command represented by *string*.
>
> **User Action:** If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.
>
> If the error occurred while printing a POSTSCRIPT file, 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.

DICTSTKOV, dictstackoverflow: Too many begins

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurred while printing a POSTSCRIPT file, 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.

DICTSTKUF, dictstackunderflow: Too many ends

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurred while printing a POSTSCRIPT file, 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.

DISABLED, Print queue is currently disabled and cannot accept jobs

Explanation: You submitted a print job for a print queue that is not accepting jobs.

User Action: Ask your system manager to enable job acceptance on that print queue.

DPLXNOSUP, *printername* does not support duplex printing

Explanation: The ScriptPrinter does not support printing on two sides of the paper.

User Action: Resubmit your print job without asking for duplex printing or send it to a printer that supports duplex printing.

DRIVEERR, Print Engine driving unit error — FATAL ERROR

Explanation: The ScriptPrinter has a fatal print driver problem.

User Action: Call DIGITAL Field Service. Refer to the *LN03R ScriptPrinter Operator Guide*.

DTSCOPEN, Print Engine developer tray/side cover is open

Explanation: The developer tray is open or the side cover is open.

User Action: Close the developer tray or the side cover.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurred while printing a POSTSCRIPT file, 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.

EXITSRVR, exitserver has been executed — permanent state may be changed

Explanation: The POSTSCRIPT operator **exitserver** has been successfully executed. The permanent POSTSCRIPT system parameters may have been altered. This message is also displayed during the first job after printer startup or after the system manager has changed a job default.

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

If this message appears at any time except for the conditions listed, notify the system manager.

FLUSHING, Rest of job (to EOJ) will be ignored

Explanation: An error or some other event caused POSTSCRIPT to ignore the rest of the job. Preceding messages should provide information on why the job is being flushed.

User Action: Check the other messages returned.

FPGGTPPS, First page > pages per sheet. First page set to 1.

Explanation: Your print job includes a layup definition file that uses the Firstpage and PagesPerSheet options. However, the FirstPage option requests that the first page be printed in a page spot that is beyond the number specified with the PagesPerSheet option.

User Action: No action is required if you want the first page to be printed in the page spot given with the PagesPerSheet lauyup option. For example, if PagesPerSheet=5, the first page on the first sheet will print in the fifth page spot. If you want a differenct spot, edit the layup definition file and resubmit your print job.

FTLDEVCTL, Fatal device control library problem. Config error: *configuration error* on *printername*

Explanation: The device control library has a fatal error.

User Action: Submit an SPR.

FUSCOPEN, Print Engine fuser cover is open

Explanation: This message occurs when the paper exit cover is open.

User Action: Close the paper exit cover.

FUSINGERR, Print Engine fusing error — FATAL ERROR

Explanation: There is a fatal error in the fusing mechanism.

User Action: Call DIGITAL Field Service. See the chapter on service in the *LN03R ScriptPrinter Operator Guide*.

HANGUP, Data set hang-up

Explanation: The ScriptPrinter was power cycled during a print job and therefore aborts the job.

User Action: Resubmit the lost print job. Subsequent jobs will print normally.

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 store into read-only object — offending command is *string*

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

INVEXIT, invalidexit: Exit not in loop

> **Explanation:** The POSTSCRIPT interpreter sensed an error while trying to execute the POSTSCRIPT command *exitserver*.
>
> **User Action:** If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.
>
> If the error occurs while printing a POSTSCRIPT file, 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.

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:** If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.
>
> If the error occurs while printing a POSTSCRIPT file, 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.

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:** If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.
>
> If the error occurred while printing a POSTSCRIPT file, 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.

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: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

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

Explanation: The ScriptPrinter sensed an I/O error as a result of a communication fault.

User Action: Resubmit the job to be printed. If the error occurs again, examine the communications line for hardware, data rate, or parity/framing problems.

ITCOPN, Print engine paper input tray cover is open

Explanation: The paper input tray is open.

User Action: Close the paper input tray cover.

JOBABORT, Job *jobnumber* aborted

Explanation: The specified job was aborted for one of the following reasons:

1. You deleted an active job on the local print queue.
2. POSTSCRIPT requested that the job be aborted due to a printer or internal error.
3. The symbiont process crashed.
4. An internal error occurred in the Job Controller.

User Action: Take the following corresponding action:

1. None.

2. Another error message accompanies the JOBABORT message. That message gives you specific information about the required action.
3. Submit a Software Performance Report with a description of your LN03R configuration and the circumstances under which this error occurred.
4. An internal error occurred in the Job Controller.

JOBFINISH, Job *jobnumber* finish

Explanation: The symbiont has sent the last packet of user data to the LN03R. Job trailer and job log pages follow, if enabled.

User Action: None.

JOBSTART, Job *jobnumber* start

Explanation: The symbiont has started to print the specified job.

User Action: None.

KEYNOTREC, The keyword in the PARAMETERS qualifier was not recognized

Explanation: The symbiont detected an invalid keyword in the PARAMETERS qualifier of the PRINT command.

User Action: Check the spelling and/or the documentation of the parameter you entered. Resubmit the print request with the correct keyword.

LAYUPIGNORED, layup_definition parameter ignored, since Number_Up=0 was specified

Explanation: Specifying NUMBER_UP=0 disables layup.

User Action: If you want to use a layup definition file, specify a value for NUMBER-UP, or do not use the NUMBER_UP parameter. The default is NUMBER_UP=1.

LAYUPOPENERR, Can't open layup definition file *filename*

Explanation: The layup definition file you specified in the PRINT command is not in LPS$LAYUP.

User Action: Check the spelling of the layup definition filename and try the command again.

LAYUPREADERR, Read error on layup definition file *filename*

Explanation: The layup definition file you specified in the PRINT command is not in LPS$LAYUP.

User Action: Check the spelling of the layup definition filename and try the command again.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

LOGOPENERR, Can't open log file *filename*

Explanation: This error occurs with /PARAMETERS= messages=keep, as either a PRINT line request or a queue logical, when you cannot open a log file.

User Action: See the system manager for help.

LOGWRITERR, Write error on log file *filename*

Explanation: This error occurs with /PARAMETERS= messages=keep, as either a PRINT line request or a queue logical, when you cannot write to a log file.

User Action: See the system manager for help.

LOSTPATH, The current path may have been lost

Explanation: You receive this message when layup interacts with a POSTSCRIPT program that contains the results of the **charpath** operator in the current path during a **showpage**.

User Action: Correct the POSTSCRIPT program. Refer to the *PostScript Language Reference Manual.*

LRJACCESSDENIED, Access denied

Explanation: A LAT message indicating your group is not authorized to access the port connected to the ScriptPrinter. Your connection request is rejected.

User Action: Enable the group for the port and start the queue again.

LRJACCESSREJECT, Immediate access is rejected

Explanation: A LAT message indicating you cannot access LAT services at the moment.

User Action: Start the queue again at a later time.

LRJCORRUPT, Corrupted request

Explanation: A LAT message indicating your connection request for LAT services was corrupt and therefore rejected.

User Action: Restart the queue at a later time.

LRJDELETED, Queue entry deleted by server

Explanation: A LAT message indicating the DECserver deleted your queue entry and terminated your LAT connection.

User Action: Restart the queue and send the print job again.

LRJDISABLE, Service is disabled

Explanation: A LAT message indicating LAT service node software is disabled.

User Action: Start the queue again, when the LAT service node is enabled.

LRJILLEGAL, Illegal request parameters

Explanation: A LAT message indicating an internal ScriptPrinter error has occurred.

User Action: Submit a Software Performance Report.

LRJINUSE, Port of service in use

Explanation: A LAT message indicating the port you selected is in use.

User Action: Select a different port for your PrintServer application in the LTLOAD.COM and restart the queue.

LRJNAMEUNKNOWN, Port Name is unknown

Explanation: A LAT message indicating you requested a port unknown to the service.

User Action: Specify the correct port name in the LTLOAD.COM and CPS$STARTUP.COM. Restart the queue.

LRJNOSERVICE, No such service

Explanation: A LAT message indicating the service name is invalid or does not match any authorized group for the port.

User Action: Specify a valid service name or enable your group for the port connected to the ScriptPrinter. Restart the queue.

LRJNOSTART, Session cannot be started

Explanation: A LAT message indicating you are unable to start another session on that port at the present time.

User Action: Reissue your request again later.

LRJNOTOFFERED, Service is not offered on the requested port

Explanation: A LAT message indicating you have requested a service that is not offered on that particular port.

User Action: Use SHOW SERVICES and SHOW NODES to check service names and node names. Specify the name of a port that offers the required service (connection to a ScriptPrinter) in the LTLOAD.COM and restart the queue.

LRJNOTSUPPORT, Requested function is not supported

Explanation: A LAT message indicating an internal ScriptPrinter error has occurred.

User Action: Submit a Software Performance Report.

LRJRESOURCE, Insufficient resources at server

Explanation: A LAT message indicating the DECserver cannot service your request.

User Action: Connect to another DECserver with an available port.

LRJSHUTDOWN, System shutdown in progress

Explanation: The system is shutting down.

User Action: Resubmit your connection request when system resources have returned.

LRJUNKNOWN, Unknown

Explanation: Your request for LAT connection is rejected for reasons that cannot be determined.

User Action: Refer to the *LAT/VMS Management Guide* for help to correct the problem and restart the queue.

NEGAREA, Layup definition margins overlap one another

Explanation: You have included a layup definition file in your print job that sets the margins so that they overlap.

User Action: Edit the layup definition file to change the margins and resubmit your job. Refer to the MARGINS option in Chapter 6 for information.

NOCOPYPG, "copypage" is not supported by multipage layup

Explanation: The *copypage* operator is ignored when you use layup to print more than one page to a sheet.

User Action: None.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

NODEVCTLLIB, No device control library specified for the queue

Explanation: The printer queue does not have a device control library.

User Action: Ask your system manager to associate the standard device control library with the queue and submit your request again.

NODUPFACEUP, Duplex to "face_up" outputtray is not supported on *printer name*

Explanation: The ScriptPrinter does not support printing on both sides of a sheet of paper.

User Action: Resubmit your print job to a printer that supports duplex printing or resubmit your job to the ScriptPrinter without requesting duplex printing.

NOINPTRAY, INPUT_TRAY selection not supported for *printer name*

Explanation: The ScriptPrinter does not support the INPUT_TRAY parameter to the PRINT command.

User Action: Resubmit your job without requesting an input tray.

NOLAYUPFIL, Layup definition module *module name* is not accessible

Explanation: The layup definition file you specified in the PRINT command is not in LPS$LAYUP.

User Action: Check the spelling of the layup definition filename and try the command again.

NOONESDUP, One-sided-duplex is not supported on *printer name.*

Explanation: The ScriptPrinter does not support one sided duplex printing.

User Action: Resubmit your job without requesting one sided duplex printing.

NOOUTTRAY, OUTPUT_TRAY, *tray name*, not supported on *printer name*

Explanation: The ScriptPrinter does not support the specified option of the OUTPUT_TRAY parameter on the PRINT command.

User Action: Resubmit your job without requesting an output tray or use one of the acceptable options, TOP or SIDE.

NOSETTRAY, Tray selection is not supported by multipage layup

Explanation: Output tray selection operators are ignored when you use layup to print more than one page to a sheet.

User Action: None.

NOT_READY, Printer not ready.

Explanation: You receive this message when you try to start a print queue and the printer is not connected or powered on.

User Action: Connect or turn on the printer and start the print queue again.

OFFLINE, Print engine has gone off line

Explanation: The ScriptPrinter is $\boxed{\text{off line}}$.

User Action: Put the printer back $\boxed{\text{on line}}$.

OPCMARKERR, OPC synchronous mark error

Explanation: The printer has a problem with its OPC mechanism.

User Action: Call DIGITAL Field Service. Refer to the *LN03R ScriptPrinter Operator Guide*.

OPTICERR, Print Engine optical unit error — FATAL ERROR

Explanation: The printer has a fatal problem in the optical unit.

User Action: Call DIGITAL Field Service. Refer to the *LN03R ScriptPrinter Operator Guide*.

PAPERJAM, Paper jam, job put on hold

Explanation: A paper jam has occurred at the printer. The print job has been put on hold.

User Action: First clear the paper jam at the printer. Next, issue the following DCL command to release and restart the print job:

```
$ SET QUEUE/ENTRY=nnn/RELEASE queue-name
```

where:

nnn	is the print job entry number in the queue.
queue-name	is the name of the print queue.

The print job will be printed again from the beginning.

PARSYNERR, Syntax error in the PARAMETERS qualifier at or near *string*

Explanation: The symbiont detected a syntax error in the PARAMETERS qualifier of the PRINT command. The error was at or near the *string* parameter.

User Action: Check the syntax and/or the documentation of the parameters you are trying to enter. Resubmit the print request with the correct syntax.

PPSGTNUP, Pages per sheet > Number_Up. Pages per sheer set to Number_Up

Explanation: You have included both a layup definition file and the NUMBER_UP parameter in your print job. However, the layup definition file requests that the number of pages printed per sheet be greater than the number specified with the NUMBER_UP parameter.

User Action: No action is required if you want the number of pages per sheet to equal the number you specified with the NUMBER_UP parameter. If you want the number of pages per sheet to equal the number given in the layup definition file, reissue the PRINT command, specifying a larger number with the NUMBER_UP parameter.

PREADY, The printer is ready

Explanation: Informational message.

User Action: None.

PRESET, resetting printer

Explanation: The printer controller has detected an internal error. It executes the power-up sequence, which effectively resets the printer and the POSTSCRIPT interpreter.

User Action: No user action required.

PRHDWERR, Print Engine hardware error

Explanation: The printer needs professional help.

User Action: Call DIGITAL Field Service. Refer to the *LN03R ScriptPrinter Operator Guide*.

PRINTERSTALLED, Printer *printer name* is stalled

Explanation: You usually see this message when the printer is out of paper. Sometimes POSTSCRIPT commands that take a long time to print cause this message.

User Action: Check the input tray for paper.

PRUNKERR, Print Engine unknown error status

Explanation: The printer needs professional help.

User Action: Call DIGITAL Field Service. Refer to the *LN03R ScriptPrinter Operator Guide*.

PRWRDWERR, printer timed out

Explanation: The print engine reports no error condition, but it is not yet ready to print after correcting a print engine error. This state is temporary.

User Action: No user action required.

PWRCYCL, The printer power was cycled

Explanation: Informational message. The power to the ScriptPrinter was turned off and on.

User Action: None.

PWRFAIL, The printer power was cycled while a job was active

Explanation: The power to the ScriptPrinter was turned off and on while a job was printing.

User Action: Check your print job to see if you need to resubmit it for printing.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

SETUPNOTFOUND, Setup module *module name* not found

Explanation: The setup module you requested cannot be found in the device control library.

User Action: Check the spelling of the module name and resubmit the print request.

SETUPREADERR, Setup module *module name* read error

Explanation: The setup module you requested cannot be read.

User Action: See your system manager for help.

SHEETLIMOBS, Sheet_limit parameter is obsolete. Page_limit is being used instead

Explanation: The SHEET_LIMIT parameter has been replaced. If the user specifies SHEET_LIMIT in the VMS PRINT command, the job proceeds as if the PAGE_LIMIT parameter was specified and the user receives this informational message.

User Action: None. Use the PAGE_LIMIT parameter in the future.

SIZNOTAVL, No *paper size* size medium is loaded in *printer name*

Explanation: The paper size you requested is not loaded in the ScriptPrinter.

User Action: Load the desired paper size in the printer or resubmit the PRINT command with the paper size already loaded in the printer.

SIZNOTRAY, *paper size* size medium is not in the *tray name* tray in *printer name*

Explanation: The paper size you requested is not loaded in the ScriptPrinter's input tray.

User Action: Either load the desired paper size, or resubmit your job requesting the paper size already in the printer.

SIZNOTSUP, *paper size* size medium is not supported by *printer name*

Explanation: You requested a paper size that the ScriptPrinter does not support.

User Action: Resubmit your job with a supported paper size.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

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

Explanation: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

User Action: If the error occurs while printing a POSTSCRIPT file, 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.

STRTOOLON, String is too long — it has been truncated

Explanation: A string passed internally is too large for the receiving buffer.

User Action: This is a software problem; submit a Software Performance Report.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, it indicates an error in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

TIMOUT, timeout: Time limit exceeded

Explanation: A POSTSCRIPT program has executed beyond the time limit set by the system manager. This can be due to either an unusually complex page definition or to an error in the POSTSCRIPT application program that results in an infinite loop.

User Action: If the error is a result of a complex page definition, ask the system manager to increase the value of the job time-out limit.

If this is an error condition while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

TNREND, Print Engine toner supply is exhausted

Explanation: The printer is out of toner.

User Action: Replace the toner cartridge and cleaning pad. Refer to the instructions in the toner replacement kit or in the *LN03R ScriptPrinter Operator Guide*.

TONEROFL, Toner collection container full

Explanation: The toner collection container is full.

User Action: Following the directions in the toner kit, you must remove the full toner container bottle and replace it with a new bottle.

TRAYEMP, Print Engine paper input tray is empty

Explanation: The paper input tray is empty.

User Action: Add paper to the input paper tray.

TRAYSUBST, Output will be delivered to the *output tray* tray on *printer name*

Explanation: Informational message.

User Action: None.

TUMBNOSUP, *printer name* does not support tumble printing

Explanation: The ScriptPrinter does not support tumble printing.

User Action: Resubmit your job without requesting tumble printing.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

UNKDATATYPE, Unknown datatype: *string* or translator not available

Explanation: The symbiont does not recognize the requested DATA_TYPE.

User Action: Check the spelling of the DATA_TYPE and resubmit your request.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

USERDATA, *string*

Explanation: The POSTSCRIPT program requested that data be printed (POSTSCRIPT operators *print, pstack,* =, = =).

User Action: None. Message is informational.

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

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

User Action: If this error occurs while printing a file with a data type other than POSTSCRIPT, an error exists in the translation process. If the translator is DIGITAL-supplied, submit a Software Performance Report.

If the error occurs while printing a POSTSCRIPT file, 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.

ZEROAREA, Layup definition margins result in no usable sheet area

Explanation: You have included a layup definition file in your print job that sets margins that leave no area for printing.

User Action: See Chapter 6 for information on the margins option in a layup definition file.

9.2.1 Layup Definition (BADLAYDEF) Messages

This section descibes the layup definition error messages you receive with the message identification BADLAYDEF. Messages appear in the following format:

```
%CPS_W_BADLAYDEF, <condition> on line <line-number>
in layup definition
```

The <condition> describes the problem and <line-number> is the line number in the layup definition file on which the error occurred. The following error messages are generated by errors in a layup definition file and reported as the <condition>.

NOTE

Line numbers reported in the BADLAYDEF message may not always be correct for some RMS file organizations.

Bad form for margin values

Explanation: The value given for the MARGINS option cannot be understood by that option.

Bad form for page grid values

Explanation: The values given for the GRID option cannot be understood by that option.

Cannot give values with negated option *option*

Explanation: A line contains both a negated option and a value, for example, NoAlternate=left.

Cannot negate option *option*

Explanation: An option that cannot be negated is negated.

Cannot supply a value for *option*

Explanation: A value is given for an option that cannot take a value.

Could not find a number as a value

Explanation: The value given for an option cannot be understood by that option.

Extra characters present after values

Explanation: Legitimate values for an option are followed by extra characters.

Must express a value for option *option*

> **Explanation:** No value is given for an option that requires a value.

No option present

> **Explanation:** A line is not blank, but it also does not contain an option.

No values specified for option *option*

> **Explanation:** A line contains an equal sign but no values.

Number must be greater than 1

> **Explanation:** The numeric values for an option are out of range, for example, PagesPerSheet=0.

Page count must be less than 100

> **Explanation:** The GRID option must have positive values. The product of the two values must be 100 or less.

Unrecognized keyword *keyword*

> **Explanation:** The value given for a keyword cannot be understood.

Unrecognized option *option*

> **Explanation:** A line does not contain a recognized option.

Appendix A

Programming the ScriptPrinter

This appendix gives guidelines for writing POSTSCRIPT programs that execute successfully in the ScriptPrinter in a VAX/VMS environment. For more general programming considerations (applicable to all operating systems supported by the ScriptPrinter), see the *LN03R ScriptPrinter Programmer's Supplement.*

On some POSTSCRIPT printers, a user program interacts with the content of the native POSTSCRIPT interpreter. The ScriptPrinter printing system has additional POSTSCRIPT procedures between the user program and the POSTSCRIPT interpreter. These procedures provide features that must be considered when writing POSTSCRIPT programs for the ScriptPrinter in a VMS environment. System software that controls POSTSCRIPT printers sends POSTSCRIPT procedures for device control to the printer ahead of the user's POSTSCRIPT program.

A.1 Printing Device Control

System software sends POSTSCRIPT procedures to the POSTSCRIPT printer. The function of these procedures is determined by the host operating system and printer control software. Some functions performed by VMS device control procedures are:

- Report error conditions to the user
- Report print job accounting information to the user
- Control the portion of a print job that will produce output
- Provide custom character set encodings

The device control procedures are sent to the printer before or after the user's POSTSCRIPT program. These procedures reside in the printer until the print job terminates and can modify the behavior of the print job by:

- Using POSTSCRIPT extension operators
- Redefining native POSTSCRIPT operators

A.1.1 Using POSTSCRIPT Extension Operators

The POSTSCRIPT extension operators are used to control the printer. These operators are unique to the printer model or class of printers. The POSTSCRIPT extension operators for the ScriptPrinter are described in the *LN03R ScriptPrinter Programmer's Supplement.*

To ensure device-independent page descriptions, application software should not refer to the operator extensions in a POSTSCRIPT program. When extension operators must be used, have your program check to see that the operator exists before executing it. Use extensions that produce the desired output on the largest set of different printer models.

A.1.2 Redefining Native POSTSCRIPT Operators

Printer control software can redefine native POSTSCRIPT operators to modify a print job's behavior. This programming technique is called **layering**; it is made possible by the way POSTSCRIPT performs implicit name searches. Layering may also be used by document-formatting utilities that include portions of one POSTSCRIPT program in a second POSTSCRIPT program.

Layering inserts a **context layer** between the user's POSTSCRIPT program and the native POSTSCRIPT interpreter. The context layer consists of the POSTSCRIPT stacks and the graphics state. This layer modifies the behavior of the user's POSTSCRIPT program, which is executed on top of the context layer. Although the context layer modifies the behavior of the user program, the context layer is transparent to the user program. In fact, the user program may run on top of several different context layers, all equally unaware of each other's existence.

The **current context** is the POSTSCRIPT state resulting from all the layers present when a program runs: user, device control, system management, and native POSTSCRIPT.

POSTSCRIPT programs that violate the implicit name search order can cause unexpected results, as they bypass and possibly conflict with underlying context layers.

POSTSCRIPT programs cannot assume the order of objects on the various POSTSCRIPT stacks at the start of a job, because the stack order may have been modified by an underlying context layer.

Make sure that the context at the program's completion is equivalent to the context at program start by beginning each program with a **save** operator, and ending each program with a **restore** operator.

A.2 Interaction of PostScript Operators and Command Line Qualifiers

A command qualifier specified on the command line may conflict with a POSTSCRIPT operator embedded in a page description. In such cases, the command invoked last takes effect. In almost all cases, this is the POSTSCRIPT operator in the page description.

The following PRINT command qualifiers can conflict with POSTSCRIPT operators in a print request:

- OUTPUT_TRAY
- PAGE_ORIENTATION
- PAGE_SIZE
- SHEET_COUNT
- SHEET_SIZE

A.3 Restrictions to PostScript on the ScriptPrinter

The ScriptPrinter imposes restrictions on the standard use of **gsave** and the dictionary stack.

A.3.1 Limit to Use of gsave

The *PostScript Language Reference Manual* states that you can have a maximum of 31 active **gsave** commands before you get a **limitcheck** error. However, the ScriptPrinter POSTSCRIPT interpreter uses two active **gsave** commands outside the server loop, so the maximum number available to programs is 29. If other libraries also use active **gsaves**, fewer are available for user programs.

A.3.2 Limit to Dictionary Stack Entries

The *PostScript Language Reference Manual* states that the maximum number of entries on the dictionary stack is 20. However, on the ScriptPrinter, POSTSCRIPT uses two entries outside the server loop and the device control modules use 3 entries, so a maximum of 15 entries on the dictionary stack is available to user programs.

Appendix B

Differences Between the ScriptPrinter and PrintServer Software

This appendix describes some of the differences between the ScriptPrinter Software, Version 2.0 and the PrintServer Software, Versions 2.0 and 2.1. For more information on compatibilities and differences between the ScriptPrinter and the PrintServer, see the *LN03R ScriptPrinter Programmer's Supplement.*

B.1 Print Parameters

DCL commands are entirely compatible on the ScriptPrinter, Version 2.0, and the PrintServer, Versions 2.0 and 2.1. The only differences are the parameters used to specify input/output tray selections and paper size selections. Table B–1 compares the tray and paper selection parameters of the two printers.

Table B–1: Comparison of Tray and Paper Selection Parameters

Parameter	PrintServers	ScriptPrinter
INPUT_TRAY=		
TOP	Selects top input tray	Produces an error; job aborts
MIDDLE	Selects middle input tray	Produces an error; job aborts
BOTTOM or LCIT	Selects large capacity input tray	Produces an error; job aborts
OUTPUT_ TRAY=		
TOP	Selects top output tray; face-down stacking	Prints job
SIDE	Selects side output tray; face-down stacking	Prints job
FACE_UP	Selects side output tray; face-up stacking	Produces an error; job aborts
PAGE_SIZE=		
LETTER (A)	Supported	Supported
LEDGER (B)	Supported	Available through page layup feature
LEGAL	Supported	Available through page layup feature
EXECUTIVE	Supported	Available through page layup feature
A5	Supported	Available through page layup feature
A4	Supported	Supported
A3	Supported	Available through page layup feature
B5	Supported	Available through page layup feature
B4	Supported	Available through page layup feature
SHEET_SIZE=		
LETTER (A)	Supported	Supported if same as paper size switch
LEDGER (B)	Supported	Produces an error; job aborts

Table B–1 (Cont.): Comparison of Tray and Paper Selection Parameters

Parameter	PrintServers	ScriptPrinter
LEGAL	Supported	Produces an error; job aborts
EXECUTIVE	Supported	Produces an error; job aborts
A5	Supported	Produces an error; job aborts
A4	Supported	Supported if same as paper size switch
A3	Supported	Produces an error; job aborts
B5	Supported	Produces an error; job aborts
B4	Supported	Produces an error; job aborts

Errors in the /PARAMETER switches are not detected or reported until the job is being printed. Both the PrintServers and the ScriptPrinter will accept the same command lines since the user may not know which printer is in use. Refer to the Chapter 8 for more information on the qualifiers, parameters, and switches.

B.2 Job Separation Pages

The job separation pages of the ScriptPrinter Software, Version 2.0, and the PrintServer Software, Versions 2.0 and 2.1 differ in the following ways:

- The ScriptPrinter has a device name; the PrintServer has a node name.
- The ScriptPrinter symbiont identifies its name and version; the PrintServer symbiont does not.
- The ScriptPrinter job trailer page contains two job log messages; the PrintServer job trailer page does not.

B.3 Other Differences

In additon to print parameter and separation page differences, the ScriptPrinter, Version 2.0 varies in the following ways:

- The INITIALIZE/QUEUE/ON command takes a device name rather than a node name.
- Some logical names and device control modules use the CPS$ facility code.
- PRINT/NOTIFY results in one end-of-job message rather than two.
- The DATA_TYPE parameter supports ANSI2, LINE, and TEXT as synonyms for ANSI and PS as a synonym for POSTSCRIPT.
- You can use PSM$ANNOUNCE on flag pages.
- STOP/QUEUE, STOP/QUEUE/ABORT, and SHOW QUEUE do not differ from standard VMS usage.

Glossary

applications port The logical device for an application program on a node running the local area transport (LAT) software.

bitmap An image in digitized form that can be stored, transmitted, and reproduced.

composite character A character that is made up of two or more other characters. For example, an accented letter is a composite character.

device control library A library that contains a series of text modules that can be sent to the device associated with a queue to affect the behavior of that device.

device control module On a ScriptPrinter, a POSTSCRIPT procedure that is stored in the print symbiont's device control library. You can use device control modules to affect the behavior of a device. For example, you can send a device control module to set a printer to a known state.

encoding vector The association between character codes and character descriptions. An encoding vector on the ScriptPrinter is a 256-element array, indexed by character code. The elements of the array are character names.

Ethernet network A local area network that uses coaxial cable as a passive communication medium to interconnect different types of computers, printer products, and office equipment at a site.

execution queue On the ScriptPrinter, the queue with which the symbiont is associated.

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

generic queue On the ScriptPrinter, a queue that has a default data type associated with it.

host The computer that provides services and enables startup and management of the peripheral devices, such as printers.

initialize To set counters, switches, addresses, or contents of memory to 0 or other starting values at the beginning of, or at prescribed points in, a computer routine.

interpreter A stored program in the printer that converts data for an imaging data syntax, such as text or graphics, into a bitmap.

kerning Subtracting the space between characters.

language extensions POSTSCRIPT operators and objects added to the standard POSTSCRIPT operators and objects in a specific implementation of the POSTSCRIPT interpreter. These extensions control the system parameters for that type of POSTSCRIPT printer.

line A physical communications path.

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 by communications lines to share information and resources.

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

object type of extension POSTSCRIPT language extensions can be of type **operator**, **integer**, or **string**. See the *PostScript Language Reference Manual* for more information.

operator A built-in POSTSCRIPT language object that performs an operation. An operator, when invoked, can result in data being manipulated, system parameters being modified, and/or sheets being printed.

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

page spot The place on a sheet that a page can be printed. Using layup, you can have multiple page spots on a single sheet.

persistent parameters System parameters that remain unchanged across POSTSCRIPT print jobs. These parameters are stored in nonvolatile memory and, therefore, are maintained even when power is removed from the printer.

port The hardware on the DECserver that transmits and receives data to and from an attached peripheral device or the Ethernet transceiver.

port device The hardware unit attached to a DECserver port. Typically, a port device is a video display terminal or a serial line printer.

POSTSCRIPT interpreter A stored program in the ScriptPrinter that converts POSTSCRIPT commands into page description.

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.

printer controller firmware The firmware that interprets the data in a print request according to a specified data syntax, builds bitmaps of each page to be printed, and forwards the bitmaps to the print engine that produces the hard-copy output.

printer software The software that handles the communications among the process that makes a print request (terminal), the process that provides resources (a host), and the process that performs the printing service (a print queue).

print spooler A data storage capability that enables a computer to maintain a print queue and print documents while performing other computer tasks.

print symbiont The software that processes a user's print request, arranges to have data translated if required, and queues the request and data for transfer to the ScriptPrinter.

privilege The level of system access allowed to a user.

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

service A resource provided by network computer services that is available to DECserver port users.

session A connection or interaction between a port user and a service.

sheet In the context of the ScriptPrinter, a sheet is the physical piece of paper that is printed.

system parameter Values maintained within the POSTSCRIPT interpreter and used by the POSTSCRIPT interpreter to control various system specific features. These features are related to the print engine model and the type of communication between the printer and the host. Also see **persistent parameters** and **volatile parameters**.

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 services. These requests are handled by the host, which forwards them to the appropriate queue.

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

Index

HOW TO ORDER ADDITIONAL DOCUMENTATION

From	Call	Write
Alaska, Hawaii, or New Hampshire	603–884–6660	Digital Equipment Corporation P.O. Box CS2008 Nashua, NH 03061
Rest of U.S.A. and Puerto Rico*	1–800–DIGITAL	

* Prepaid orders from Puerto Rico, call DIGITAL's local subsidiary (809–754–7575)

Canada	800–267–6219 (for software documentation) 613–592–5111 (for hardware documentation)	Digital Equipment of Canada Ltd. 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6 Attn: Direct Order desk
Internal orders (for software documentation)	—	Software Distribution Center (SDC) Digital Equipment Corporation Westminster, MA 01473
Internal orders (for hardware documentation)	DTN: 234–4323 508–351–4323	Publishing & Circulation Serv. (P&CS) NRO3–1/W3 Digital Equipment Corporation Northboro, MA 01532

Reader's Comments

VAX/VMS Management/User's Guide: ScriptPrinters

AA–JF62C–TE

Your comments and suggestions will help us improve the quality of our future documentation. Please note that this form is for comments on documentation only.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (product works as described)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page layout (easy to find information)	☐	☐	☐	☐

What I like best about this manual: ______________________

____

What I like least about this manual: ______________________

____

My additional comments or suggestions for improving this manual:

____

____

I found the following errors in this manual:

Page Description

_____ ______________________

_____ ______________________

_____ ______________________

Please indicate the type of user/reader that you most nearly represent:

☐ Administrative Support
☐ Computer Operator
☐ Educator/Trainer
☐ Programmer/Analyst
☐ Sales
☐ Scientist/Engineer
☐ Software Support
☐ System Manager
☐ Other (please specify) __________

Name/Title ______________________ Dept. __________

Company ______________________ Date __________

Mailing Address ______________________

______________________ Phone __________

— Do Not Tear — Fold Here and Tape

digital™

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
CORPORATE USER PUBLICATIONS
MLO5–5/E45
146 MAIN STREET
MAYNARD, MA 01754–2571

— Do Not Tear — Fold Here