ScriptServer Printing System
User's Guide

Software Version V5.1
Revision E
March 1998

ScriptServer
PRINTING SYSTEM

The information in this document is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by GrayMatter Software Corporation. GrayMatter Software Corporation assumes no responsibility or liability for any errors, omissions, or inaccuracies that may appear in the software or in this document. The software described in this document is furnished under license and may only be used or copied in accordance with the terms of that license.

Contents

Chapter 1

Introducing ScriptServer Version 5.1

Welcome to version 5.1 of the ScriptServer Printing System. With this release, we have improved the reliability, compatibility, and versatility of the software. We are also introducing a new component, called the ScriptServer Network Console, which allows you to manage and monitor ScriptServer queues from a Windows 95 or Windows NT 4.0 based personal computer.

1.1 OVERVIEW

Some of the major features in this revision are:

- ScriptServer Network Console for Windows NT 4.0 and Windows 95 — This allows the monitoring of all ScriptServer queues and their current status from one location. Users are alerted to alarm conditions such as paper out, toner low, etc. In addition to the status monitor, the Network Console provides the ability to create, modify, delete, start, and stop ScriptServer queues on any OpenVMS host in their network.
- Self-Configuring TCP/IP Queues — When a ScriptServer queue is started on an IP address, it will attempt to connect to several different TCP/IP ports, setting the appropriate values for such configuration parameters as backchannel and Control-T support.
- Repository Queue — This provides the user with the ability to print text files to a queue pointing to an OpenVMS directory, rather than a print device. This gives you the ability to capture PostScript files in a specified directory.
- Robust LPD mode -— This supports printing to all UNIX and Windows NT LPD implementations, and to some printers that were previously unsupported.

1.2 THE SCRIPTSERVER NETWORK CONSOLE

Managing print queues that are dispersed geographically or across multiple systems in a network is a difficult task; typically, you must first connect to the OpenVMS host, log in, and execute a SHOW QUEUE command to see the status of the queue. With the release of the ScriptServer Network Console, however, you now have the ability to monitor printing activity, oversee problems, and control ScriptServer queues throughout your TCP/IP network from a centralized location.

1.2.1 Supported Platforms

The Network Console is completely Microsoft Windows complicant and runs under the following operating systems:

- Microsoft Windows NT Workstation 4.0
- Microsoft Windows NT Server 4.0
- Microsoft Windows 95 (including OEM Service Release 2)

1.2.2 Licensing Information

Though included in the release of ScriptServer 5.1, the ScriptServer Network Console is a separately licensed product that offers a fixed number of seats for remote administration. You may install the Network Console on as many machines as you like, however, the number of simultaneous running Network Console users is dictated by the number licensed seats. Furthermore, a seat is composed of 8 simultaneous users. As an example, if a site wanted to have 16 Network Consoles running at the same time they would need to be licensed for 2 seats.

Presently, a ScriptServer print symbiont can only manage 63 Network Consoles (or 8 seats) running simultaneously across an Enterprise. The maximum number of licensed Network Consoles is controlled by the NCx licensing module in the ScriptServer license key on your OpenVMS system.

If, for any reason, you have any questions concerning the ScriptServer Network Console licensing, please contact GrayMatter Software directly.

1.3 INSTALLING THE NETWORK CONSOLE APPLICATION

The Network Console can be installed from diskette or from downloaded from Gray-Matter Software's web site (http://www.graysoft.com) or FTP site (ftp.graysoft.com).

1.3.1 Hardware/Software Requirements

Aside from needing one of the supported Microsoft operating systems, there are no additional software requirements needed to install and run the ScriptServer Network Console. However, because the Network Console remotely monitors ScriptServer queues, Microsoft Windows must be configured with some sort of connection(i.e. 10 Base-T, 10 Base-2, etc.) to a TCP/IP network.

In terms of hardware requirements, we at GrayMatter Software recommend at least the following configuration:

- IBM PC or compatible computer
- 486/66 mHz processor
- 16 MB RAM
- 5 MB of Hard Drive space
- VGA (or better) video card

1.3.2 Installing From Diskette

To install the Network Console Application onto a personal computer from diskette, follow these steps:

1. Insert the diskette labeled "Setup #1" into your diskette drive
2. Start the SETUP program by either:
   - clicking on the START menu
   - selecting RUN…,
   - typing in A:\SETUP.EXE and hitting the "OK" button

   or…

   - double-clicking on the SETUP icon from within Windows Explorer
3. Finish the installtion by following the on-screen instructions given during SETUP

1.3.3 Installing After Downloading via FTP or the Web

To install the Network Console from a downloaded file:

1. From within Windows Explorer, double-click on the downloaded file…
2. Finish the installtion by following the on-screen instructions.

1.4 CONFIGURING THE NETWORK CONSOLE

After installation has been completed, you will need to configure ScriptServer to manage the various Network Consoles across the Enterprise.

To get the Network Console(s) up and running, you will need to:

- Define the Network Console(s), including their rights and priviledges, for ScriptServer to manage.
- Start the Network Console Agent on the OpenVMS system.

1.5 NETWORK CONSOLE RIGHTS AND CLASSES

Since you may not want all users of the Network Console application to have complete control over your OpenVMS queues, you can control the scope of what users can manipulate through the use of rights and classes.

Rights define what a Network Console user can manipulate. Classes are a set of rights that are associated with a Network Console. There are three classes: User, Operator, and Control. The following table lists the rights and the default classes.

Table 1-1:Network Console rights and classes

Rights	Description	In Classes (by default)
VIEW	can view the status of queues	User, Operator, Control
PAUSE	can pause active queues	Operator, Control
ABORT	can abort the currently active job on a queue	Operator, Control

Table 1-1:Network Console rights and classes

Rights	Description	In Classes (by default)
FORM	can change the currently mounted form on a queue	Operator, Control
START	can start a stopped queue	Control
STOP	can stop a running queue	Control
MODIFY	can change the way that a queue is defined	Control
CREATE	can create new queues	Control
DELETE	can delete an existing queue	Control

1.5.1 Defining Network Consoles

Network Consoles are defined by identifying the IP address of the Windows NT 4.0 or Windows 95 personal computers running the Network Console application. This can be done in either of two ways: by creating logical names on the OpenVMS hosts running ScriptServer queues, or by creating aliases on your network's DNS (Domain Name Services) server. The latter method is recommended because it allows you to establish the IP addresses of your Network Consoles network-wide in one operation.

Rights and classes are defined using logical names that specify the rights in a given class, and associate classes with specific Network Consoles.

NOTE: After defining any Network Console s, you will need to restart the Network Console Agent on the OpenVMS host.

Method 1 — OpenVMS Logical Name

To define Network Consoles using OpenVMS logical names, the logical name SSV$CONSOLE is defined as the IP address of the computer running the Network Console application, followed by the rights-class of that console, separated by a semicolon. Additional Network Consoles are defined by appending a numeral to the logical name. For example:

```
$ DEFINE/SYS SSV$CONSOLE "joe.mydomain.com;operator"

$ DEFINE/SYS SSV$CONSOLE1 "jane.mydomain.com;control"

$ DEFINE/SYS SSV$CONSOLE2 "bill.mydomain.com"
```

In this example, three Network Consoles are defined. The first is at IP address joe.mydomain.com, and has the rights associate with the Operator class. The computer at address jane.mydomain.com has the rights associated with the Control class, and the computer at bill.mydomain.com has the rights associated with the User class.

If you would like to create all of your Network Console definitions in one place, it may be more convenient to do so using your organization's DNS server as described below.

Method 2 — DNS Server Aliases

To define Network Consoles using DNS, you must create alias (canonical) names in the host name table on the DNS server for your network. The first Network Console's alias is "ssv-console" plus the domain name, for example ssv-console.mydomain.com, where mydomain.com is the local domain name. The second Console's alias is ssv-console1.mydomain.com, and so forth. For each Network Console that you add, increase the number in the host name by one.

For example, to define hosts joe and jane in the domain mydomain.com as Network Consoles on a Unix-based DNS server, you might add two entries to the .hosts file similar to the following:

```
ssv-console        INCNAMEjoe.mydomain.com.

ssv-console1       INCNAMEjane.mydomain.com.

ssv-console2       INCNAMEbill.mydomain.com.
```

Note: *If the OpenVMS logical name method is used, it overrides the use of the DNS server method. Or, if you use both methods, the logical name takes precedence and DNS aliases are ignored.*

1.5.2 Changing the Rights Associated with Classes

To change the rights associated with Network Console classes, you create logical name definitions in the format SSV$NC_class, which equate to the rights associated with that class. For example, the default rights settings for each class would be defined as follows:

```
$ DEFINE/SYS SSV$NC_USER “VIEW”

$ DEFINE/SYS SSV$NC_OPERATOR “VIEW,PAUSE,ABORT,FORM”
$ DEFINE/SYS SSV$NC_CONTROL “VIEW,PAUSE,ABORT,FORM,START,STOP,MODIFY,CRE-
ATE,DELETE”
```

1.6 STARTING THE NETWORK CONSOLE AGENT

Once you have defined all the Network Consoles and their priveleges, you will need to start the Network Console Agent on the OpenVMS host. This Agent must be running on the OpenVMS system in order for the Network Console(s) to receive any information regarding the status of ScriptServer queues.

To start the Network Console Agent, at the OpenVMS command prompt type:

```
$ @SCRIPTSERVER:SSV$START_AGENT.COM
```

1.7 NETWORK CONSOLE FUNCTIONAL OVERVIEW

The Network Console receives informational, status, OPCOM, and alarm messages from the ScriptServer print symbionts running on the network. To minimize network overhead, these messages are based on the UDP protocol. As messages are received, they are displayed in the currently selected sort order. The information displayed in the status window is also saved in an on-disk backing store, so that the last known status information can be loaded the next time the Network Console starts up.

When manipulating queues, the Network Console sends the control messages to a Network Console Agent on the selected host. The Agent is a separate program that runs detached on the OpenVMS system to which the command is directed.

1.7.1 Using the ScriptServer Network Console

The main window of the Network Console displays all active ScriptServer queues, their hosts, active jobs, and status information. It lets you know at a glance if there is a problem that requires manual intervention to correct, such as paper out, paper jam, and other hardware conditions that cause a ScriptServer print queue to be stalled.

The main window is divided into status, summary, OPCOM, and informational sections. By clicking on a status line, the queue description for the selected queue is displayed along the bottom of the window. Clicking on the "Show Alarms" button displays another window summarizing stalled queues with the reason for the stall. Clicking on the button to the right of the column headings collapses the window so that OPCOM messages are not visible.

1.7.2 Controlling the Queue Status Display

This window is updated whenever a message is received from a ScriptServer symbiont. You can change the way things are displayed in several ways:

- clicking in the column headings changes the display so it is sorted in ascending order by that column
- right-clicking in the column headings sorts the column in descending order
- dragging on the column heading borders allows you to resize the columns

if an alarm has been received, the button in the lower right corner becomes activated; clicking on this button brings up the alarm window

- clicking on the double arrow icon to the right of the column heading collapses the window such that the OPCOM messages are not displayed. The OPCOM display can also be controlled from the View menu.

To remove a queue's status line from the display, select "Delete Status Line" from the Edit menu. There is no confirmation dialog; if you want to restore the last deleted status line, select "Undo Delete Status Line" from the Edit menu. Only the last status line deleted can be restored.

To clear the status display, select "Clear All Data ..." from the File menu. A confirmation dialog will be displayed to verify that this is what you want to do. When you select this option, data in both the on-screen display and the on-disk backing store are deleted.

Including and Excluding Queues and Hosts from the Display

To control which queues are displayed, select "Filters ..." from the View menu. From this dialog you can control exactly which queues and hosts are displayed.

1.7.3 Manipulating Queues

Queues are manipulated by selecting a status line and choosing a command from the *Queues* menu. Using this menu you can create, delete, stop, start, pause, and change the forms on queues. Alternatively, you can right-click on a status line to get a pop-up queue menu allowing manipulation of the selected queue or to create a new queue on the selected queue's host.

The *Queues* menus are context sensitive; commands that do not apply (given the current state of the selected queue) are not available.

Creating Queues

To create a queue, select "Create Queue ..." from the *Queues* menu. This process can be simplified by first selecting a status line for a queue on the host for which you want to create a queue. Enter the name of the queue that you want to create, and select the type of queue from the "Type" drop-down menu.

The different types of queues are summarized in the following table.

Table 1-2:Queue Types

Type of Queue	IP Protocol	Port	Backchannel	Status Reporting
foreign	n/a	n/a	no	no
LPD	lpd	515	no	no
parallel	n/a	n/a	no	no
QMS Status1	proprietary	35	no	partial
repository	n/a	n/a	no	no
serial/LAT	n/a	n/a	yes	yes
telnet	telnet	varies*	yes	yes*

* depends on manufacturer protocol type used

If the Network Console has received a message from the host you want to create the queue on, it will appear in the "OpenVMS host" drop-down menu, otherwise, type in the name of the host and click on the "Configure..." button.

If you are creating a TCP/IP network queue, the following dialog is presented.

Queue Creation Options

After entering the printer's IP address and selecting the manufacturer's protocol, click on the "Options..." button to manipulate the following dialog.

The options that are available in this dialog may be slightly different depending on the type of queue that you are creating.

Queue Options

Table 1-3:Queue Options

Resource accounting	When selected, the ScriptServer print symbiont will retrieve page count information from the printer (assuming that a backchannel is available) before and after each print job. The difference in these values is sent to the OpenVMS Queue Manager as the number of pages printed for the job. This operation may slow throughput on the queue; disable it unless your organization uses resource accounting charge-back information.
Job flag page	This option causes a banner page displaying the host and username owning the print job to be printed before each job.
Job trailer page	This option causes a page displaying the host and username owning the print job to be printed after each job.
Default file flag page	Controls whether banner pages are generated by default before each file printed. This option can be overridden by users when a print job is submitted.
Default file trailer page	Controls whether trailer pages are generated by default after each file printed. This option can be overridden by users when a print job is submitted.
Print Log level	This option controls when the ScriptServer symbiont will generate Print Logs. A setting of All indicates that a Print Log should be generated whenever a message is received from the printer. The Fatal setting indicates that a Print Log should be generated when a fatal error has been detected. None indicates that a Print Log should never be generated.
Text language	Controls whether text files are converted to PostScript (the default setting) or are printed in PCL mode.

Device Options

Table 1-4:Device Options

Disable backchannel	This device option is used to control whether the symbiont will attempt to extract device, configuration, and status information from the printer. In certain configurations (such as parallel and LPD), there is no communications path for backchannel information.
Disable control-T support	Controls whether the ScriptServer symbiont uses the control-T character to get current status information from the printer when printing in PostScript mode. Not all printer models support using control-T for status acquisition.
User info on printer display	This option controls whether the host and username will be displayed on the printer control panel during print jobs. Not all printer models support this feature.
Coalesce LPD packets	This option allows you to control packet coalescence on LPD queues, where send operations are delayed to fill up the send buffer for more efficient I/O operations. This is normally handled using the standard record blocking feature of the symbiont.
Inhibit LPD acknowledgements	Controls whether the symbiont will abort the print job if LPD acknowledgements are not received from the printer.
PostScript support	Controls whether the symbiont will use PostScript commands to communicate with the printer.
PCL/PJL support	Controls the level of PCL and PJL support used by the symbiont to communicate with the printer. Possible values are:
None	There are no PCL or PJL capabilities on the printer
NoPJL	The printer supports PCL commands but not PJL language commands.
Limited PJL	The printer supports the PCL language and a limited number of PJL commands. The command set used by the symbiont in this case is consistent with that found in HP LaserJet 3Si printers.

Table 1-4:Device Options

Full PJL	The printer supports the full complement of PCL and PJL commands.
Dynamic	The symbiont will attempt to determine the printer’s PCL and PJL commands automatically on queue startup.

PART 1

INSTALLATION GUIDE

Chapter 2

Installation

This chapter describes the installation of the ScriptServer Printing System. Before installing the software, you should read this entire chapter, as well as the release notes. The installation procedure for ScriptServer consists of:

- preparing for the installation
- installing the software with VMSINSTAL
- installing the software license key
- creating ScriptServer print queues
- modifying system startup files and
- printing the *Installation Manager's Notes*

2.1 PREPARING THE INSTALLATION

The ScriptServer Printing System uses VMS queue characteristics to route print jobs to the appropriate printer based on font, paper size, envelope feed, color, printer language, as well as duplexing capabilities. The use of queue characteristics is particularly valuable in environments with mixed PostScript and non-PostScript printers, and where PostScript printers of varying capabilities are in use.

Queue Characteristics Usage by ScriptServer

Although the use of queue characteristics by the ScriptServer software is optional, GrayMatter Software recommends their use for the following reasons:

- the SCRIPTSERVER_SPOOLER characteristic can be used by software other than ScriptServer (such as with the PRINT command) to prevent attempts to route PostScript files to non-PostScript printers or PCL to non-PCL printers
- queue characteristics are also used to specify the paper size and font desired when used with PRINT
- jobs can be explicitly routed to, or away from, higher cost-per-page color printers

Queue characteristics consist of a keyword / value pair, where the keyword must be a unique characteristic name on the system. Under the new Queue Manager shipped with VMS V5.5 and later, characteristic values must also by unique. With the Queue Manager shipped prior to VMS V5.5, the values associated with characteristic names need not be unique.

Within ScriptServer, there is a queue characteristic for each of the following:

- 65 printer-resident font names
- 9 paper sizes
- envelope feed capability
- color / monochrome capability
- duplex capability
- PostScript Level 2 capability
- PCL5 capability
- SCRIPTSERVER_SPOOLER

The last characteristic is used to route jobs supporting the ScriptServer Printing System (the SCRIPT facility, for example) to the ScriptServer print spooler. To prevent improper job routing, the value associated with the SCRIPTSERVER_SPOOLER characteristic must be unique.

By default, the installation procedure uses the value 20 for the SCRIPTSERVER_SPOOLER characteristic. If this value is in use on your system by another characteristic name, the value assigned to the SCRIPTSERVER_SPOOLER characteristic must be changed to a unique value.

To determine if you need to change the numeric value associated with the SCRIPTSERVER_SPOOLER characteristic, enter the command:

```
$ SHOW QUEUE/CHAR
```

If any queue characteristics are defined, they will be listed at your terminal. If a characteristic has the value 20 associated with it, the SCRIPTSERVER_SPOOLER characteristic must be changed.

The installation procedure allows you to define the queue characteristics at installation time. If the value associated with the characteristic SCRIPTSERVER_SPOOLER must be changed, the characteristic definitions can be deferred until after the installation is completed, so that the definitions file may be edited.

If you defer queue characteristic definition, the procedure SCRIPTSERVER:CHAR_DEF.COM can be used to create the queue characteristic definitions after installation. Note that this file must first be edited to assign a unique value to the SCRIPTSERVER_SPOOLER characteristic. The procedure contains comments to facilitate editing.

Preparing Your Printer(s)

The procedure allows you to create ScriptServer print queues during installation. If you choose to do this, the hardware must be ready to be put to use.

Before beginning the installation, ensure that your printer(s) are physically and logically connected properly. All printers should be on-line and switched to PostScript mode if possible, or PCL mode if necessary.

2.2 INSTALLING THE SCRIPTSERVER SOFTWARE

The ScriptServer Printing System is installed with the standard VMSINSTAL utility. To perform the installation, log into the SYSTEM account, mount the distribution medium, and start the VMSINSTAL procedure with the command:

```
$ @SYS$UPDATE:VMSINSTAL SCRIPTSERVER MUA0:
```

where **MUA0:** is the VMS device name for the installation device.

2.2.1 Prompts Generated During Regular Installation

At this point in the installation procedure, you will be asked whether or not you wish to utilize the Quick Install option of the software. If you choose not to, you will proceed with the regular installation outlined below. The Quick Install procedure is detailed in the next section.

During the regular installation, there are several prompts in addition to the initial prompts generated by VMSINSTAL. You should be prepared to respond to each prior to beginning the installation. They are:

- Disk and directory in which to place ScriptServer files.
- Whether or not to define queue characteristics.
- Whether or not you wish you invoke the international defaults.
- Whether replaced files should be purged.
- Whether or not you wish to install the on-line documentation.
- Whether the software license key should be installed.
- Whether or not you wish to create any ScriptServer queues.

If you respond to the first prompt with the name of a directory that doesn't exist, it will be created automatically for you. GrayMatter recommends that you use the default location of SYS$SYSDEVICE:[SCRIPTSERVER], unless some special requirement of your installation precludes it.

VMS queue characteristics, and their use by ScriptServer, are discussed in detail in Section 2.1 *"Preparing The Installation*" on page 35 of this document. Respond to the second prompt as the needs of your installation dictate.

The fourth prompt determines whether VMSINSTAL purges files that are replaced during the course of installation. At sites that are installing ScriptServer for the first time, this will only effect the system DCL tables (SYS$LIBRARY:DCLTABLES.EXE).

The sixth prompt of the installation asks you whether or not you wish to install the on-line documentation. This documentation consists of:

- the *ScriptServer Printing System User's Guide*;
- the *ScriptServer Printing System Reference Manual*;
- the *ScriptServer Printing System Fax Facility User's Guide*; and
- the *ScriptServer Printing System Release Notes*

Responding YES to this prompt will install all of the above in the ScriptServer Directory in both PostScript and PDF formats.

Responding affirmatively to the final prompt of the installation will invoke the SSV$CONFIG command procedure. This will allow you to create any LAT ports you wish to use, generic queues, print queues, etc. The procedure is described in detail in Chapter 3.

2.2.2 Quick Installation Option

By utilizing the Quick Install option of the software, you avoid all prompts generated during a regular installation except key installation and queue creation. Additionally, the ScriptServer Fax facility will not be installed, as this requires additional information from the user.

Using the Quick Install option requires that you use the ScriptServer defaults for all of those items requiring a prompt in the regular installation:

- SYS$SYSDEVICE:[SCRIPTSERVER] for the disk and directory, or the directory referenced by the logical "SCRIPTSERVER" for upgrades.
- Queue characteristics will be automatically defined. The characteristic "SCRIPTSERVER_SPOOLER" will default to 20.
- "Letter" paper size and margins measured in inches are the defaults.
- If this is an upgrade, all files included in the installation will be purged.
- All of the on-line documentation will be installed into the ScriptServer directory.
- A generic ScriptServer queue SSV$PRINT will be created.

If you do not want to accept any of these defaults (for instance if characteristics are not in use at your site, or characteristic 20 is already defined), then you will want to go ahead with the regular installation (see previous section).

2.2.3 Installing the Software License Key

After the ScriptServer files have been moved to their target locations, you are prompted for whether you wish to install the software license key. A valid license key must be installed before you can start any ScriptServer queues.

In the case of an upgrade where the software is currently running on a valid license key, providing that no features have been added to the software requiring additional modules to operate, you will not need to reenter the license key. However, if features have been added to the software with the upgrade that do require additional modules to operate, a new key is necessary and will need to be entered with the upgrade.

NOTE: *If upgrading from a ScriptServer version prior to version 5.1, you will need to install a new license key in order to accommodate changes made to the licensing scheme in ScriptServer version 5.1.*

If you wish, installation of the license key may be deferred until after the installation. To install the license key after software installation, execute the PKMS_LOAD procedure in the SCRIPTSERVER directory. For example:

```
$ @SCRIPTSERVER:PKMS_LOAD
```

The procedure will prompt you for the information contained in your software key. You should have received your software key with the media. If you have lost or misplaced your key, or require an emergency or additional key(s), contact the GrayMatter Software Sales Department.

After entering the information contained in the license key, you will be given the opportunity to correct any entry errors you may have made prior to loading the key.

2.2.4 Creating Queues During Installation

Finally, the installation procedure will prompt for whether you wish to create any ScriptServer queues. Responding YES to the prompt activates the SSV$CONFIG command procedure as described in Chapter 4.

2.2.5 Restarting Queues After Upgrade

After an upgrade all of your ScriptServer queues need to be stopped and restarted so that they can use the newer version of the ScriptServer Device Control Library. Running the following command procedure:

```
SCRIPTSERVER:SSV$RESTART_QUEUES.COM
```

will automatically stop and restart any existing ScriptServer queues.

2.2.6 Sample Installation

```
$ @SYS$UPDATE:VMSINSTAL SCRIPTSERVER MUC0:

OpenVMS VAX Software Product Installation Procedure V6.2

It is 13-NOV-1995 at 15:02.

Enter a question mark (?) at any time for help.

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

The following products will be processed:

SCRIPTSERVER V5.1
```

```
Beginning installation of SCRIPTSERVER V5.1 at 14:17

%VMSINSTAL-I-RESTORE, Restoring product save set A ...
%VMSINSTAL-I-RELMOVED, Product's release notes have been moved to SYS$HELP.

+-------------------------------------------------------------+
| ScriptServer Printing System V5.1 Installation Procedure    |
| (c) Copyright 1995 GrayMatter Software Corporation          |
+-------------------------------------------------------------+

%SCRIPTSERVER-I-SSVNOLCS, ScriptServer V5.1 license not found

This installation offers a Quick Install option in which only the key info
is requested. Other defaults used by this procedure are as follows:

1.  The default directory for this installation is SYS$SYSDEVICE:SCRIPT-
    SERVER, or the directory referenced by the logical "SCRIPTSERVER" for
    upgrades.

2.  Queue characteristics will be automatically defined. The characteristic
    "SCRIPTSERVER_SPOOLER" will default to 20 for job routing to function
    properly. If characteristic 20 is in use, reply to the next prompt with
    "No".

3.  "Letter" paper size and margins measured in inches are the defaults. To
    use international sizes, define the logical SSV$DEFAULT_SIZES to INTER-
    NATIONAL.

4.  All files included in the installation will be purged.

5.  You may create any desired queues after the installation is completed by
    running the command procedure @SCRIPTSERVER:SSV$CONFIG.COM.

6.  A generic ScriptServer queue “SSV$PRINT” will be created.

* Do you want to execute the quick install procedure [Yes] No

A full installation will continue.
```

You can install this version of the ScriptServer Printing system in the same directory and purge any files that are replaced or you can specify a new directory and leave the existing files in their present location.

NOTE: *ScriptServer Printing System upgrades will only merge the existing Script Settings into the new symbiont settings library (SPD$LIBRARY.DAT) if the files are placed in the same directory as the previous version. If another directory is used, the settings can be moved individually by using the procedure: @SCRIPTSERVER:SPD_MGR.COM.*

```
* Do you want to place ScriptServer files in SYS$SYSDEVICE:[SCRIPTSERVER]?
[YES]?
```

The ScriptServer Printing System uses queue characteristics to routeprint jobs to the PostScript language printers with the desired fontand page size combinations, and to request specific fonts and page sizes with the PRINT command.

The use of queue characteristics within ScriptServer V5.1 is optional. If you choose NOT to use characteristics with ScriptServer, reply to the next prompt with NO.

If you choose to use queue characteristics, you may wish to modify the characteristic definitions supplied with ScriptServer before executing those definitions. The characteristic "SCRIPTSERVER_SPOOLER" must have a unique value (its default is 20) for job routing to function properly. If you are NOT currently using queue characteristic number 20, reply to the next prompt with "YES", otherwise the definitions file (SCRIPTSERVER:CHAR_DEF.COM) must be edited before the characteristics are defined, and you should reply to the following prompt with "NO."

```
* Define ScriptServer queue characteristics [Yes]?
```

The ScriptServer Printing System normally uses "Letter" as the de-fault paper type, and default margins and tabs measured in inches. Customers outside the United States may wish to use "A4" as the de-fault paper size, with default margins and tabs measured in centimeters.

ScriptServer refers to these as "international" defaults. To use international defaults, reply to the following prompt with "YES."

```
* Use international defaults [No]?

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

```
%SCRIPTSERVER-I-CRECHAR, Creating queue characteristic definitions

A customized start-up file for ScriptServer (SSV$STARTUP.COM) will be moved
to the directory SYS$MANAGER: and executed. You must edit your system star-
tup file so that it executes the next time you re-boot.

%SCRIPTSERVER-I-LINKING, Linking the SCRIPT facility
%SCRIPTSERVER-I-LINKING, Linking the ScriptServer print symbiont
%SCRIPTSERVER-I-MOVEFILES, Moving files to their target directories

If you have a PKMS (tm) License Key for ScriptServer, you may load it at
this time. Or, you may load the license key after this installation is com-
plete using the procedure

SCRIPTSERVER:PKMS_LOAD.COM.

A valid License Key MUST be loaded before starting any ScriptServer queues.

* Do you want to load the License Key now [Yes]?

PKMS License Key Load Procedure for SSV051

Product Name[SSV051]:

PKMS Paper Key Management System v2.0

(c) 1991,1995 Park Software Inc.

For Help enter "?", to backup enter "\", to exit enter CTL/Z.

       DISTRIBUTOR NAME: [] :GRAYSOFT

       AUTHORIZATION CODE: [] :TBB

       PRODUCT NAME: [] :SSV051

       COMPANY NAME: [] :XYZ CORPORATION

       NODE NAME: [] :NODEA

       MODULE LIST: [] :FAX NET VAX

       START DATE: [] :27-OCT-1995

       EXPIRATION DATE: [] :31-DEC-2005

       USER COUNT: [] :0

       ALLOW CLUSTER: [] :Y

       CLUSTER ID: [*] :
```

```
       CPU TYPE: [] :0

       SERIAL NUMBER: [] :50018

       CHECKSUM A: [] :BCCVJP8X

       CHECKSUM B: [] :4ARD7T#K

IS THAT CORRECT? (Y/N): [] :Y
EXIT PROGRAM? (Y/N): [] :Y

%PKMSLOAD-I-KEYEXIST, License key already exists
Load License Key PKMS$SSV051 [Y]:

Create license key startup file: PKMS_SSV051_START.COM [Y]:

The above startup file will be called from the ScriptServer startup file so
that the license key is loaded every time the system is restarted. If you
need to reload this license key in the future then you should execute the
following line:

@SCRIPTSERVER:PKMS_LOAD SSV051

       PKMS LICENSE KEY LOAD PROCEDURE COMPLTETED

If you have a PostScript printer connected and wish to create a Script-
Server queue, you may do so at this time. Or, you may create queues after
this installation is complete using the procedure SCRIPTSERVER:SSV$CONFIG.

* Do you want to create a ScriptServer queue [Yes]? N

* Do you want to print the Installation Notes [Yes]?

* Queue name [SYS$PRINT]

Job MGR_NOTES (queue PSPRINTER, entry 840) started on PSPRINTER

+----------------------------------------------------------------+
| ScriptServer Printing System Installation Complete             |
+----------------------------------------------------------------+

       Installation of SCRIPTSERVER V5.1 completed at 15:18

VMSINSTAL procedure done at 15:18
```

2.3 AFTER SOFTWARE INSTALLATION

Queue Characteristic Definition

If you deferred the definition of queue characteristics until after installation and want to take advantage of the queue characteristic capabilities of ScriptServer, you must now edit (if necessary) and execute the queue characteristics definition procedure.

After editing to make the changes as explained in Section 2.1 *"Preparing The Installation*" on page 35 above, execute the procedure with the command:

```
$ @SCRIPTSERVER:CHAR_DEF
```

Creating ScriptServer Queues

A command procedure (SCRIPTSERVER:SSV$CONFIG.COM) is provided with ScriptServer to simplify the creation of ScriptServer print queues. To create a queue with this procedure, use the command:

```
$ @SCRIPTSERVER:SSV$CONFIG
```

This will open the menu-driven command procedure which contains prompts that ensure correct queue creation based on system requirements and availability.

For detailed information about creating ScriptServer queues, see Section 4.3 *"Creating Queues*" on page 74.

Printing the Installation Manager's Notes

If you did not do so as part of the installation procedure, you should now print a brief PostScript document that is distributed with ScriptServer: the *Installation Manager's Notes*.

This document provides a quick overview of some of the capabilities of ScriptServer and how to get started as a manager of ScriptServer. To print it, enter the command:

```
$ PRINT/QUEUE=queue-name SCRIPTSERVER:MGR_NOTES.PS
```

Editing System Startup Files

During installation of ScriptServer, a system-wide logical name is created that equates SCRIPTSERVER to the directory you designated during installation. This logical name is used to locate the various ScriptServer files by the print symbiont and the SCRIPT facility.

The installation procedure creates a command procedure in the SYS$MANAGER directory named SSV$STARTUP.COM. This procedure will create the logical name SCRIPTSERVER and should be referenced at startup.

To execute the command procedure, edit your system startup file: On VMS V4.x systems:

```
SYS$MANAGER:SYSTARTUP.COM
```

on VMS V5.x systems:

```
SYS$MANAGER:SYSTARTUP_V5.COM
```

and on VMS V6.x systems:

```
SYS$MANAGER:SYSTARTUP_VMS.COM
```

Add the following line:

```
$ @SYS$MANAGER:SSV$STARTUP
```

The SSV$STARTUP procedure also allows you to install the SCRIPT facility as a known image. You should do this if you expect heavy usage of the SCRIPT facility. The procedure has specific comments to help you do this.

If you want your system startup command procedure to create and / or start ScriptServer queues, you will need to add the appropriate INIT / QUEUE or START / QUEUE commands. The recommended format for creating ScriptServer queues with the INIT / QUEUE command is:

```
$ INIT/QUEUE/PROC=SSV$SMB/LIBR=SSV$DEVCTL/ON=device-name-

_$ /DEFAULT=(FEED,FLAG)/SCHEDULE=NOSIZE/START queue-name
```

2.4 SCRIPTSERVER FILES

Following is a list of the ScriptServer files and their locations at the conclusion of the installation. Files marked with an asterisk (*), are installed only if the Fax facility is installed (see above).

Table 2-1: SCRIPTSERVER: Directory

File	Description
CHAR_DEF.COM	Characteristics command procedure
COLOR_CUBE.PS	Sample PostScript file
DEINSTALL.COM	Deinstallation command procedure
DISK_DIRECTORY.PS	Printer hard disk directory program
DOWNLOAD.COM	Downloaded fonts / modules procedure
FONT_RENAMER.COM	Font file renaming procedure
GETTING_STARTED.PS	Getting Started with ScriptServer
MGR_NOTES.PS	Manager's Installation Notes
PERM_EHANDLER.PS	Permanent PostScript error handler
PKMS_LOAD.COM	License key load procedure
PKTEST.EXE	License validation modules
PRINTER_STATUS.PS	PostScript printer status report
REFERENCE_MANUAL.PS	The ScriptServer Reference Manual
REFERENCE_MANUAL.PDF	The Reference Manual in PDF format
RELEASE_NOTES.PS	The ScriptServer Release Notes
SCRIPT.CLD	Script command language definition
SCRIPT.EXE	Script executable image
SCRIPT.HLP	Script help file
SETPRINTERNAME.PS	Printer name setting file
*SET_PRINTER_TIME.COM	Sets time and date on the printer
SITE_DEF.COM	Site-specific characteristics procedure
SPD_MGR.COM	Default settings management procedure
SPD$LIBRARY.DAT	Print symbiont settings library

Table 2-1: SCRIPTSERVER: Directory

File	Description
SSV$ADD_PROCSETS.COM	Adds Pathworks procsets to device control library
SSV$CONFIG.COM	Queue creation procedure
SSV$CONFIG.SSV	Configuration File only added at time of installation if created during queue creation
SSV$CONVERT_LOGICALS.COM	Places existing logicals in SSV$CONFIG.SSV
SSV$EPS2FORM.EXE	Creates images for use with Soft Forms
USERS_GUIDE.PS	The ScriptServer User's Guide
USERS_GUIDE.PDF	The User's Guide in PDF format

Table 2-2:SYS$SYSTEM Directory (SYS$COMMON:[SYSEXE])

File	Description
SSV$SMB.EXE	Print symbiont executable image

Table 2-3: SYS$LIBRARY Directory (SYS$COMMON:[SYSLIB])

File	Description
SSV$DEVCTL.TLB	ScriptServer device control library

Table 2-4:SYS$MANAGER Directory (SYS$COMMON:[SYSMGR])

File	Description
SSV$STARTUP.COM	System startup command procedure
PKMS_SSV051_LOAD.COM	Software key (startup) procedure, only added at installation if created during license key loading

PART 2

USER’S GUIDE

Chapter 3

Overview

3.1 POSTSCRIPT, PCL5, AND SCRIPTSERVER

3.1.1 About PostScript

PostScript is the page description language developed by Adobe Systems Incorporated that revolutionized printing and publishing. It provides a powerful device-independent environment for creating integrated text and graphics, with the ability to change the size and orientation of characters as if they were any other graphic object. PostScript has this ability because of the way its character shapes are rendered on the page. Instead of using bitmapped fonts, which cannot be increased in size without loss of resolution, the PostScript language uses “outline fonts”, which are mathematical descriptions of the curves and strokes that make up the graphic shapes of characters.

A printer that implements the PostScript language contains one or more high speed microcomputers executing the PostScript interpreter, which translates the various language commands into a graphic image that can be rendered on that particular printer's imaging hardware. As an independent computer executing programs (PostScript files), the printer encounters various software and hardware error conditions. The printer reports these errors back to the host if the communications link in use allows it.

3.1.2 About PCL and PJL

PCL

PCL is a printer language created by Hewlett-Packard that allows applications to control a variety of printer features across a number of different printing devices. PCL commands consist of escape sequence codes embedded in the print job data stream. Implementing the language in this way minimizes both transmission time and command decoding overhead. The PCL printer language has evolved through five major levels of functionality:

- Print and Space
- EDP (Electronic Data Processing) / Transaction
- Office Word Processing
- Page Formatting and
- Office Publishing

The current version of PCL and that supported by ScriptServer is known as PCL5. There are four general types of HP printer language commands:

- Control Codes
- PCL Commands
- HP-GL / 2 Commands and
- PJL Commands

Control codes initiate printer functions. PCL commands control all of the printer’s features except those used for vector graphics which are controlled by HP-GL / 2 commands. PJL commands are described below.

NOTE: *See the PCL5 Printer Language Technical Reference Manual from Hewlett-Packard for a more detailed discussion of PCL.*

PJL

Unlike PCL and HP-GL / 2 commands, PJL commands supply job-level control. One of the major features of PJL is the ability to switch printer languages between jobs. For example, applications supporting PJL can print one job using PCL, and the next using PostScript or another language - without any operator intervention.

PJL also provides two-way communication with the printer which allows the host to receive information from the printer such as printer model, configuration, printer status, and job status. Additionally, PJL commands signal the beginning and end of the print job and include commands that dictate the number of copies to be printed, the page orientation, and whether or not to duplex print the job, etc.

NOTE: *See the PJL Technical Reference Manual from Hewlett-Packard for a more detailed discussion of PJL.*

3.1.3 THE SCRIPTSERVER PRINT SYMBIONT

The ScriptServer print symbiont is the component of the ScriptServer Printing System that communicates directly with the printer and provides support for all PCL5 printers manufactured by Hewlett-Packard as well as all PostScript compatible printers. It is via the symbiont that files reach the printer and likewise that error and information messages from the printer reach the host CPU.

When they occur, these messages are reported to the user through the ScriptServer Print Log. See Section 3.4.1 *"THE SCRIPTSERVER PRINT LOG*" on page 57 for more information about the Print Log.

At startup, the symbiont queries the printer to determine type. If the printer is determined to bePCL5 or PCL / PJL compatible, the symbiont will open the appropriate connection and further query the printer using various PJL commands to determine its various capabilities. If determined to be PostScript, the software conducts a dialog with the PostScript interpreter on the printer in order to open the appropriate connection for communication. Even without being able to conduct this dialogue, the symbiont can still open a connection to the printer through which data is sent.

3.1.4 Printing Supported File-types

The ScriptServer Printing System can print any PostScript file to any PostScript printer and any PCL file to any PCL5 printer or any printer supporting both PCL and PJL commands. PostScript and emulation jobs printed to PostScript-only printers are passed directly to the printer, all others are converted to PostScript. PCL files printed to PCL / PJL printers or printers supporting PCL5 are encapsulated with the appropriate PJL commands and sent to the printer. In these cases, flag pages, print logs, page count, status, etc. are also managed using PCL5, PJL, and HP-GL / 2.

Some printers provide emulations that allow the printing of non-PostScript or non-PCL files. Often, these emulations can be invoked through software-switching, which allows the host to change the emulation mode on the printer with special commands. With the proper commands, ScriptServer can provide this software switching capability.

The logical:

```
SSV$queue_ASSUME_PCL
```

has been included to allow more flexibility when printing PCL files. By default, ScriptServer assumes that escape sequences sent to a PCL printer are PCL-based escape sequences. However, defining this logical to FALSE:

```
$ DEFINE/SYSTEM SSV$queue_ASSUME_PCL FALSE
```

where “queue” is the queue through which you are sending the print job, allows you to send non-PCL escape-sequence-based emulations to the printer while in software switching mode.

3.1.5 File-type Translation

The ScriptServer Printing System also contains provisions allowing you to print ASCII files as PostScript or PCL and HPGL as PostScript. This is accomplished through translation. The conversion process is handled automatically by the ScriptServer print symbiont, and the manner in which the conversion is performed is determined by how the print request was created. ScriptServer will convert ASCII text files to PostScript for printing when possible. However, when PostScript is not available on the output device, ScriptServer will convert the text to PCL and submit the job. This behavior can be altered through use of the logical:

```
SSV$TEXT_LANGUAGE
```

3.1.6 Connectivity

The ScriptServer Printing System supports all of the following connections:

- Serial
- Parallel
- LAT
- TCP / IP
- lpd
- DECnet

Through these different connection types, the symbiont can print to virtually any PostScript, PCL5, or PCL / PJL supported printer on your network.

3.2 CREATING PRINT REQUESTS

There are two commands that may be used to print files with ScriptServer:

- the standard PRINT command and
- the SCRIPT command

Print PostScript, PCL, or HP-GL files with the PRINT command. ASCII files can be printed using either command, but control over formatting parameters is much easier with SCRIPT than with PRINT, however the SCRIPT facility can only be used when converting a text file into PostScript.

3.2.1 Using the PRINT Command

When handling print requests created with the PRINT command, ScriptServer determines the type of data in the print file, which dictates the appropriate translator if required. If there is no default data type assigned to the queue, the symbiont assumes that the file is an ASCII file requiring translation.

This assumption, however, may be overridden based on the following criteria:

- the file type file extension
- a file type of ".PS" indicates that this is a PostScript file
- a file type of ".HP" indicates an HP-GL file
- a file type of ".PCL" indicates a PCL file
- the first five records of the file: if it finds Escape sequences, it will determine the file to be PCL
- if it finds "%!" it will determine the file to be PostScript
- if it finds neither, it will determine the file to be ASCII text, performing translation to either PCL or PostScript depending on printer capability and / or the value of the logical:

```
SSV$TEXT_LANGUAGE
```

- the DATA_TYPE parameter to the PRINT command - which can be used to specify the type of print data in the file

Each of these overrides the previous criterion. In the following example the file has a file type of "HP" but, since it is submitted to a PostScript printer with the DATA_TYPE=ASCII parameter, the ASCII-to-PostScript translator will be used:

```
$ PRINT/PARAM=DATA_TYPE=ASCII file-name.HP
```

According to the Adobe standard, "%!" is used to indicate that a file is PostScript.Use the logical:

```
SSV$queue_PS_CLUE
```

to tell ScriptServer to send some other string to the printer indicating that the subsequent file is PostScript. Additionally, you can define this logical to call a setup module and send the contents of that file appended to the front of the PostScript file. See the discussion in Chapter 18 for a more detailed description of this logical.

For more information on using the PRINT command with ScriptServer, see Chapter 10.

```
SSV$OVERRIDE
```

When it is not possible to control the file's name, the first five records do not contain "%!", and there is no control over the print parameters specified when the file is submitted for printing on a PostScript printer, it is necessary to specifically identify that a file is already in PostScript, and that no translation is to be performed on it. This is done by printing a file using the special form name:

```
SSV$OVERRIDE.
```

To set up your system so that you can use SSV$OVERRIDE, you must define the form using the DEFINE/FORM command. In most circumstances, the /STOCK=DEFAULT qualifier should also be used, so that the file will be printed even when the currently mounted form is DEFAULT.

To set up the form SSV$OVERRIDE:

1. Using the SHOW QUEUE/FORM command, find a form number that is not currently in use.
2. Execute the following command:

   ```
   $ DEFINE/FORM/STOCK=DEFAULT SSV$OVERRIDE number
   ```

   Where number is an unused form number from the first step.

NOTE: *Using SSV$OVERRIDE disables ScriptServer's processing of non-PostScript print files. It is only recommended in environments where all of the files to be printed are known to be in PostScript.*

3.2.2 Using the SCRIPT Command

The SCRIPT facility is used when more control over ASCII-to-PostScript translation is required. With SCRIPT you select all the parameters that describe how your file is printed, including: fonts, font sizes, line spacing, heading text, and more. Any file submitted with the

SCRIPT command will be treated as an ASCII file regardless of file-type and the symbiont will simply print the contents of the file.

For more information on using the SCRIPT facility, see Chapter 9.

NOTE: *The SCRIPT Facility is for use with PostScript printers only and will not allow you to submit jobs to PCL5 or PCL/PJL supported printers.*

3.3 SCRIPTSERVER SOFT FORMS

The ScriptServer Soft Forms utility (SSV$EPS2FORM) allows you to create page descriptions (containing graphics, text, images, etc.) and use the resulting image as a "backdrop" to text files printed with the ScriptServer Printing System. Page descriptions can be created with any software that generates Encapsulated PostScript (EPS) files, such as Adobe Illustrator, Freehand, and PageMaker. This EPS file is converted to a PostScript Level 2 form which is called by the ScriptServer print symbiont at the beginning of every page. The resulting "soft form" can be associated with an OpenVMS printer form and automatically generated when a user prints a text file with that form.

For a more detailed discussion of ScriptServer SoftForms, see Chapter 13.

3.4 ERROR MESSAGING

3.4.1 THE SCRIPTSERVER PRINT LOG

ScriptServer monitors messages received from the printer and has the ability to print a Print Log at the conclusion of the print job that generated them. The Print Log lists all of the job-specific messages received from the printer as well as the record number of the file being processed when the message is detected.

PostScript Print Logs

PostScript errors received by ScriptServer are considered fatal when they are accompanied by the following message from the printer:

```
Flushing: rest of file (to end-of-job) will be ignored
```

If this message is detected by ScriptServer, no further data from the input file is sent to the printer and the job terminates with the error:

```
DEVCMD, device command error.
```

This behavior can be modified for special cases by using the logical:

```
SSV$queue_PASSALL_MODE
```

The logical might be used for printing PostScript files containing multiple PostScript jobs intended to have sections of the print data flushed. For more information on messages printed in the Print Log, see Chapter 11.

For more information on the SSV$PASSALL_MODE logical see Chapter 12.

You can also control when ScriptServer Print Logs are printed with the logical:

```
SSV$queue_PRINTLOG_LEVEL
```

which is also discussed in detail in Chapter 12.

PCL Print Logs

When associated with PCL print jobs, ScriptServer print logs will be printed in PCL as well. These differ from PostScript print logs only in layout. PostScript print logs are centered by default, because PCL has no inherent centering capability, print logs associated with PCL print jobs will be left justified.

3.4.2 PCL ERROR CODES

PCL errors come from the printer as five digit codes. ScriptServer interprets the codes and reports them appropriately (see Chapter 16). Continuable errors are reported via the print log. Errors requiring operator intervention will be reported via OPCOM.

3.5 QUEUE CHARACTERISTICS IN SCRIPTSERVER

When ScriptServer is installed, the installer may optionally create queue characteristics that associate the various printer capabilities such as font names, paper sizes, color, and duplexing with numeric values from 0 to 127. ScriptServer Printing System utilities (such as the SCRIPT facility) associate these values with the print job whenever a print request is queued. This ensures that the required fonts and paper are available in multiple-printer environments.

Queue characteristics with ScriptServer are optional, (but recommended) and may not be in use at your site. To determine if ScriptServer is using characteristics at your site, enter the command:

```
$ SHOW QUEUE/CHAR SCRIPTSERVER_SPOOLER
```

If the system displays “No such characteristic”, the ScriptServer software is not using characteristics at your installation. ScriptServer uses the queue characteristics feature of the OpenVMS Queue Manager to route print jobs to printers with the necessary fonts, paper sizes, and printer languages. When a queue characteristic is associated with a print request, the queue (printer) on which that print job executes must have the same queue characteristic values associated with it in order to print.

Queue characteristics are also used within ScriptServer to relay PostScript printer-specific font information to the SCRIPT facility. When characteristics are enabled, SCRIPT’s font selection menu accurately displays the fonts that are resident in the printer and those available for downloading.

3.6 POSTSCRIPT FONTS SUPPORTED

The ScriptServer Printing System supports a wide range of PostScript fonts and paper sizes in addition to downloadable fonts which are site-defined. Table 3-1: on page 59 lists supported printer-resident PostScript fonts.

NOTE: *It is possible to add site-specific fonts to the supported font list using the SSV$FONTS logical name in the SSV$DOWNLOAD_TABLE logical name table.*

At queue startup, the ScriptServer print symbiont determines if PostScript fonts in the standard support list (Table 3-1: on page 59) and fonts in the SSV$FONTS search list are available, setting the appropriate characteristics if they are. See Chapter 7 for information on font management and downloading.

Table 3-1:Supported Printer Resident Fonts

OpenVMS Characteristic Name	PostScript Name
AGARAMOND_REGULAR	AGaramond-Regular
AGARAMOND_ITALIC	AGaramond-Italic

Table 3-1:Supported Printer Resident Fonts

OpenVMS Characteristic Name	PostScript Name
AGARAMOND_SEMIBOLD	AGaramond-Semibold
AGARAMOND_SEMIBOLDITALIC	AGaramond-SemiboldItalic
AGARAMOND_BOLD	AGaramond-Bold
AGARAMOND_BOLDITALIC	AGaramond-BoldItalic
AVANTGARDE_BOOK	AvantGarde-Book
AVANTGARDE_BOOKOBLIQUE	AvantGarde-BookOblique
AVANTGARDE_DEMI	AvantGarde-Demi
AVANTGARDE_DEMIOBLIQUE	AvantGarde-DemiOblique
BOOKMAN_LIGHT	Bookman-Light
BOOKMAN_LIGHTITALIC	Bookman-LightItalic
BOOKMAN_DEMI	Bookman Demi
BOOKMAN_DEMIITALIC	Bookman-DemiItalic
COURIER	Courier
COURIER_OBLIQUE	Courier-Oblique
COURIER_BOLD	Courier-Bold
COURIER_BOLDOBLIQUE	Courier-BoldOblique
GARAMOND_LIGHT	Garamond-Light
GARAMOND_LIGHTITALIC	Garamond-LightItalic
GARAMOND_BOLD	Garamond-Bold
GARAMOND_BOLDOBLIQUE	Garamond-BoldOblique
HELVETICA_BLACK	Helvetica-Black
HELVETICA_BLACKOBLIQUE	Helvetica-BlackOblique
HELVETICA	Helvetica
HELVETICA_OBLIQUE	Helvetica-Oblique
HELVETICA_BOLD	Helvetica-Bold
HELVETICA_BOLDOBLIQUE	Helvetica-BoldOblique
HELVETICA_CONDENSED	Helvetica-Condensed
HELVETICA_CONDENSED_BOLD	Helvetica-Condensed-Bold

Table 3-1:Supported Printer Resident Fonts

OpenVMS Characteristic Name	PostScript Name
HELVETICA_CONDENSED_OBLIQUE	Helvetica-Condensed-Oblique
HELVETICA_CONDENSED_BOLDOBLIQUE	Helvetica-Condensed-BoldOblique
HELVETICA_LIGHT	Helvetica-Light
HELVETICA_LIGHTOBLIQUE	Helvetica-LightOblique
HELVETICA_NARROW	Helvetica-Narrow
HELVETICA_NARROW_OBLIQUE	Helvetica-Narrow-Oblique
HELVETICA_NARROW_BOLD	Helvetica-Narrow-Bold
HELVETICA_NARROW_BOLDOBLIQUE	Helvetica-Narrow-BoldOblique
KORINNA_REGULAR	Korinna-Regular
KORINNA_KURSIVREGULAR	Korinna-KursivRegular
KORINNA_BOLD	Korinna-Bold
KORINNA_KURSIVBOLD	Korinna-KursivBold
LUBALINGRAPH_BOOK	LubalinGraph-Book
LUBALINGRAPH_DEMI	LubalinGraph-Demi
LUBALINGRAPH_BOOKOBLIQUE	LubalinGraph-BookOblique
LUBALINGRAPH_DEMIOBLIQUE	LubalinGraph-DemiOblique
NEWCENTURYSCHLBK_ROMAN	NewCenturySchoolbook-Roman
NEWCENTURYSCHLBK_ITALIC	NewCenturySchoolbook-Italic
NEWCENTURYSCHLBK_BOLD	NewCenturySchoolbook-Bold
NEWCENTURYSCHLBK_BOLDITALIC	NewCenturySchoolbook-BoldItalic
PALATINO_ROMAN	Palatino-Roman
PALATINO_ITALIC	Palatino-Italic
PALATINO_BOLD	Palatino-Bold
SOUVENIR_LIGHT	Souvenir-Light
SOUVENIR_LIGHTITALIC	Souvenir-LightItalic
SOUVENIR_DEMI	Souvenir-Demi
SOUVENIR_DEMIITALIC	Souvenir-DemiItalic
SYMBOL	Symbol

Table 3-1:Supported Printer Resident Fonts

OpenVMS Characteristic Name	PostScript Name
TIMES_ROMAN	Times-Roman
TIMES_ITALIC	Times-Italic
TIMES_BOLD	Times-Bold
TIMES_BOLDITALIC	Times-BoldItalic
ZAPFCHANCERY_MEDIUMITALIC	ZapfChancery-MediumItalic
ZAPFDINGBATS	ZapfDingbats

NOTE: *ScriptServer's PCL/PJL support does not include support for font downloading or full re-encoding of fonts. Additionally, PCL uses ISO Latin-1 Character Set rather than DEC Multi-national Character Set which may result in some subtle font substitution.*

3.7 PAPER SIZES SUPPORTED

The following table summarizes the paper sizes and keywords supported by the ScriptServer Printing System. All paper types except envelopes have fixed pre-defined sizes. There are approximately 20 different envelope sizes supported by the various PostScript language printers with envelope feed support. Since such a wide variety of envelope sizes are supported, the site manager must establish the logical name:

```
SSV$queue_ENVELOPE_COMMAND
```

to support sizes other than the default size of 4.125 by 9.5 inches. Additionally, certain printers may require the definition of this logical name even to support the default envelope size.

ScriptServer software does not support printing on envelopes when printing in PCL mode. Support for printing on envelopes in PostScript mode is discussed in greater detail in Chapter 8.

By default, ScriptServer uses North American paper sizes in both the SCRIPT menu and when dynamically selecting paper-trays for printing. To activate International paper sizes, simply use the following logical name definition:

```
SSV$DEFAULT_SIZES INTERNATIONAL
```

NOTE: *This logical also affects the default units of measure for ScriptServer size expressions.*

Table 3-2:Supported Paper Sizes

Paper Keyword	Paper Size
ENVELOPES	4.125 by 9.5 inches (default)
LETTER	8.5 by 11 inches
LEGAL	8.5 by 14 inches
TABLOID	17 by 11 inches
LEDGER	11 by 17 inches
A3	297 by 420 millimeters
A4	210 by 297 millimeters
A5	148 by 210 millimeters
B4	257 by 364 millimeters
B5	182 by 257 millimeters

3.8 SCRIPTSERVER CONFIGURATION FILE

ScriptServer provides a configuration file for easy configuration and management of your ScriptServer logical name definitions. While these logicals can still be defined within the System Table and included in a Startup file, including them in the file:

```
SCRIPTSERVER:SSV$CONFIG.SSV
```

has the same results while centralizing their location. Use of this file is discussed in detail in Chapter 4.

Chapter 4

ScriptServer Queues

4.1 OVERVIEW

ScriptServer supports nine types of queues, the majority of which are based on connection type:

- serial
- LAT
- parallel
- DECnet
- TCP/IP
- lpd
- Foreign queue
- Repository
- Generic

There are two ways to go about creating these queues with the ScriptServer software:

- INIT/QUEUE - the standard OpenVMS command for initializing standard print queues
- SSV$CONFIG.COM - a command file provided with the software to facilitate the process

Before queue creation can begin, however, you must connect your system to your printer(s). Once this has been completed, you can then initialize, start, modify, and delete your ScriptServer queues.

4.2 CONNECTING A PRINTER

ScriptServer supports the connection of printers to OpenVMS systems using any of the connection methods listed above in section Section 4.1 *"OVERVIEW"* on page 65. The following setup procedures for establishing printer connections assume enough privileges to manipulate system hardware.

NOTE: *All of the following connection procedures are handled automatically by the command procedure:*

```
SCRIPTSERVER:SSV$CONFIG.COM
```

described below and included with the software

4.2.1 Setting Up a Parallel Connection

Parallel connections are strictly one-way communications paths that provide high-speed data throughput to the printer but allow no way for messages from the printer to be transmitted back to the host. With a parallel connection no dialogue between printer and host can be conducted at queue startup, and error messages and conditions reported by the printer cannot be relayed to the host.

NOTE: *If the parallel connection is on an LP port (a terminal driver), Backchannel is automatically disabled, otherwise it must be disabled manually as described below.*

Identifying Printers Without Backchannels

When a parallel printer port or a serial printer without backchannel capability is being used, it must be identified as such with the following logical definition:

```
$ DEFINE/SYS SSV$device_BACKCHANNEL FALSE
```

where “device” is the name of the port to which the printer is connected.

In instances where characteristics are in use, but there is no backchannel, the symbiont will supply a default set of characteristics which make the printer appear as a PostScript LaserWriter to the symbiont using the characteristic number for the default paper size as determined by the value of the logical:

```
SSV$DEFAULT_SIZES
```

If set to INTERNATIONAL, the characteristic list will indicate “A4” as the paper size. If not, the characteristic list will indicate “letter”. Additionally, the characteristic list will indicate the characteristics corresponding to the standard 39 PostScript fonts:

Table 4-1:Standard PostScript Fonts

OpenVMS Characteristic Name	PostScript Name
AVANTGARDE_BOOK	AvantGarde-Book
AVANTGARDE_BOOKOBLIQUE	AvantGarde-BookOblique
AVANTGARDE_DEMI	AvantGarde-Demi
AVANTGARDE_DEMIOBLIQUE	AvantGarde-DemiOblique
BOOKMAN_LIGHT	Bookman-Light
BOOKMAN_LIGHTITALIC	Bookman-LightItalic
BOOKMAN_DEMI	Bookman Demi
BOOKMAN_DEMIITALIC	Bookman-DemiItalic
COURIER	Courier
COURIER_OBLIQUE	Courier-Oblique
COURIER_BOLD	Courier-Bold
COURIER_BOLDOBLIQUE	Courier-BoldOblique
HELVETICA	Helvetica
HELVETICA_OBLIQUE	Helvetica-Oblique
HELVETICA_BOLD	Helvetica-Bold
HELVETICA_BOLDOBLIQUE	Helvetica-BoldOblique
HELVETICA_CONDENSED	Helvetica-Condensed
HELVETICA_CONDENSED_BOLD	Helvetica-Condensed-Bold
HELVETICA_CONDENSED_OBLIQUE	Helvetica-Condensed-Oblique
HELVETICA_CONDENSED_BOLDOBLIQUE	Helvetica-Condensed-BoldOblique
HELVETICA_LIGHT	Helvetica-Light
HELVETICA_LIGHTOBLIQUE	Helvetica-LightOblique
HELVETICA_NARROW	Helvetica-Narrow
HELVETICA_NARROW_OBLIQUE	Helvetica-Narrow-Oblique
HELVETICA_NARROW_BOLD	Helvetica-Narrow-Bold

Table 4-1:Standard PostScript Fonts (Continued)

OpenVMS Characteristic Name	PostScript Name
HELVETICA_NARROW_BOLDOBLIQUE	Helvetica-Narrow-BoldOblique
NEWCENTURYSCHLBK_ROMAN	NewCenturySchoolbook-Roman
NEWCENTURYSCHLBK_ITALIC	NewCenturySchoolbook-Italic
NEWCENTURYSCHLBK_BOLD	NewCenturySchoolbook-Bold
NEWCENTURYSCHLBK_BOLDITALIC	NewCenturySchoolbook-BoldItalic
PALATINO_ROMAN	Palatino-Roman
PALATINO_ITALIC	Palatino-Italic
PALATINO_BOLD	Palatino-Bold
SYMBOL	Symbol
TIMES_ROMAN	Times-Roman
TIMES_ITALIC	Times-Italic
TIMES_BOLD	Times-Bold
TIMES_BOLDITALIC	Times-BoldItalic
ZAPFDINGBATS	ZapfDingbats

NOTE: *If no backchannel is available, the symbiont is unable to determine the printer's capabilities for declaration of characteristics. In situations where queue characteristics other than the default set used by the symbiont are required on a printer without a backchannel, they must be set manually with the logical definition:*

```
$ DEFINE/SYSTEM SSV$queue_CHARACTERISTICS “#,#,#,...”
```

where “queue” is the name of the queue to which you wish to add these characteristics and “#,#,#,...” are the characteristic numbers you wish to add.

If characteristics are not in use on your system, it is still possible to indicate that a printer is PCL5compatible with the following logical definition:

```
$ DEFINE/SYSTEM SSV$device_PCL_SUPPORT TRUE
```

where “device” is the device name of the printer in question. If the device in question supports only a limited set of PJL commands including the universal exit command, the enter language command, and PJL comments, then define the logical to LIMITED as follows:

```
$ DEFINE/SYSTEM SSV$device_PCL_SUPPORT LIMITED
```

The logical:

```
SSV$LIMITED_PRODUCTS
```

accomplishes the same thing on a system-wide basis, but requires bi-directional communication. If the device supports no PJL commands, define the logical to NOPJL. If the queue doesn’t support PJL or PCL, then define the logical to FALSE. See Chapter 18 for details on this and other logicals.

Conversely, if a printer is a PostScript-only printer you can indicate as much with the following logical definition:

```
$ DEFINE/SYSTEM SSV$device_POSTSCRIPT_SUPPORT TRUE
```

If a printer is a color printer and the device has no way of communicating this to the symbiont the following logical definition is required in order to indicate the fact to the symbiont:

```
$ DEFINE/SYSTEM SSV$device_COLOR TRUE
```

where “device” is the color printer.

4.2.2 Setting Up a Serial Connection

After you have connected the printer's communication cable to the serial port, enter the command:

```
$ SET TERM/NOECHO/NOAUTO/SPEED=baud-rate/PERM term-port
```

This command should also be added to the system startup file thus ensuring that the terminal port is set properly the next time the system is rebooted. It is good policy to verify that communications are intact by verifying your serial connections using the procedure described below.

Verifying Serial Connections to PostScript Printers

After connecting your printer to a serial port, you should verify that it is communicating properly. To test the serial connection of a PostScript printer:

1. Make sure the device is not SPOOLED with the command:

   ```
   $ SET DEVICE/NOSPOOL port
   ```

2. Enter the command:

   ```
   $ SET HOST/DTE port
   ```

   where “port” is the serial port to which the PostScript printer is attached.

3. After connecting to the printer successfully, press CTRL / T. If connected properly, the printer will respond with a one-line message describing its current status.

   ```
   %%[ status: waiting; source: serial 25 ]%%
   ```

   If you do not receive a status message similar to the one above, refer to Chapter 14.

4. Once you have received the status message, you should send an end-of-job signal to the printer by pressing CTRL / D.

5. Finally, you can exit the procedure by pressing CTRL / \.

   The procedure with responses should be similar to that shown here for device TXA0:

   ```
   $ SET DEV/NOSPOOL TXA0:
   $ SET HOST/DTE TXA0:
   %REM-I-TOEXIT, connection established, type ^\ to exit
   <CTRL/T> (generates status line)
   %%[ status: waiting; source: serial 25 ]%%
   <CTRL/D>   (signals end-of-job to printer; not echoed)
   <CTRL/\>   (control-backslash to exit SET HOST)
   ```

PostScript Printers not Supporting CTRL/T

For parallel printers without Backchannel support, ScriptServer automatically disables the CTRL / T functionality. On serial connected printers that don’t support CTRL / T, you need to do so manually with the following logical definition.

```
$ DEFINE/SYSTEM SSV$device_CTRLT_SUPPORT FALSE
```

With this definition you can bypass the device inactive error message that would normally result.

4.2.3 Setting Up a LAT Connection

First configure your terminal server for remote communication on the printer's designated port (consult your terminal server documentation for how to do this). Once this has been done, create a LAT port on each of the VAX hosts that will be configured with ScriptServer print queues using the following commands.

```
$ LCP:==$LATCP
$ LCP CREATE PORT LTA9999
$ LCP SET PORT/NODE=node-name/PORT=port-name LTA9999
```

NOTE: *This procedure assumes that LAT port LTA9999: is not in use and does not exist. If this is not the case on your system, assign a different value and substitute it for "LTA9999" in the commands above.*

In the last command, "node-name" is the node name of the terminal server to which the printer is attached and "port-name" is the name of the printer's terminal server port. Add these commands to the system startup file so that the LAT port is created and properly configured the next time the system is rebooted. You can verify that the printer is set up properly using the same verification process as described above for verifying a serial connection.

NOTE: *The verification procedure doesn't allow you to connect to a LAT port in releases prior to OpenVMS V5.1.*

For proper interaction with printers attached to serial LAT ports, the terminal server port must have the queuing characteristic disabled. Otherwise, the ScriptServer print symbiont may think it is connected to the printer when it is actually in the port's connection queue.

4.2.4 Setting up a DECnet Connection

Currently, only printers implementing QMS Crown technology are supported over DECnet connections. Additionally, QMS printers connected by DECnet do not provide a backchannel for status information from the printer. As a result, accounting data does not include pages printed. In addition, there is no way for the symbiont to obtain font, paper, and other information at queue startup. See the discussion concerning how to setup a Parallel connection above for the implications of using a device without backchannel.

However, accounting data can be obtained in full by using an alternate serial BACKCHANNEL described below.

4.2.5 Setting up a TCP/IP Connection

There is nothing special required by ScriptServer when setting up a TCP/IP connection. However, you must make sure you have a TCP/IP stack installed on your system and that the TCP/IP protocol is supported by your printer. Refer to your TCP/IP software and printer documentation respectively for details regarding these requirements.

LPD Support for TCP/IP Printers

LPD support for TCP/IP connections is dictated by the network card or printer in use. Connecting to a network printer with lpd support is virtually no different than connecting a regular TCP/IP printer.

However, once a connection is established, one of the TCP/IP protocol options becomes packet coalescence, where the send operation will be delayed to fill up the send buffer for more efficient I/O operations. Whether or not packets are coalesced is determined by the combination of the TCP/IP stack being used and the TCP/IP card on the printer. By default, packet coalescence is turned off. To enable packet coalescence on a device-specific or system-wide basis use the following logical name definitions:

- Device-specific:

```
$ DEFINE/SYSTEM SSV$device_LPD_COALESCE TRUE
```

- System-wide for all ScriptServer lpd queues:

```
$ DEFINE/SYSTEM SSV$LPD_COALESCE TRUE
```

Alternate LPD implementation

Setting the following logical to FALSE:

```
$ DEFINE/SYSTEM SSV$device_LPD_READ FALSE
```

where “device” is the name of the queue to which you want the logical to apply, results in a more flexible, but non-standard implementation of lpd. In this incarnation, ScriptServer won’t read and wait for acknowledgments in the lpd protocol. You would normally only use the logical for debugging purposes.

gethostbyname Support for TCP/IP Printers

Support for *gethostbyname* works for any TCP/IP connection/protocol combination. This feature maps TCP/IP addresses to user-defined names. However, a host-name/address resolution must exist either in the local host table or your network must have a host-name Resolver (like DNS) that will return the host-address by issuing the *gethostbyname* call from the standard TCP / IP socket libraries.

4.2.6 Alternate Serial Backchannel for DECnet and TCP/IP Connections

ScriptServer supports the use of an alternate serial backchannel for those TCP / IP and DECnet printers for which direct backchannel is unavailable. This alternate serial backchannel is intended to provide Startup Dialogue / Error Reporting / Accounting Data for these printers.

- For QMS Crown technology printers, spooling must be set to “ON IDLE” in the TCP / IPDECnet communications submenu on the printer.
- Define the following logical where “device” is the IP address or DECnet node name of the printer and “port” is the serial port to be used as the BACKCHANNEL

```
$ DEFINE/SYSTEM SSV$device_BACKCHANNEL port
```

A status message on startup will indicate that the alternate channel is being used.

4.2.7 The Printer Display

When applicable, the symbiont contains mechanisms to manipulate the printer’s LED display to show various information associated with the print job (i.e. user name and node name).

Defining this logical system-wide:

```
$ DEFINE/SYSTEM SSV$INHIBIT_DISPLAY FALSE
```

allows ScriptServer to display this information. To define this logical on a device specific basis, define the logical in the form:

```
$ DEFINE/SYSTEM SSV$device_INHIBIT_DISPLAY FALSE
```

where “device” is the name of the printer for which you want to manipulate the LED display. The ScriptServer logical:

```
SSV$DISPLAY_PRODUCTS
```

is used system wide for the same purpose. This logical assumes bi-directional communication capability in order to recognize product names that might be included in this list. The logical name definition contains a list of product names that are capable of displaying PJL messages on an LED display. Product names should be defined in exactly the same way as appears on the status page from the printer, or by using a wild card.

The wild card character (*) indicates that the product name must contain merely the string within the quotation marks to be recognized by the symbiont. For example the definition:

```
DEF/SYS SSV$DISPLAY_PRODUCTS “*HP LASERJET 4”
```

includes the entire family of Hewlett Packard Laser Jet 4 printers. The equivalence value is not case sensitive, but if the string includes spaces it must be contained within quotation marks.

A product having its name included in this list is the equivalent of having the SSV$device_INHIBIT_DESCRIPTION logical set to FALSE for that device, meaning that PJL messages will be displayed on the LED display of the printer. The device-specific logical takes precedence in the event of a conflict.

This logical is defined by default in the SSV$CONFIG.SSV configuration file at installation time. See below for instructions on using this file and manipulating the logicals within it.

NOTE: *To check the product name of a particular printer, print the file:*

```
SCRIPTSERVER:PRINTER_STATUS.PS
```

to the printer in question.

4.3 CREATING QUEUES

ScriptServer queues of all types may be created in either of two ways:

- Through the OpenVMS INIT/QUEUE command
- Through the ScriptServer command procedure, SSV$CONFIG

4.3.1 Using the INIT/QUEUE Command

When creating ScriptServer print queues with INIT/QUEUE, the following qualifiers are required with the specified arguments:

- /PROCESSOR=SSV$SMB
- /LIBRARY=SSV$DEVCTL
- /DEFAULT=FEED
- $ INIT/QUEUE/PROC=SSV$SMB/LIBR=SSV$DEVCTL/DEFAULT=FEED queue-name

Where “queue-name” is the name of the queue you are creating. Additionally, you may also want to include the qualifier:

- /ON=device_name

NOTE: *All of these qualifiers are automatically used when creating queues with the SSV$CONFIG command procedure (see below).*

The /ON Qualifier

For standard queues, the argument to the /ON qualifier is the name of the device to which the PostScript printer is connected:

```
$ INIT/QUEUE/PROC=SSV$SMB/LIBR=SSV$DEVCTL/DEFAULT=FEED -

_$ /ON=device_name PSQUEUE
```

This is critical to the functioning of your ScriptServer queue. Where you point the qualifier is where you point the queue, and this determines your ScriptServer queue type. To result in a queue of a particular type, the argument of the /ON qualifier must be of a format appropriate to that type of queue:

Table 4-2:Connection Type Device Names

Connection Type	Device Name Used	Examples
LAT	LTA port	LTA99:
serial	TX or TT port	TXA0:
parallel	LP port	LPA0:
DECnet	DECnet Printer Node Name	VAX::PTR::
TCP/IP	TCP/IP address or Printer Host Name	100.90.10.22 or jingle.bell.com lpd lpd server queue see below
Foreign	Queue Name	PS$QUEUE
Repository	Directory Specification	MUB0:[USER]

For example, to create PSQUEUE on LAT device LTA100, you would use the following command:

```
$ INIT/QUEUE/PROC=SSV$SMB/LIBR=SSV$DEVCTL/DEFAULT=FEED/ON=LTA100: -
_$ PSQUEUE
```

TCP/IP Queues

When initializing TCP/IP queues, the argument to the “ON” qualifier must be the printer’s TCP/IP address or printer host name. The TCP/IP address may be any of the following:

- four numbers, each less than or equal to 255 separated by periods, for example, 192.100.10.5. On most printers with TCP/IP capability, this address can be assigned by manipulating the printer's front panel menu. To create a TCP/IP queue on a printer assigned to TCP/IP address 192.100.20.5 you would enter:

```
$ INIT/QUE/PROC=SSV$SMB/LIBR=SSV$DEVCTL/DEFAULT=FEED -
_$/ON="192.100.20.5" PSQUEUE
```

Double quotation marks are required around the TCP/IP address when using INIT/QUE.

- Alternatively, the printer host name may be used (see Section 4.2.5 *"Setting up a TCP/IP Connection*” on page 72 above for a discussion of printer host names).
- A logical name can also be created that equates to the printer’s TCP/IP address, and that can also be used as the argument to the /ON qualifier. For example, the following logical definition equates the logical name TORONTO to the printer with TCP/IP address 100.192.34.5:

```
$ DEFINE/SYSTEM TORONTO “100.192.34.5”
```

Before starting your TCP/IP queue, you must determine the correct TCP/IP protocol for communication with your printer. If the protocol in use is other than Telnet, you need to define a logical in the following format:

```
SSV$address_TCPIP_PROTOCOL protocol_type
```

Where “address” is the TCP/IP address of the printer in question and “protocol_type” is the type of TCP/IP protocol in use on the system. ScriptServer currently supports three protocol types: HP, QMS, and Telnet, with the following qualifications.

Table 4-3:TCP/IP Protocol Summary

Protocol	TCP/IP Port Number	Status Reporting	Backchannel
Telnet	23	Yes	Yes
QMS	35	Partial	No
HP	9100	Yes	Yes

As stated above, ScriptServer's default TCP/IP protocol is Telnet, which uses port 23, and allows both backchannel and status reporting. When a ScriptServer TCP/IP queue starts, an OPCOM message is generated that reports the protocol being used. If the default port number (as determined by protocol) is overridden (see below), the overriding value for port number also appears in this message.

If your printer uses a protocol not directly supported, you can explicitly specify a different port number for ScriptServer to use. After identifying the appropriate port number, override the default value by defining a logical name as follows:

```
$ DEFINE/SYS SSV$address_TCPIP_PORT port-number
```

where "address" is the TCP/IP address as outlined above and "port-number" is the appropriate port number for the printer.

LPD Queues

lpd support is "turned on" by initializing the queue for a particular device using the following format for the /ON qualifier:

```
/ON="lpd_queue@host_printer"
```

Where "lpd_queue" is the name of the lpd print server queue and "host_printer" is either the printer's IP number address or the printer host name (see Section 4.2.5 *"Setting up a TCP/IP Connection*" on page 72). The lpd_queue specification is device specific, for example QMS uses “lp” and the Emulex NETJET Printer Server Card uses “PASSTHRU”. Table 4-4: on page 78 details the supported cards and the correct lpd-queue declaration used by each.

Table 4-4:lpd_queue Card Variations

Card/Printer	lpd Queue	Example
XCD XJetIV	HP_PRINTER	/ON="HP_PRINTER@192.100.10.2" queue
HP JetDirect	raw	/ON="raw@192.100.10.2" queue
QMS/Crown[a]	lp	/ON="lp@192.100.10.2" queue
Dataproducts	lpd[b]	/ON="lpd_queue@192.100.10.2" queue
Emulex NetJet	PASSTHRU	/ON="PASSTHRU@192.100.10.2" queue

[a] For lpd protocol to work properly on QMS/Crown printers, SPOOLING must be turned ON in the TCP/IP - DECnet submenu. This will prevent jobs from being lost on the printer.

[b] lpd is a “lpd gateway” on the Dataproducts VPT printers that must be configured to point to a lpd server queue, in this example, “lpd_queue”.

The following are two examples, the first using QMS and the second the Emulex Card:

```
$INIT/QUEUE/PROC=SSV$SMB/LIBR=SSV$DEVCTL/DEFAULT=FEED -
_$/ON="lp@192.100.10.3" JUGHEAD

$INIT/QUEUE/PROC=SSV$SMB/LIBR=SSV$DEVCTLE/DEFAULT=FEED -
_$/ON="PASSTHRU@192.100.10.3" JUGHEAD
```

where 192.100.10.3 is the Printer IP Address and JUGHEAD is the queue name. Status messages should indicate the lpd mode and queue name at startup. Backchannel will automatically be turned off.

When using the IP address, any lpd device specific logicals should use the IP address as the device and not “lpd_queue@host_printer”. And, when using the gethostbyname capability of ScriptServer, “device” should use the printer host-name (see Section 4.2.5 *"Setting up a TCP/IP Connection*” on page 72 above) as the device, not "lpd_queue@host_printer".

Foreign Queues

A ScriptServer Foreign Queue is a ScriptServer queue that accesses devices to which ScriptServer cannot directly connect. Rather than pointing to a device, these queues point to a non-ScriptServer execution queue.

To create a ScriptServer queue named PSQUEUE that feeds a foreign queue named LPS$PRINT, use the command:

```
$ INIT/QUEUE/PROC=SSV$SMB/LIBR=SSV$DEVCTL/DEFAULT=FEED -
_$ /ON=LPS$PRINT PSQUEUE
```

NOTE: *During queue startup, the ScriptServer symbiont first checks the output name to see if it is a queue name, creating a foreign queue if it is. Otherwise, ScriptServer assumes it is a device name and initializes as a standard queue.*

Foreign queues are specified by using the name of the queue to which output will be spooled as the argument to the “ON” qualifier. This may cause problems at sites that use the same name for queues as the device they service. For example, queue LPA0 on device LPA0:. This is not a valid name for a foreign queue because it is ambiguous; the software would assume that it is serving the device LPA0.

Furthermore, you cannot have multiple queues on the same device when one of the queues’ names is the physical device name; the ScriptServer symbiont would not know whether the name refers to the physical device or the other queue.

When submitting print jobs to foreign queues, ScriptServer does so with the equivalent to the command:

```
$ PRINT/USER=user-name
```

where “user-name” is the name of the user’s account from which the print job was submitted, and the job will be submitted with that user’s designated privileges.

Defining this logical to FALSE:

```
$ DEFINE/SYSTEM SSV$queue_PRINT_BY_USER FALSE
```

where “queue” is the name of the ScriptServer queue submitting the job, disables this feature. This means that the job will be submitted under the system account with all of its privileges, allowing compatibility with other manufacturers’ queues. Additionally, by default, ScriptServer passes print jobs to the execution queue with the /PASSALL qualifier. If this behavior is undesirable, with the following logical definition:

```
$ DEFINE/SYSTEM  SSV$queue_FOREIGN_PASSALL FALSE
```

the /PASSALL qualifier is not used.

Foreign Queues and Setup Modules

If you are printing a file via ScriptServer's Foreign Queue capability using a setup module, the setup module must exist in both the ScriptServer device control library (see Chapter 5) and the device control library used by the execution queue. However, the module in the execution queue's library should be a null file for correct execution.

Repository Queues

The repository queue sends the resulting output to a file in a directory instead of to an output device (e.g. a printer).

Creating and Starting a Repository Queue

When you create and start a repository queue, the argument to the /ON in the INIT/QUEUE qualifier must be set to a directory rather than a device:

```
$ INIT/QUEUE/PROC=SSV$SMB/LIBR=SSV$DEVCTL/DEFAULT=FEED -
_$ /ON=device:[directory] REPQUEUE
```

where "device:[directory]" are the appropriate device and directory specifications for the destination of the repository queue file and "REPQUEUE" is the name of your repository queue.

You can start a repository queue with the START/QUEUE command and change the destination directory simply by changing the device specification:

```
$ START/QUEUE/ON=device:[directory] REPQUEUE
```

NOTE: *There is a limit imposed by OpenVMS of 29 characters when using the /ON= qualifier. If the directory specification exceeds that limit, you will want to define a logical of shorter length equating to the longer directory specification.*

UIC and Protection Codes

By default, files printed on a repository queue will be translated to PostScript as necessary, creating the new output file in the specified directory with the original file owner UIC and Protection code.

You can force all of these output files to have a single UIC and / or new Protection code through the use of two logicals:

```
SSV$REPOSITORY_UIC
SSV$REPOSITORY_PROTECTION
```

For example, suppose you want all output files from the repository queue to be owned by the system, AND give no other users access to them (assume that the system account UIC is [1,12]). You would use the following logical definitions:

```
$ DEFINE/SYSTEM SSV$REPOSITORY_UIC “[1,12]”
$ DEFINE/SYSTEM SSV$REPOSITORY_PROTECTION “(S:RWED,O:RWED)”
```

NOTE: *For a more detailed explanations of these two logicals, see Chapter 18.*

Generic Queues

Generic ScriptServer queues conform to standard OpenVMS generic queues. A generic queue feeds target queues based on availability and capability. Initialize a generic ScriptServer queue with the following command:

```
$ INIT/QUEUE/GENERIC=[(queue[,...])] generic_queue_name
```

where “queue” is the name of target ScriptServer queue(s) and “generic_queue_name” is the name of the generic queue being initialized. When initializing a generic queue made up of more than one ScriptServer queue, the component queue list should be comma delineated and enclosed in parentheses. Because each component queue is also a regular ScriptServer print queue, it can be used independently of the ScriptServer generic queue just as if it were not a part of the generic queue. ScriptServer provides the option of creating a generic queue at installation time as well as during queue creation via the SSV$CONFIG command procedure discussed below.

Adding ScriptServer Print Queues

To add a ScriptServer queue to the list of sub queues targeted by the generic queue:

6. If the queue has not yet been created, create the ScriptServer queue either from the command line (see above) or the SSV$CONFIG command procedure (see below). If you are adding an existing print queue go on to step three below.
7. If using SSV$CONFIG, you will be prompted for whether or not you wish to add this queue to the list of sub-queues (see below). To do so, merely respond appropriately to the prompts.

8. If you wish to add a ScriptServer queue from the command line, you will first need to stop the generic queue:

```
$ STOP/RESET queue-name
```

where “queue-name” is the name of the generic queue, and then re-start the queue with the following command:

```
$ INIT/QUEUE/GENERIC=[(queue[,...])] queue-name
```

where “queue” is the name of component ScriptServer queue(s) and “queue-name” is the name of the generic queue being initialized. Simply add the new ScriptServer print queue to the target queue list designated for this generic queue.

NOTE: *You must specify ALL target queues you wish to have utilized by the generic queue every time to change the generic queue definition.*

4.3.2 Using the SSV$CONFIG Command Procedure

The SSV$CONFIG command procedure is the recommended way to create ScriptServer print queues. It is provided to make it easier to create queues and to help avoid problems caused by typing errors. The SSV$CONFIG command procedure is designed for use either at the DCL command level, or as a menu-driven facility.

```
SSV$CONFIG Command Line Mode
```

From the command line, SSV$CONFIG.COM only allows you to create and start ScriptServer queues. Additional options are available from the menu which is described below.

Parallel, Serial, and LAT Queues

To invoke SSV$CONFIG from the DCL command level for parallel, serial, and LAT connected printers, type:

```
$ @SCRIPTSERVER:SSV$CONFIG CREATE queue-name device
```

where “queue-name” is the name of the queue to be created, and “device” is the name of the port to which the PostScript printer is connected. This form of the procedure:

- sets the output device as SPOOLED to the queue that is being created

- creates the queue with the qualifiers listed above in Section 4.3.1 *"Using the INIT/QUEUE Command*" on page 74 as well as the /DEFAULT=(FEED,FLAG) and /SCHEDULE= NOSIZE qualifiers and
- starts the queue

Foreign Queues

The syntax for creating a foreign queue from the command line is:

```
$ @SCRIPTSERVER:SSV$CONFIG CREATE queue-name destination-queue
```

where “queue-name” is the name of the queue to be created, and “destination queue” is the name of the queue to which the ScriptServer queue will spool the print data. This form of the procedure:

- creates the queue with the qualifiers listed above in Section 4.2.1 *"Setting Up a Parallel Connection*" on page 66 as well as the /DEFAULT=(FEED,FLAG) and /SCHEDULE= NOSIZE qualifiers, and
- starts the queue.

TCP/IP Queues

For those printers connected over TCP/IP networks, the format is:

```
$ @SCRIPTSERVER:SSV$CONFIG TCPIP queue-name-
_$ TCP/IP-address protocol
```

Where “queue-name” is the name of the queue to be created, “TCP/IP-address” (see Section 4.3.1 *"Using the INIT/QUEUE Command*" on page 74 for a detailed discussion of address options) is the address to which the device in question is assigned, and “protocol” is the TCP/IP protocol in use on the printer to which the queue is pointing. The “protocol” parameter is optional, but will default to Telnet if left out. Other options currently supported are HP and QMS. For more detailed discussion of protocols and ports see the discussion surrounding Table 4-4: on page 78 above.

NOTE: *It is important that logical names for protocol and/or port be properly defined before running SSV$CONFIG from the command line level, otherwise the values of the logical names will default to those described in Table 4-4: on page 78.*

This form of the procedure:

- creates the queue with the qualifiers listed above in Section 4.3.1 *"Using the INIT/ QUEUE Command*" on page 74 as well as the /DEFAULT=(FEED,FLAG) and /SCHEDULE= NOSIZE qualifiers, and
- starts the queue.

DECnet Queues

For queues connected over DECnet networks the proper method is as follows:

```
$@SSV$CONFIG DECNET queue_name DECnet_printer_node_name
```

where “queue_name” is the name of the queue to which you wish to connect, and “DECnet_printer_node_name” is the node name of the printer to which you wish to connect.

This form of the procedure:

- creates the queue with the qualifiers listed above in Section 4.2.1 *"Setting Up a Parallel Connection*" on page 66 as well as the /DEFAULT=(FEED,FLAG) and /SCHEDULE= NOSIZE qualifiers, and
- starts the queue.

SSV$CONFIG Menu-driven Mode

To invoke SSV$CONFIG in the menu-driven mode, type:

```
$ @SCRIPTSERVER:SSV$CONFIG
```

The following is a diagram of the basic SSV$CONFIG menu options.

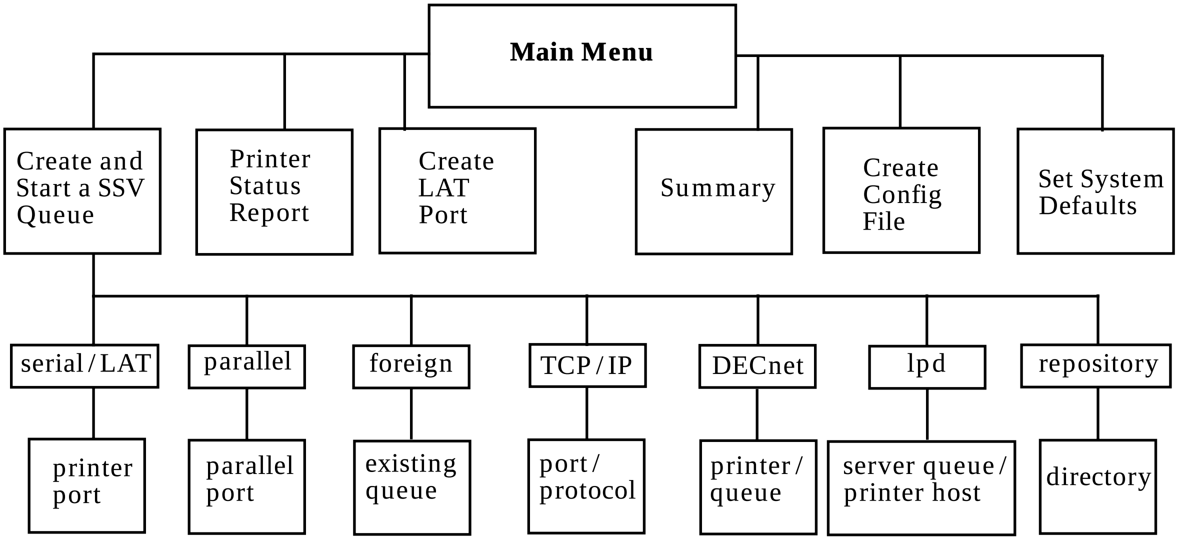


Figure 4-1: SSV$CONFIG.COM Menu Summary

The SSV$CONFIG.COM menu prompts you for all the information required for successful queue setup and configuration. Using the menu format:

- checks specified LAT ports and gives you the option to create LAT ports without exiting the procedure;
- lets you check each queue (via Printer Status Reports) for functionality without having to exit the procedure;
- displays which logical names should be placed in a system startup file for a successful reboot;
- gives you a summary of the queues created during a session, presenting you with a description of each queue detailing the options chosen during creation;
- allows you to add ScriptServer print queues to generic ScriptServer queues and will create generic queues if they do not already exist; and
- lets you create the ScriptServer configuration file, SSV$CONFIG.COM for management of your ScriptServer logicals (this file is discussed in detail below in Section 4.4 *"ScriptServer Configuration File*” on page 87).

- lets you set ScriptServer defaults via the SPD_MANAGER command procedure (described in detail in Section 12.1.1 *"Using SPD_MGR to Manage Defaults*" on page 205).

NOTE: *If an error does occur while using the SSV$CONFIG menu format, the procedure will abort with the error message:*

```
%SSV$CONFIG-F-FATEXITERR, Fatal exit error
```

TCP/IP Queues with SSV$CONFIG.COM

When creating your TCP/IP queue you will be asked for the TCP/IP address to which you may respond with the numerical TCP/IP address, the printer host name (see Section 4.3.1 *"Using the INIT/QUEUE Command*" on page 74 for details), or a logical name pointing to the printer's TCP/IP address (see Section 4.3.1 *"Using the INIT/QUEUE Command*" on page 74). The numerical address consists of four numbers, each less than or equal to 255 and separated by periods, for example, 192.100.10.5. On most printers with TCP/IP capability, this address can be established using the printer's control panel.

You will also be asked for the printer's TCP/IP port. If using one of the following protocol types: Telnet, QMS, or HP, the port number will automatically be defined according to Table 4-3: on page 77.

The default protocol type with ScriptServer is Telnet, which uses port number 23, and allows both backchannel and status reporting. If you are using either Hewlett-Packard or QMS TCP/IP-compatible printers, you must indicate the protocol to be used. If you are using some protocol type other than those discussed here you will need to determine the appropriate port number for your printer and enter it accordingly.

Generic Queues and SSV$CONFIG.COM

Once you have created a ScriptServer print queue with this command procedure you will be asked whether or not you wish to add the recently created queue to the list of subqueues for a ScriptServer generic queue:

```
Add queue-name to ScriptServer generic queue? [YES] ?
```

where “queue-name” is the name of the ScriptServer print queue just created. Responding YES to this prompt will result in the following prompt:

```
Generic queue to use queue-name as a target queue? [SSV$PRINT]
```

Pressing <RETURN> will add the print queue to the list of sub-queues for the generic queue, “SSV$PRINT”. If this generic queue does not exist, ScriptServer will create it. If your generic queue has a different name, you can type that name as the answer to this prompt and have the recently created print queue added to the list of target queues for that generic queue. If the generic queue name entered does not exist, ScriptServer will create it and add the queue to its list of target queues.

4.4 SCRIPTSERVER CONFIGURATION FILE

ScriptServer Printing System provides you the option of using a configuration file for management of your ScriptServer logicals. The file:

```
SCRIPTSERVER:SSV$CONFIG.SSV
```

is optional. If you choose not to use the configuration file, your ScriptServer logicals will be contained in the system table and behave as normal. If you use the configuration file, your ScriptServer logicals will be contained within a single text file that can be accessed and modified with ease.

4.4.1 Creating SSV$CONFIG.SSV

This file is included in the ScriptServer directory at installation and includes the two logicals:

```
SSV$DISPLAY_PRODUCTS
SSV$LIMITED_PRODUCTS
```

both of which are discussed above and in detail in Chapter 18 below. To edit these logical name definitions or these logicals, see the section below concerning modification of the file.

If this logical is not in your ScriptServer directory, either it has been corrupted, deleted or moved. There are three ways to create this file:

- through the SSV$CONFIG command procedure located in your ScriptServer directory (described in detail above in Section 4.3 *"Creating Queues*” on page 74);
- with the SSV$CONVERT_LOGICALS command procedure also located in your ScriptServer directory; or
- manually with a text editor.

NOTE: *None of these methods for creating the configuration file deassign existing logicals found in the system table. This will have to be done manually according to Chapter 3 ScriptServer Queues ScriptServer Printing System 39 OpenVMS procedure.*

Additionally, if a logical is defined in both the system table and the configuration file, the value associated with the logical definition in the system table will take precedence.

SSV$CONFIG.COM

When you run this menu based procedure with the command:

```
$ @SCRIPTSERVER:SSV$CONFIG
```

The topmost menu of the procedure offers you six options, the fifth of which:

```
5 Create ScriptServer Configuration File
```

creates the configuration file and converts all of your existing ScriptServer logicals into SSV$CONFIG.SSV file entries.

SCRIPTSERVER:SSV$CONVERT_LOGICALS.COM

If you have not created the SSV$CONFIG.SSV configuration file, when you run this procedure with the command:

```
$ @SCRIPTSERVER:SSV$CONVERT_LOGICALS
```

the file will be created and the appropriate existing ScriptServer logicals will be converted into SSV$CONFIG.SSV file entries.

If this file already exists, this command procedure will check the existing file for existing entries, appending new ScriptServer logicals found in the system table to the end of the configuration file.

Manual Creation of the Configuration File

The configuration file is simply a text file in the following format:

```
!
!
!
!
! S S V $ C O N F I G . S S V
!
!
! Logical Name         Value                Comments
!===============       =====                ========
!
SSV$DISPLAY_PRODUCTS   "*HP LaserJet 4"      !supports display
SSV$LIMITED_PRODUCTS   “*HP LaserJet IIIsi”
```

NOTE: *The “SSV$” prefix is optional when used within the configuration file.*

When adding a logical to the configuration file, simply add it to the list using the same format:

- logical name followed by some form of white space (i.e. tab or spaces);
- the value to which you want to define the logical;
- comments are optional, but if you want to add comments to the entry, simply precede the comments by an exclamation point (!).

NOTE: *Anything following an exclamation point (!) in the configuration file will be ignored.*

Once created, simply save the file in the ScriptServer directory with the name:

```
SSV$CONFIG.SSV
```

The ScriptServer Printing System will check this file for configuration just as if it had created the file itself by one of the previous two methods.

4.4.2 Modifying SSV$CONFIG.SSV

Adding Logicals

Definition of logicals can occur in a number of ways:

- from the command line;
- manual editing of the configuration file;
- through the SSV$CONFIG command procedure; or
- through the SPD_MGR command procedure.

Command Line Definition

Whether or not you are using the configuration file, definition of new logicals from the command line will automatically place those logicals in the system table. If you are using the configuration file and you want to include those logicals as entries in that file, you can run the SSV$CONVERT_LOGICALS command procedure (see above) which will convert all previously unconverted ScriptServer logicals in your system table.

Editing SSV$CONFIG.SSV

If you are using the configuration file, another way to add new ScriptServer logicals to your system is to add them as entries to the configuration file manually. SSV$CONFIG.SSV is a flat text file that can be edited with any standard text editor. Simply open the file using your favorite editor and add the logical to the file.

NOTE: *See Section 4.4.1 "Creating SSV$CONFIG.SSV" on page 87 on manually creating the SSV$CONFIG configuration file for details on the file format.*

This is the recommended way for adding entries to the configuration file.

SSV$CONFIG.COM

This command procedure is used to create queues for the ScriptServer Printing System (see above). The procedure often must define certain logicals to ensure correct operation of a ScriptServer queue. If you are using the SSV$CONFIG.SSV configuration file, this command procedure will automatically add these logicals as entries to the file and will NOT define them in the system table. If you are not using the configuration file, logicals will be defined within the system table.

SPD_MGR

This command procedure is used to associate SCRIPT settings with VMS forms and ScriptServer queues using logical names. Like the SSV$CONFIG.COM procedure mentioned above, if the configuration file is in use, this procedure will automatically add the appropriate logicals as entries to the file and will not define them in the system table.

If you are not using the configuration file, logicals will be defined within the system table.

Removing Entries from the Configuration File

Removing entries from SSV$CONFIG.SSV must be done manually. This is done by editing the file with your favorite text editor and either deleting the logicals from the file or commenting them out of the file by preceding each entry with an exclamation point (!).

Limitations to SSV$CONFIG.SSV

There are four logicals that will not be recognized if contained only within the configuration file:

```
SSV$DOWNLOAD_TABLE table (font and modules download data)

SSV$queue_HPGL_WIDTH logical names

SSV$queue_HPGL_COLORS logical names

SCRIPTSERVER directory logical
```

4.5 SCRIPTSERVER QUEUE DESCRIPTIONS

If characteristics are in use on the system when a ScriptServer queue starts (and a backchannel is available), the symbiont obtains the product and printer names from the printer and declares them as part of the description of the queue. This description is displayed in angle brackets by the SHOW QUEUE command. For example:

```
$ SHOW QUEUE LASER

Printer queue LASER, busy, on AL::KRUSTY::, mounted form DEFAULT
<ScriptServer V5.0 Queue on QMS Crown Printer [connecting]>
/BASE_PRIORITY=4 /CHAR=(10,11,12,13,19,20,21,22,23,24,25,26,
27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,
47,48,49,50,51,52,53,54,55,56,57,58,59,80,81,82,83,84,85,86,
90,91,92,93,94,95,96,97,104,105,106,107,108,109,110,111,112,
113,114,115,116,117,118,119,120,121,122,123,124,125,126)
/DEFAULT=(FEED,FORM=DEFAULT) /LIBRARY=SSV$DEVCTL Lowercase
/OWNER=[1,4] /PROCESSOR=SSV$SMB /PROTECTION=(S:M,0:D,G:R,W:S)
/RETAIN=ERROR
```

If this behavior is undesirable at your site because you use queue descriptions for some other purpose, you may disable the description by defining a logical name as follows:

```
$ DEFINE/SYSTEM SSV$INHIBIT_DESCRIPTION TRUE
```

4.6 STARTING QUEUES

ScriptServer Queues are started automatically upon successful creation by the SSV$CONFIG command procedure. You can, however, start a ScriptServer queue with the standard OpenVMS START command:

```
$ START/QUEUE queue-name
```

where “queue-name” is the name of the ScriptServer queue you wish to start.

To promote flexibility, a ScriptServer queue will only “hold on” to a printer port for as long as it takes to process the job; the ScriptServer print symbiont will release the printer port whenever it is not actively printing a job. In a LAT environment, this allows print spoolers on other nodes to access the printer. In non-LAT environments, releasing the printer port allows the creation of multiple queues (usually with different default settings or form names), all of which are communicating with the same printer.

The /ON Qualifier

You can use the /ON qualifier to change the device to which a queue is pointing when starting queues in much the same way as it can be used when initializing queues. The following example starts the queue PSQUEUE on device LTA100:

```
$ START/QUEUE/ON=LTA100: PSQUEUE
```

For detailed information about the /ON qualifier see Section 4.3.1 *"Using the INIT/QUEUE Command*” on page 74.

4.6.1 ScriptServer Queue Status Dialogue

With Backchannel Enabled

When a ScriptServer queue is started, you will see some startup dialogue on the second line of the queue description. This dialogue describes what is taking place between the symbiont and the output device. The “[connecting]” message highlighted in bold below is an example of one possible status message and a demonstration of where it will occur:

```
Printer queue DORY, busy, on AL::KRUSTY::, mounted form DEFAULT
<ScriptServer V5.1 Queue on QMS Crown Printer [connecting]>
/BASE_PRIORITY=4 /CHAR=(10,11,12,13,19,20,21,22,23,24,25,26,
27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,
47,48,49,50,51,52,53,54,55,56,57,58,59,80,81,82,83,84,85,86,
90,91,92,93,94,95,96,97,104,105,106,107,108,109,110,111,112,
113,114,115,116,117,118,119,120,121,122,123,124,125,126)
/DEFAULT=(FEED,FORM=DEFAULT) /LIBRARY=SSV$DEVCTL Lowercase
/OWNER=[1,4] /PROCESSOR=SSV$SMB /PROTECTION=(S:M,0:D,G:R,W:S)
/RETAIN=ERROR
```

Status messages that make up the startup dialogue are:

- [connecting]
- [downloading fonts]
- [preloading fonts]
- [awaiting completion]
- [stopped]

During the time the queue is [connecting] the ScriptServer print symbiont is conducting a dialogue with the printer to determine resident fonts, available paper sizes, color / monochrome capabilities, envelope feed capability, PostScript Level 2 support, product and printer names. After ScriptServer has completed sending a print job and is awaiting a return from the printer signalling completion, the status line in the queue description will read [awaiting completion].

When the symbiont shuts down by normal channels (i.e. some form of the STOP command), the status line will show [stopped].

NOTE: *If the queue doesn't complete the startup dialogue relatively quickly, it is generally because the printer is unavailable (frequently caused by another node being connected to the printer’s LAT port), or a communications problem has occurred.*

At the conclusion of the startup process (but only if queue characteristics were selected at installation time), the symbiont sets the appropriate queue characteristics on itself and puts the queue into an idle state.

Page Counter

The ScriptServer queue description will also display the number of the page being printed based on the frequency with which the symbiont performs checkpointing (described in Chapter 10). By default and unless declared otherwise with the logical described below, the symbiont checks the print job every two pages. Thus the queue description will display the number of every other page being printed. By changing the value of the logical:

```
SSV$CHECKPOINT_FREQUENCY
```

described inChapter 12 as well, you can change the accuracy of the number displayed in the ScriptServer queue description.

You can also turn off the display of the page number being printed in the queue description altogether with the following logical definition:

```
$ DEFINE/SYSTEM SSV$INHIBIT_PAGE_COUNTER TRUE
```

Without Backchannel Enabled

Without backchannel enabled (i.e. for parallel connected printers) no startup dialog will take place. The symbiont will simply open the connection specified, and (if characteristics are in use on your system) use a default set of characteristics.

NOTE: *These characteristics can be changed completely using the command:*

```
SET QUEUE/CHARACTERISTIC
```

Or modified with the logical:

```
SSV$queue_CHARACTERISTICS “#,#,#”
```

where “#,#,#” are queue characteristic numbers that you wish to add to your ScriptServer print queue. This is the recommended way for adding or modifying queue characteristics.

Additionally, ScriptServer will not provide a printer description in the queue status line, however, you may manually set this with the appropriate variation of the OpenVMS SET QUEUE command (see below). Finally, ScriptServer will not be able to return messages from the printer (i.e. printer error messages or page accounting information).

4.7 CHANGING AND DELETING QUEUES

ScriptServer has been designed with transparency in mind for both installation managers and end users. For this reason, most of the queue management tasks are identical to those for standard OpenVMS queues.

4.7.1 Modifying Queues with the SET QUEUE Command

After a queue has been created, it can be modified using the standard SET QUEUE command. The following sections describe how such queue modifications affect the ScriptServer Printing System where the behavior of the ScriptServer print symbiont is different from that of standard print symbionts.

Using SET QUEUE/CHARACTERISTICS

Since the ScriptServer print symbiont dynamically assigns queue characteristics (unless BACKCHANNEL is disabled) based on printer capabilities, do not use this qualifier to modify ScriptServer print queues that use backchannel. It may, however, be used with the following syntax to modify queues without backchannels, foreign queues, or generic queues that feed ScriptServer print queues:

```
$ SET QUEUE/CHARACTERISTICS=(#,#,#,...) queue-name
```

NOTE: *If a queue without a backchannel is pointing to a PCL printer, you will need to assign characteristic #4 to this list in order to indicate that the printer is a PCL printer.*

See also the discussion of the SSV$queue_CHARACTERISTICS logical name above or Chapter 18 of this manual.

Using SET QUEUE/DEFAULT=[NO]FEED

ScriptServer interprets the absence of the FEED characteristic associated with jobs as “allow text from the print file to be printed below the bottom margin.” Although ScriptServer will always start a new page when the bottom edge of the paper is reached, this may cause lines of text to fall outside of the imageable area on laser printers. For this reason, GrayMatter recommends that you assign the FEED characteristic to all ScriptServer queues. The SSV$CONFIG procedure (see Section 4.3.2 *"Using the SSV$CONFIG Command Procedure*" on page 82) creates queues in this manner. If you need to change a queue to FEED, use the DCL command:

```
$ SET QUEUE/DEFAULT=FEED queue-name
```

If you need to switch to NOFEED, use the following:

```
$ SET QUEUE/DEFAULT=NOFEED queue-name
```

Using SET QUEUE/NORECORD_BLOCKING

The ScriptServer print symbiont uses record blocking to optimize I/O performance on output to the printer. Although this is the recommended mode of operation, it may cause an increased time delay between when a PostScript error occurs on the printer and when the error is detected by the print symbiont. This may result in a significant discrepancy between the record number as reported in the Print Log and when the error actually occurred.

Defining Queue Setup Modules

The OpenVMS print facilities provide a way to reset the printer after each job using queue reset modules. These are specified using the SET QUEUE /SEPARATE=RESET command. However, there is no way to specify a setup module to be sent before each print job on a queue-specific basis. For this reason, the ScriptServer software supports a logical name that indicates which setup module(s) should be sent to the printer before every job printed on a particular queue. By defining a logical name in the form:

```
$ DEFINE/SYS SSV$queue_JOB_SETUP_MODULES "module-list"
```

you can specify that a set of modules indicated by the “module-list” are to be sent to the printer before each print job on queue “queue”.

This is discussed in more detail in Chapter 5.

4.7.2 Deleting Queues

There are no particular requirements for deleting ScriptServer queues. After stopping the queue to be deleted, use the standard DCL command DELETE/QUEUE. For example:

```
$ STOP/QUEUE/NEXT queue-name

$ DELETE/QUEUE queue-name
```

Chapter 5

Setup Modules

5.1 OVERVIEW

Setup Modules

Setup modules are generally used to insert device-dependent escape sequences that perform special functions such as setting horizontal pitch, margins, and type style for a particular print job. In the context of PostScript, however, these escape sequences are meaningless and usually cause the PostScript software error:

```
Undefined
```

For this reason, you may use setup modules when in PostScript mode as long as they are written in PostScript.

This chapter describes the different types of setup modules supported by ScriptServer, along with some guidelines to follow when writing PostScript setup modules.

The Device Control Library

Generally speaking, a device control library is a text library that holds printer setup modules.

A device control library (SSV$DEVCTL) is distributed with the ScriptServer software which contains several modules including the symbiont's translator prologues, SSV$PROLOG and SSV$HPGL_PROLOG, which must not be deleted from the library. If one of these modules is deleted from the library, ScriptServer will generate an error message when it tries to use the module reporting the problem to the operator(s) through OPCOM.

The remainder of this chapter describes the use of setup modules only as their use differs from that of conventional device control libraries. For a complete discussion of setup modules, refer to the OpenVMS system management documentation.

5.2 USING SETUP MODULES

Before a setup module can be used by ScriptServer in any of the forms described below in Section 5.3 "*Types of Setup Modules*" on page 98, it must first be placed in the ScriptServer Device Control Library, SSV$DEVCTL. This can be done with the following command:

```
$ LIBRARY/INSERT SYS$LIBRARY:SSV$DEVCTL.TLB module
```

where “module” is the name of the setup module you wish to insert.

5.3 TYPES OF SETUP MODULES

Table 5-1: on page 98 summarizes the types of device control setup modules supported by the ScriptServer Printing System, and the general role that each module plays.

Table 5-1:Setup Module Summary

Module	Description
Automatic Setup	sent at start of job and executed by the Post-Script interpreter whenever a new page is started for ASCII jobs (PostScript only)
Job Setup	sent to printer at start of job
Form Setup	sent at start of jobs printed on a specific form
Queue Setup	sent once at queue startup
Queue Setup	sent once at queue startup
Queue Reset	sent on completion of jobs printed on a specific queue
Page Setup	sent at the start of every page

5.3.1 Automatic Setup Modules

The ScriptServer software allows you to define a PostScript procedure that is called automatically by the PostScript interpreter whenever a new page is started. The advantage of automatic setup modules over page setup modules (see Section 5.3.6 *"Page Setup Modules*" on page 103 below) is that they are defined within the printer, so no form feed characters are required to trigger the downloading of a setup module from the device control library.

Another advantage is that they are transmitted to the printer only once at the beginning of the print job rather than once for each page, thus reducing transmission time.

The PostScript procedure that is called at the beginning of each page must be named *ssv$PageSetup*, which must be defined in the PostScript dictionary *ssv$dict*, which is created by the symbiont.

The recommended method for using automatic setup modules is as a form setup module (see Section 5.3.3 *"Form Setup Modules*" on page 101 below), which defines the procedure *ssv$PageSetup* in *ssv$dict*. A form may then be created that uses the setup module as shown below.

NOTE: *Because this type of setup module uses the PostScript interpreter on a PostScript printer, it functions correctly only in PostScript mode.*

The ScriptServer Printing System is distributed with several automatic setup modules, including:

- GREEN-BAR
- GREEN-BAR-SMALL
- AUTO_CONFIDENTIAL

See Section 5.8 *"Included Setup Modules*" on page 108 below for a listing of all included setup modules and instructions for their use.

Multiple Page Setup Modules

Multiple page setup modules must be implemented as automatic setup modules. When using multiple setup modules per page, each setup module must be defined within a single module as in the following PostScript language code:

```
%!PS-Adobe

%%Title:Multisetup.ps

ssv$dict begin
```

```
/pagesetup1 { gsave /Times-Bold findfont 24 scalefont setfont 200 400
              moveto (multi-pagesetup 1) show grestore} def

/pagesetup2 { gsave /Times-Bold findfont 24 scalefont setfont 200 200
              moveto (multi-pagesetup 2) show grestore} def

/ssv$PageSetup [/pagesetup1 /pagesetup2] def end
```

The rules for using multiple page setup modules are fairly simple:

1. Each module must be defined in the *ssv$dict* dictionary. In the above example the setup modules are defined as the keys pagesetup1 and pagesetup2;
2. For page setup modules to be used in conjunction with ASCII data, the key *ssv$PageSetup* must be defined as either a PostScript procedure, or as an array of PostScript procedures.

In the above example, the key has been defined as an array consisting of the two previously defined keys: *pagesetup1* and *pagesetup2*.

Backside Setup Modules

You can print a static image on the back side of every page of a print job with the page setup module designated as *ssv$BackSetup*. This image cannot be used with any additional data, ASCII or otherwise and because it forces the PostScript operator showpage, will always print on the back side of a duplexed page or every other page of a single-sided print job beginning with the second page.

To use this capability you will need to treat the *ssv$BackSetup* module as if it were one of several page setup modules as described above. This module follows the same rules:

1. it must be defined within the *ssv$dict* dictionary;
2. it must be defined as the key *ssv$BackSetup*;
3. if you are only using one BackSetup module, it can be defined as a PostScript procedure;
4. if you are using multiple BackSetup modules, it must be defined as an array of PostScript procedures, the members of which must also be defined within the *ssv$dict* dictionary as PostScript procedures.

The following example uses a single back setup module in conjunction with the multiple page setup modules described above:

```
%!PS-Adobe
%%Title:Multisetup.ps
ssv$dict begin
/pagesetup1 { gsave /Times-Bold findfont 24 scalefont setfont 200
              400 moveto (multi-pagesetup 1) show grestore} def

/pagesetup2 { gsave /Times-Bold findfont 24 scalefont setfont 200
              200 moveto (multi-pagesetup 2) show grestore} def

/ssv$PageSetup [/pagesetup1 /pagesetup2] def

/ssv$BackSetup {gsave 5 setlinewidth .5 72 72 moveto 540 720 lineto
                stroke grestore} def

end
```

NOTE: *No "data" can be printed on these pages which will print on the back side of every page of a print job or every other page if not used with a duplex printer.*

5.3.2 Job Setup Modules

Job setup modules are assigned on a job-specific basis at the time a print request is made with either the PRINT or SCRIPT command. The ScriptServer Printing System is distributed with a job setup module named EHANDLER, which provides PostScript error handling.

See section 4.8 below for a description of the EHANDLER and instructions for its use.

5.3.3 Form Setup Modules

You can define form setup modules that are associated with a given form name on the system. These modules are automatically sent to the printer prior to each file printed on a given form. Because setup module names are not verified at the time a print request is made with PRINT or SCRIPT, using form setup modules can prevent the loss of print jobs due to spelling or typing errors in setup module names.

Form setup modules are also a very convenient way to have the PostScript printer automatically generate predefined images such as company letterhead without having to change the paper in the printer.

See Chapter 13 on ScriptServer SoftForms for more information on replacing pre-defined forms. This is best accomplished using the automatic setup module capability of ScriptServer for those jobs printed using ASCII-to-PostScript translation. See Section 5.3.1 *"Automatic Setup Modules*" on page 99 above for more information on automatic setup modules.

To define a setup module associated with a form that prints on the DEFAULT paper stock, use the command:

```
$ DEFINE/FORM/SETUP=module-name/STOCK=DEFAULT name number
```

NOTE: *ScriptServer does not support VMS Forms metrics (i.e. margins, lines-per-page, etc.) in PCL mode.*

5.3.4 Queue Setup Modules

The OpenVMS operating system provides no direct mechanism for defining queue setup modules, having left this functionality to the reset module(s) as described below. However, because of the PostScript server loop and printer-sharing between nodes in a network, it may be necessary to provide the ability to download setup modules on a queue-specific basis for PostScript printing.

To define a module or set of modules as queue setup modules, define a system-wide logical name as follows:

```
$ DEFINE/SYSTEM SSV$queue_MODULES module-list
```

If you are defining a list of setup modules, separate the module names with commas and enclose the list in quotation marks.

5.3.5 Queue Reset Modules

On non-PostScript printers, this device control library module is used to reset the printer to a given state at the conclusion of each print job, so that subsequent jobs don't “inherit” properties of previous jobs. On PostScript printers, this is done automatically by the server loop within the PostScript interpreter.

Although resetting the graphics state in the printer is handled automatically, a print job can perform input tray selection that is not automatically reset on some printers. For example, an IBM Personal PagePrinter II (model 4019) can be configured with up to three input trays of different sizes. If the default paper size is Letter, and a job is submitted on Legal, subsequent PostScript print jobs will print on Legal paper unless they explicitly select a particular tray.

The ASCII translator in ScriptServer always selects input trays by paper size, but no tray selection is performed for PostScript jobs, as that is left to the PostScript-generating application.

To associate a queue setup module with a queue, use the command:

```
$ SET QUEUE/SEPARATE=RESET=module-name queue-name
```

5.3.6 Page Setup Modules

Page setup modules are sent to the printer at the start of every page that is printed on a given form name. These setup modules allow you to create a template image that is placed on the page before any text from the input file. The downloading of page setup modules is triggered by form feed characters as they are encountered in the print file. If a new page is started by an attempt to exceed the bottom margin on the current page, the page setup module will not be downloaded to the printer.

For this reason, you may wish to use the automatic setup module capability (see above) of ScriptServer, which doesn't require form feed characters to be triggered.

5.4 WRITING POSTSCRIPT SETUP MODULES

This section describes how to create setup modules for use in conjunction with the ScriptServer Printing System in PostScript mode. These setup modules must be written in PostScript, so a thorough knowledge of PostScript programming is necessary.

NOTE: *PostScript is a fairly complex programming language, it is beyond the scope of this document to provide tutorial instruction on its use.*

5.4.1 Behavior Rules

Any PostScript code to be imported as a setup module should conform to rules known as “well-behaved.” An example of good behavior would be to enclose the code within a save/ restore construct, which allows all printer virtual memory (VM) to be recovered and the graphics state restored. Since the setup module will be incorporated within the PostScript code being generated by the print symbiont, it must not disrupt the program it is operating within. These behavior rules are outlined in the sections below and are essentially the same as those for creating Encapsulated PostScript files.

A detailed description of Encapsulated PostScript files is available in the document *Encapsulated PostScript File Format* from Adobe Systems. It is also documented in an appendix in the *PostScript Language Reference Manual,* Second Edition, published by Addison-Wesley.

Preserving Graphic Context

When writing PostScript setup modules, it is important to avoid irreversible graphic transformations and other wholesale modifications to the PostScript environment, such as those caused by the following PostScript operators:

- *grestoreall*
- *initmatrix*
- *initgraphics*
- *initclip*
- *erasepage*
- *nulldevice*
- *exitserver*

PostScript Stacks and Dictionaries

All PostScript stacks should be returned to the state that they were in before the setup module is invoked. This includes the dictionary stack, which you should never assume to be or have been in any given state. It is recommended that the setup module create any dictionaries that may be needed for local variable definitions. This dictionary must be removed from the stack (with the end operator) before the setup module is complete to avoid the occurrence of the PostScript error:

```
invalidrestore
```

ScriptServer Graphics State

The PostScript code generated by the ASCII translator operates within the default PostScript graphics state, where:

- the origin (location [0,0]) is in the lower left corner
- the x axis increases to the right of the origin
- the y axis increases going up from the origin
- the scale is 1 unit = 1 point
- the default line width is 1 point

Any modification to the graphic state must be placed within a *save/restore* or *gsave/grestore* construct.

The *showpage* and *copypage* Operators

The PostScript operators *showpage* and *copypage* are used to cause the current page to be ejected from the printer.

As this is generally undesirable in the context of a setup module, any occurrence of these operators should be removed from the setup module. Or, if the setup module is operating within a *save/restore* construct, these operators can be temporarily disabled within the module using code similar to the following segment:

```
/sm$save save def
/showpage {} def
.
.
.
sm$save restore
```

5.5 WORKING WITH SETUP MODULES

As has been noted, ScriptServer requires that setup modules be written in PostScript. Occasionally, you will need to edit or modify an existing setup module. This is easily done with the following procedure:

1. Stop all of your ScriptServer queues using the following command:

   ```
   $ STOP/RESET queue-name
   ```

2. Extract the modules you wish to work with from the ScriptServer Device Control Library with the following command:

   ```
   $ LIBRARY/EXTRACT=module/OUT=module.PS SYS$LIBRARY:SSV$DEVCTL.TLB
   ```

   Where “module” is the name of the module you wish to extract and “module.PS” is what you want to call the module once it has been extracted. It is generally a good idea to name the file the same name as the module with the .PS file extension.

3. Using your favorite text editor, edit the module with whatever changes you wish to make.

4. Once changes have been made, return the module to the Device Control Library with the following command:

   ```
   $ LIBRARY/REPLACE SYS$LIBRARY:SSV$DEVCTL.TLB module
   ```

Where “module” is the name of the file you have edited without the .PS file extension.

5. Restart your ScriptServer queues:

```
$ START/QUEUE queue-name
```

Your modified setup module should now be available to all ScriptServer queues on your network. As long as you left the module name unchanged, any forms that you may have created using this setup module should now use the modified version without any additional changes.

5.6 PERMANENT SETUP MODULES

In some printing environments, it may be necessary to ensure that certain PostScript setup modules (such as the Apple LaserPrep prologue) are always resident in the printer. You can be certain that a given prologue is always resident by defining it as permanently-resident with the following procedure:

1. Place the setup module (which must be written in PostScript) in the ScriptServer device control library SSV$DEVCTL.TLB with the LIBRARY command. For example:

```
$ LIBR/INSERT SYS$LIBRARY:SSV$DEVCTL.TLB MYMODULE
```

2. Equate the logical name SSV$queue_MODULES (where “queue” is the name of the Script-Server queue) within SCRIPTSERVER:DOWNLOAD.COM to the name or list of names of the permanently-resident module(s), for example:

```
$ DEFINE/TABLE=SSV$DOWNLOAD_TABLE SSV$LASER_MODULES -
_$ "MYMODULE,EHANDLER"
```

   When set up this way, ScriptServer will download the setup module(s) to the printer each time the queue is started. The file, SCRIPTSERVER:DOWNLOAD.COM, contains comments detailing this process.

3. You may define a module or group of modules that are to be permanently downloaded to all ScriptServer queues by substituting the logical name SSV$BLANKET_MODULES for SSV$queue_MODULES in the above command.

For example, to specify that the modules MSWORD_PROLOG and LETTERHEAD are to be permanently downloaded to all ScriptServer-controlled printers, use the command:

```
$ DEFINE/TABLE=SSV$DOWNLOAD_TABLE SSV$BLANKET_MODULES -
_$ "MSWORD_PROLOG,LETTERHEAD"
```

These modules will be permanently downloaded to the printer each time the queue or queues are started with the OpenVMS START/QUEUE command.

Permanently Downloading SSV$PROLOG

Another practical example involves the use of the SSV$PROLOG module mentioned above in Section 5.1 *"Overview*" on page 97. By adding the logical:

```
DEFINE/TABLE=SSV$DOWNLOAD_TABLE SSV$queue_MODULES "SSV$PROLOG"
```

to the last part of SCRIPTSERVER:DOWNLOAD.COM, you can permanently download this module to a specific queue. Doing this will make your ASCII files print that much faster because ScriptServer will not have to download the prologue every time a job is sent for printing. To permanently download this module system wide, simply substitute BLANKET for “queue” in the above example:

```
DEFINE/TABLE=SSV$DOWNLOAD_TABLE SSV$BLANKET_MODULES "SSV$PROLOG"
```

Server Loop Exit Control

When ScriptServer downloads permanent setup modules (and fonts), the symbiont assumes responsibility for exiting the server loop using the PostScript operator *exitserver*. This operator requires a password which is usually an integer. By default, the password is 0 (or in the case of the LN03R ScriptPrinter, it is the string “LN03R”). Unless a logical name of the form SSV$device_EXITSERVER is equated to another value (as in the following example):

```
DEFINE/SYSTEM SSV$device_EXITSERVER “555”
```

where “device” is the device to which the printer is attached (see Section 5.3.1 *"Automatic Setup Modules*” on page 99 for the different possible device types.) ScriptServer will assume that the password is the default value (“0” or “LN03R”).

5.7 CONTROLLING DOWNLOAD MESSAGES

When downloading permanent setup modules or fonts (see Chapter Six below), the ScriptServer print symbiont sends messages to OPCOM with the name of the module or font being downloaded. By default, it does this on both job-specific and permanent downloads. This behavior can be modified on a system-wide or queue-specific basis by defining the appropriate system-wide logical name.

To define download message handling for all queues system-wide, use the logical name:

```
SSV$DOWNLOAD_MESSAGES
```

For queue-specific message handling, use the name

```
SSV$queue_DOWNLOAD_MESSAGES
```

Define the logical name to the value:

- ALL;
- NONE; or
- PERMANENT.

For example, if you want messages for all queues system-wide to indicate only setup modules and fonts that are permanently downloaded at queue startup, use the command:

```
$ DEFINE/SYSTEM SSV$DOWNLOAD_MESSAGES PERMANENT
```

To disable all download messages on a single queue named PSQUEUE, use the command:

```
$ DEFINE/SYSTEM SSV$PSQUEUE_DOWNLOAD_MESSAGES NONE
```

NOTE: *Similar messages are generated for downloaded Fonts (See Chapter 7 below). All download messages are controlled by the above logical names.*

5.8 INCLUDED SETUP MODULES

The following prologues are necessary for the translation of ASCII and HPGL files to PostScript but should never be modified by the user.

- SSV$PROLOG

In addition to the several setup modules included in the software for purely functional reasons, ScriptServer provides some additional setup modules that can be manipulated by the end user:

- EHANDLER
- IMAGESHIFT
- GREEN-BAR
- GREEN-BAR-SMALL
- CLASSIFIED
- SECRET
- TOP_SECRET
- PRELIMINARY
- DRAFT
- CONFIDENTIAL
- AUTO_CONFIDENTIAL

5.8.1 Translator Prologues

SSV$PROLOG

This setup module is one of the symbiont's translator prologues. It provides the information necessary for the symbiont to perform translations from ASCII to PostScript and may be permanently downloaded to your printer to increase the speed of the printing cycle, see Section 5.6 "*Permanent Setup Modules*" on page 106. This module is required in order to perform this translation.

SSV$HPGL_PROLOG

This setup module is one of the symbiont's translator prologues. It provides the information necessary for the symbiont to perform translations from HPGL to PostScript and may be permanently downloaded to your printer to increase the speed of the printing cycle, see Section 5.6 "*Permanent Setup Modules*" on page 106. This module is required in order to perform this translation.

5.8.2 EHANDLER

The EHANDLER module is a modified version of the standard PostScript error handler from Adobe Systems. The standard error handler is designed to be downloaded permanently (until power-off) into the printer, and generates an error page containing various information about the error when one occurs.

The modified error handler distributed with the ScriptServer software is only resident in the printer during the processing of the job that it is submitted with. To submit a job with the error handler use the command:

```
$ PRINT/SETUP=EHANDLER file-spec
```

ScriptServer is also distributed with the permanent version of the error handler in the file SCRIPTSERVER:PERM_EHANDLER.PS. To use this version which remains in the printer until it is powered off, simply submit the file to the queue servicing the printer:

```
$ PRINT/QUEUE=queue-name SCRIPTSERVER:PERM_EHANDLER.PS
```

5.8.3 IMAGESHIFT

ScriptServer provides a setup module which facilitates printing duplex on three-hole punched paper. To use the “imageshift” feature with text files, simply define a VMS form with IMAGESHIFT as the setup module:

```
$ DEFINE/FORM/SETUP=IMAGESHIFT/STOCK=DEFAULT formname 99
```

where “formname” is the name of the form you have defined and “99” is the unique formnumber to which you wish to associate the form. To print with this module, use the form as follows:

```
$ PRINT/QUEUE=queue/FORM=formname/PARAM=DUPLEX file.txt
```

where “queue” is the name of the queue to which you want to submit the job, “formname” is the name of the form previously defined, and “file.txt” is the text file you want to shift.

The default shift distance is defined as 36 points (1/2 inch) which works well with standard portrait orientation documents using ScriptServer's default margins. However, with landscape orientation, lines of text will be lost without adjusting the top and bottom margins. This can be easily remedied by creating another setup module with a shift distance of 18 points (1/4 inch) as follows:

1. Extract the IMAGESHIFT module from the ScriptServer device control library:

```
$ LIB/EXTR=IMAGESHIFT/OUT=IMAGESHIFT_LSCP.PS -
_$ SYS$LIBRARY:SSV$DEVCTL.TLB
```

2. Edit the file IMAGESHIFT_LSCP.PS, and change the “imageshift” to 18:

```
/ssv$shiftdict 5 dict def ssv$shiftdict begin /imageshift 18 def
```

3. Put the module into the device control library (you will need to stop ALL Script-Server queues before doing this):

```
$ LIB/INSERT SYS$LIBRARY:SSV$DEVCTL.TLB IMAGESHIFT_LSCP.PS
```

4. Define another form using this new setup module with its own form number and name and use when printing landscape jobs:

```
$ PRINT/QUEUE=queue/FORM=formname -
_$ /PARAM=(DUPLEX,"PAGE_O=LANDSCAPE") file.txt
```

   where “queue” is the name of the queue to which you wish to submit your file, “formname” is the name of your newly defined form, and “file.txt” is the text file you want affected.

5.8.4 Page Setup Modules

The following setup modules are predefined page setup modules that can be implemented as either regular page setup modules or as automatic setup modules. The following section concerning the setup module, AUTO_CONFIDENTIAL has instructions for using these page setup modules as automatic setup modules.

GREENBAR

GREEN_BAR emulates the standard green-bar paper frequently used on line printers. This setup module is designed to be used with the setting LINE-PRINTER, which is pre-defined in the symbiont settings library.

GREEN_BAR_SMALL

A smaller version of the traditional GREENBAR module described above.

CLASSIFIED

If implemented as a standard page setup module, prints the word “CLASSIFIED” across the first page of a print job. If implemented as an automatic setup module, prints the word across all pages of a print job.

SECRET

If implemented as a standard page setup module, prints the word “SECRET” across the first page of a print job. If implemented as an automatic setup module, prints the word across all pages of a print job.

TOP_SECRET

If implemented as a standard page setup module, prints the word “TOP SECRET” across the first page of a print job. If implemented as an automatic setup module, prints the word across all pages of a print job.

PRELIMINARY

If implemented as a standard page setup module, prints the word “PRELIMINARY” across the first page of a print job. If implemented as an automatic setup module, prints the word across all pages of a print job.

DRAFT

If implemented as a standard page setup module, prints the word “DRAFT” across the first page of a print job. If implemented as an automatic setup module, prints the word across all pages of a print job.

CONFIDENTIAL

If implemented as a standard page setup module, prints the word “CONFIDENTIAL” across the first page of a print job. If implemented as an automatic setup module, prints the word across all pages of a print job.

5.8.5 Automatic Setup Modules

AUTO_CONFIDENTIAL

This setup module is provided to show the use of automatic setup modules to stamp the word CONFIDENTIAL at the bottom and top of each page of a print job according to governmental specifications for caveats on sensitive documents. This setup module is extensively commented to explain how to write your own automatic setup modules for other purposes.

Using Page Setup Modules as Automatic Setup Modules

The following sample shows how to define a form that uses the GREEN-BAR automatic setup module:

```
$ DEFINE/FORM/STOCK=DEFAULT/SETUP=GREEN-BAR GREENBAR form-number
```

To have the LINE-PRINTER settings automatically used when printing on form GREEN-BAR, define a logical name as follows:

```
$ DEFINE/SYSTEM SSV$FORM_GREENBAR_SETTINGS LINE-PRINTER
```

To print a file using GREENBAR, use a PRINT command in the following format:

```
$ PRINT/FORM=GREENBAR file-spec
```

Chapter 6

HP-GL Translation and Emulations

The ScriptServer print symbiont supports the translation of the Hewlett-Packard Graphics Language (HP-GL) to PostScript. It also allows you to access printer-based emulations supported by various PostScript printers. This chapter describes how these non-PostScript printer languages are accessed through ScriptServer.

NOTE: *Printing of HPGL files through ScriptServer will only work in PostScript mode.*

6.1 HP-GL TRANSLATION

HP-GL translation is performed by the symbiont on the fly. There are two methods by which you can indicate that a given file requires HP-GL translation:

- give it a file type of .HP; or
- submit the file with the parameter DATA_TYPE, for example:

```
$ PRINT/PARAM=DATA_TYPE=HPGL file-spec
```

6.1.1 Polygon Mode Support

The HP-GL translator includes true polygon mode support. Polygon mode is a feature of HP-GL that allows users to define a polygon without actually plotting it. Subsequent commands may be used to plot and/or fill the polygon.

Users should be aware that many printer-based HP-GL emulations do not support Polygon Mode. They simply plot the outline of the polygon as it is being defined, ignoring any subsequent “Fill Polygon” commands. As a result, HP-GL files printed on these printers may not look the same as the same files translated by ScriptServer.

6.1.2 Pen Colors and Width

The HP-GL translator includes color support for up to eight plotter pens. Most monochrome printers will convert each color to some shade of gray. Table 6-1: on page 116 shows the default pen color assignments used by the translator.

Modifying Pen Color Assignments

It is possible to change pen assignments on a per queue basis. Colors are specified using the RGB (Red-Green-Blue) color model, with each color represented by a value between zero and one. For example, the color violet consists of a red value of .5, a green value of 0, and a blue value of 1.

To change pen assignments for a queue, define a system-wide logical name in the following form:

```
$ DEFINE/SYSTEM SSV$queue-name_HPGL_COLORS -
_$"red1,green1,blue1;red2,green2,blue2;..."
```

where “red1” is the red value for pen one, “green1” is the green value for pen one, and so forth. You may define up to eight pen colors. Pens that are not given a color default to black. Thus, to assign the color black to all pens for the queue HP1, define a logical name as follows:

```
$ DEFINE/SYS SSV$HP1_HPGL_COLORS "0,0,0;"
```

Table 6-1:HP-GL Pen Color Assignments

Pen Number	Pen Color
One	Black
Two	Red
Three	Green
Four	Yellow
Five	Blue
Six	Violet

Table 6-1:HP-GL Pen Color Assignments (Continued)

Pen Number	Pen Color
Seven	Aqua
Eight	Orange

This logical name will be evaluated at the time the queue is started. If it is changed, the new colors will not take effect until the queue is stopped and restarted.

Modifying Pen Width

You can control the pen width used by the HP-GL translator by defining the logical:

```
SSV$queue_HPGL_WIDTH
```

as a standard ScriptServer size expression. Size expressions may be in terms of points (pt), inches (in), or centimeters(cm). If the size expression is omitted, the units used are determined by whether or not international defaults are in effect, which is set up at installation time. If international defaults are in use (see Section 3.7 *"Paper Sizes Supported*" on page 62 above), the default unit of measure is centimeters, otherwise it will be inches.

In the HP-GL plotter language, the commands IN (initialize), DF (default), and SP (select pen) reset the pen thickness to the default value, and the PT command sets the pen thickness to the desired width. The pen width logical name can be defined such that these commands do not override the pen thickness being established. This is done by appending a “+” to the size expression.

For example, this logical name definition will set the pen thickness to .1 inches (because international defaults are NOT in use) on queue LASER:

```
$ DEFINE/SYS SSV$LASER_HPGL_WIDTH ".1"
```

The following definition sets the default pen thickness to .1 inches, and prevents it from being overridden by the SP and PT commands:

```
$ DEFINE/SYS SSV$LASER_HPGL_WIDTH ".1+"
```

To specify a pen width of the minimum device resolution for a queue named CLIPPER, define the logical name as follows:

```
$ DEFINE/SYS SSV$CLIPPER_HPGL_WIDTH "0.0"
```

6.1.3 HP-GL Command Support Level

The set of commands supported by the HP-GL translator are modeled after the Hewlett-Packard 7550-A plotter. Because of the inherent differences between PostScript printers and plotters, not all commands are fully supported.

Commands supported by the HP-GL to PostScript translator are summarized below. The commands in Table 6-2: on page 118 that are tagged with an asterisk are supported in a limited form. The level of support for these commands is summarized in Table 6-3: on page 120. Some commands (such as those generating output messages from the plotter) are not supported. These commands are summarized in Table 6-4: on page 121.

Table 6-2:Supported HP-GL Commands

HP-GL Command	Description
AA	Arc Absolute
AF	Advance Full Page
AH*	Advance Half Page
AR	Arc Relative
BL	Buffer Label
CA*	Designate Alternate Character Set
CI	Circle
CP	Character Plot
CS*	Designate Standard Character Set
CT*	Chord Tolerance
DF	Default
DI	Absolute Direction
DL	Define Download Character
DR	Relative Direction
DT	Define Label Terminator
EA	Edge Rectangle Absolute
EP	Edge Polygon
ER	Edge Rectangle Relative
ES	Extra Space

Table 6-2:Supported HP-GL Commands (Continued)

HP-GL Command	Description
EW	Edge Wedge
FP	Fill Polygon
FT*	Fill Type
IN	Initialize
IP	Input P1 and P2
IW	Input Window
LB	Label
LT	Line Type
PA	Plot Absolute
PB	Print Buffered Label
PD	Pen Down
PG	Advance Full Page
PM	Polygon Mode
PR	Plot Relative
PT	Pen Thickness
PU	Pen Up
RA	Fill Rectangle Absolute
RO	Rotate
RR	Fill Rectangle Relative
SA*	Select Alternate Character Set
SC	Scale
SI	Absolute Character Size
SL	Character Slant
SM	Symbol Mode
SP	Select Pen
SR	Relative Character Size
SS*	Select Standard Character Set
TL	Tick Length

Table 6-2:Supported HP-GL Commands (Continued)

HP-GL Command	Description
UC*	Plot User-Defined Character
WG	Fill Wedge
XT	X-tick
YT	Y-tick

Table 6-3:Limited Support HP-GL Commands

CMD	Description	Restriction
AH	Advance Half Page	Advances a full page, same as PG and AF commands.
CA	Designate Alternate Character Set	Uses Courier for a fixed-space character set, and Helvetica for variable-spaced character set; -1 uses character set downloaded with the DL command.
CS	Designate Standard Character Set	Same as description for the CA command above.
CT	Chord Tolerance	Chord tolerance may only be selected by degrees, not distance. If chord tolerance by distance is specified, chord tolerance amounts will be ignored, and a default of five degrees will be used.
FT	Fill Type	User-defined fill types are not supported. Adaptive line types are not supported for cross-hatch and parallel line fill patterns.
SA	Select Alternate Character Set	See description for CA command above.
SS	Select Standard Character Set	See description for CA command above.
UC	Plot User-Defined Character	On some HP plotters, the behavior of this command is determined by the setting of a “Standard/Enhanced” function key. Standard is assumed. The Character is plotted on a 6-by-16 grid. (1 space=6 units, 1 line=16 units)

Table 6-4:Unsupported HP-GL Commands

HP-GL Command	Descriptions
AP	Automatic Pen Operations
AS	Acceleration Select
BF	Buffer Plot
CM	Character Selection Mode
CC	Character Chord Angle
CV	Curved Line Generator
DC	Digitize Clear
DP	Digitize Point
DS	Designate Character Set Into Slot
DV	Direction Vertical
EC	Enable Cutter
FR	Advance Frame
FS	Force Select
GC	Group Count
GM	Graphics Memory
GP	Group Pen
IM	Input Mask
IV	Invoke Character Set
KY	Define Key
LO	Label Origin
NR	Not Ready
O-	Output Data to Host Commands
PS	Paper Size
RP	Replot
SG	Select Pen Group
UF	User-Defined Fill Type
VS	Velocity Select
WD	Write to Display

6.2 PRINTER-BASED EMULATIONS

ScriptServer allows access to printer-based emulations through the use of site-defined emulation names that cause setup modules from the device control library to be sent to the printer.

6.2.1 Defining Printer-Based Emulations

Each emulation requires a set of two setup modules:

- a module to invoke the emulation, and
- another to clear the emulation and return the printer to PostScript mode.

It is imperative that the printer be returned successfully to PostScript mode or subsequent print jobs will fail.

As stated above, emulation names are site-defined. The mechanism that makes emulation work is a standardized naming convention linked to the DATA_TYPE PRINT parameter. For example, su pose you want to define an emulation called LN03. To invoke the emulation, enter:

```
$ PRINT/PARAM=DATA_TYPE=LN03 file-spec
```

When the ScriptServer print symbiont sees a data type other than the predefined data types (ASCII, ANSI, POSTSCRIPT, HPGL, PCL, and AUTOMATIC), it assumes that a site-defined emulation is being invoked. In this case, the emulation name is LN03 so the symbiont will look for the following two setup modules in the device control library:

```
SSV$SET_LN03 to invoke the emulation;

SSV$CLEAR_LN03 to clear the emulation.
```

6.2.2 Emulation Delay Time

Some PostScript printers require a period of time during which the printer sets up the emulation being invoked. On these printers, data transmitted during this period will be lost, either causing incomplete documents or failing to print at all.

To handle this situation, ScriptServer provides an emulation delay time during which the symbiont stops sending data to the printer giving it time to reset. The default emulation delay is 10 seconds. If this delay period is too long or too short, it can be adjusted through the definition of the logical name:

```
SSV$EMULATION_DELAY time-spec
```

Where “time-spec” is a delta time value given in seconds. For example, if you want to change the delay time to 15 seconds, define the logical name as follows:

```
$ DEFINE/SYSTEM SSV$EMULATION_DELAY 15
```

6.2.3 Automatic Emulation Support

Some later-model PostScript printers (for example, the HP LaserJet 4) automatically determine the type of data coming from the host and switch to the proper emulation without need of a special command or escape sequence from the host. (On the HP LaserJet 4 this capability is referred to as auto personality.)

ScriptServer supports these printers through the use of the emulation name AUTOMATIC. This mode of printing is recognized when a user submits a print file with a command similar to the following:

```
$ PRINT/PARAM=DATA_TYPE=AUTOMATIC file-spec
```

When the AUTOMATIC data type is used, the symbiont behaves similarly to other emulation modes, except that no setup modules from the device control library are sent to the printer.

6.2.4 Defining Default Data Types

When determining the data type of files as they are submitted for printing, the symbiont starts by assuming them to be ASCII files, and allows that assumption to be overridden by various criteria such as file type, the DATA_TYPE parameter, and the first few bytes of the file (see Section 3.1.5 *"File-type Translation*” on page 54 for details).

In some circumstances it may be convenient to override (on a queue-specific basis) the system-wide default data type to some other value. This is done by defining the logical:

```
SSV$queue_DEFAULT_DATA_TYPE data-type-spec
```

Where “queue” is the name of the queue for which you wish to change the default data type and “data-type-spec” is the specification for the data type you wish to use. Possible data types are:

- ASCII
- ANSI
- POSTSCRIPT

- HPGL
- PCL
- AUTOMATIC
- EMULATION

For example, if you want to define the data type HPGL to be the default data type for a queue named LASER, create the following logical name definition:

```
$ DEFINE/SYS SSV$LASER_DEFAULT_DATA_TYPE HPGL
```

Chapter 7

Font Management

An important capability in PostScript printing is the ability to download PostScript fonts. ScriptServer provides two forms of font downloading:

- job-specific, and
- permanent.

Job-specific fonts are downloaded to the printer at the beginning of a print job. These fonts “go away” at the end of the job. Permanent fonts are downloaded to the printer at queue startup. These fonts remain in the printer's memory until it is powered off.

Fonts are downloaded from the directory located by the logical name:

```
SSV$TYPEFACE_DIRECTORY
```

If this logical name doesn't exist, fonts are downloaded from the SCRIPTSERVER directory. Font files must have the file type PFB.

Depending on the media format your fonts were supplied on, you may need to upload fonts from an IBM-compatible personal computer or an Apple Macintosh or Macintosh compatible clone. Both of these procedures are described below.

NOTE: *ScriptServer’s PCL/PJL support does not currently include support for font downloading or full re-encoding of fonts. Additionally, PCL uses ISO Latin-1Character Set rather than DEC Multi-national Character Set which may result in some subtle font substitution.*

7.1 DOWNLOADABLE FONT FORMATS

Fonts that are downloaded by ScriptServer can be in one of two formats: ASCII (unpacked) or PC-format (packed). The symbiont automatically determines the format of the font file at the time it is downloaded.

7.2 DOWNLOADING JOB-SPECIFIC FONTS

Job-specific fonts are downloaded through the use of the PRELOAD print parameter, which may be used to specify a single font or a list of fonts. For example, suppose a PostScript file requires the font “BrushScript”. The appropriate PRINT command is:

```
$ PRINT/PARAM=PRELOAD=BRUSHSCRIPT file-spec
```

If more than one font is required to be pre-loaded, the list must be separated by commas and enclosed in quotes. For example:

```
$ PRINT/PARAM=PRELOAD="UTOPIA-REGULAR,UTOPIA-BOLD" file-spec
```

The SCRIPT facility determines whether a font requires downloading and automatically uses the PRELOAD parameter with the required pre-loaded fonts.

A special logical name table (SSV$DOWNLOAD_TABLE) is used to specify downloadable fonts to the ScriptServer software. This table is accessed by both the SCRIPT Facility and the ScriptServer print symbiont.

For ScriptServer to be able to identify downloadable fonts, they must exist in the search list specified by the logical name SSV$FONTS found within SSV$DOWNLOAD_TABLE located within the ScriptServer Directory. On queue startup, the ScriptServer symbiont checks to see if fonts in this list are present on the printer, setting the appropriate queue characteristics if they are.

7.3 DOWNLOADING PERMANENT FONTS

Permanent fonts are downloaded (if not already resident in the printer) at queue startup. Fonts that are to be permanently resident must be specified in the logical name table:

```
SSV$DOWNLOAD_TABLE
```

This table is created at system startup by the procedure DOWNLOAD.COM, which is called in turn by the ScriptServer startup command procedure:

```
        SYS$MANAGER:SSV$STARTUP.COM
```

Queue-Specific Permanent Fonts

The logical names defined in the SSV$DOWNLOAD_TABLE are of the format:

```
        SSV$queue-name_FONTS
```

where "queue-name" is the name of the queue to which you want to assign the permanently downloaded fonts. This logical name has multiple definitions (similar to search lists) where each definition is the name of a font. For example, suppose that the fonts "Optima", "Optima-Italic", "Optima-Bold", and "Optima-BoldItalic" are to be permanently downloaded to the queue PSQUEUE. The correct command for the logical name definition would be:

```
        $ DEFINE/TABLE=SSV$DOWNLOAD_TABLE SSV$PSQUEUE_FONTS -
        _$ "Optima",-
        _$ "Optima-Italic",-_
        _$ "Optima-Bold",-_
        _$ "Optima-BoldItalic
```

NOTE: *As with all operators and identifiers in PostScript, font names are case-sensitive. For downloadable fonts to be correctly accessed from the SCRIPT facility, the exact PostScript font name must be specified and enclosed in quotes.*

The procedure DOWNLOAD.COM contains comments demonstrating how to create downloadable font logical name definitions.

System-Wide Permanent Fonts

You may define a font or group of fonts that are to be permanently downloaded to all ScriptServer queues using the logical name SSV$BLANKET_FONTS. For example, to specify that the fonts "Futura" and "Futura Oblique" are to be permanently downloaded to all ScriptServer-controlled printers, use the command:

```
        $ DEFINE/TABLE=SSV$DOWNLOAD_TABLE SSV$BLANKET_FONTS -
        _$ "Futura",-
        _$ "Futura-Oblique"
```

7.4 DEFINING CHARACTERISTICS FOR FONTS

To provide full functionality for downloadable fonts, you can assign OpenVMS queue characteristics to font names the same way that they are assigned to the supported printer-resident fonts. This is done within the site-specific queue characteristics command procedure:

```
SCRIPTSERVER:SITE_DEF.COM
```

which is functionally the same as the CHAR_DEF.COM procedure, distributed with ScriptServer. The CHAR_DEF procedure calls the SITE_DEF procedure when it is executed, so that when characteristics need to be defined, both ScriptServer-standard and site-defined characteristics are created. SITE_DEF.COM is distributed with ScriptServer and contains examples of how to create queue characteristic definitions.

7.5 UPLOADING FONTS FROM A PC

There are two general methods that may be used to upload fonts that are distributed on PC disks:

- transferring over a serial terminal connection using a terminal emulation program, and
- transferring over a network using PATHWORKS.

These methods apply to the transfer of both Adobe-specified packed format Type 1 fonts (from PC distribution diskettes) and the unpacked ASCII format.

The default file type for font files is PFB, which is the file type used on Adobe-distributed PC diskettes for DOS. Due to the restricted file name length of eight characters under DOS, the font file names on the floppy disks are rarely the same name as that needed by ScriptServer for downloading. Refer to the section concerning the FONT_RENAMER command procedure (see below) for information on renaming font files once they have been transferred to the VAX.

Uploading with a Terminal Emulation Program

When transferring font files with a terminal emulation program, use ASCII transfer mode with 512-byte records. After copying the fonts, use the FONT_RENAMER procedure (see below) to give them names that can be used by ScriptServer software.

Uploading with PATHWORKS

To upload fonts over a network using PATHWORKS software, a file server service must first be set up with the qualifier /ATTRIBUTES= SEQUENTIAL. To create a font-uploading service named FONTS on the PATHWORKS file server, follow the steps below. In this example, substitute the name of the user that will be moving fonts for user-name.

```
$ ADMIN
PCSA_MANAGER> ADD
SERVICE/DIREC/ROOT=SSV$TYPEFACE_DIRECTORY FONTS
PCSA_MANAGER> SET FILE_SERVER SERVICE FONTS/ATTR=SEQ/PERM
PCSA_MANAGER> GRANT/ACCESS=(R,W,C) user-name FONTS
PCSA_MANAGER> START FILE_SERVER CONNECTIONS)
PCSA_MANAGER> EXIT
```

To set up for transferring fonts on the PC:

1. Verify that a disk service is available with the USE command:

```
C:\> USE
```

2. Create a connection on the PC to the FONTS service on the file server.

   In this example, we are using the disk named E:. If drive E: is in use on your PC, you should substitute another value in the range D through M.

```
C:\> USE E:/CONN/NET

       Node Name:(enter node name of file server)
       Service: FONTS(specify the FONTS file service)
       Username:(enter user-name specified above)
       Password:(enter password to user-name)

C:\>
```

You are now ready to copy fonts to drive E: as though it were a local PC disk. For example, if the floppy disk with the fonts on it is in drive A:, issue the following command:

```
C:\\> COPY A:*.PFB E:
```

7.5.1 Using FONT_RENAMER.COM

To make it easier to get file names into a usable form for ScriptServer, the software is distributed with the file:

```
SCRIPTSERVER:FONT_RENAMER.COM
```

This procedure scans all of the files in:

```
SSV$TYPEFACE_DIRECTORY
```

with a file type of .PFB. It then parses the first line of the file for the font's name and renames the file accordingly.

7.6 UPLOADING FONTS FROM A MACINTOSH

To upload fonts from a Macintosh or Macintosh compatible clone, use the following process:

1. In an application on your Macintosh (i.e. Microsoft Word), create a document that uses only the desired font.
2. Stop a ScriptServer queue on your OpenVMS system with the command:

   ```
   $ STOP/RESET queue-name
   ```

   where “queue-name” is the name of the queue you are stopping.
3. Send the document to a stopped ScriptServer queue on your OpenVMS system using PATHWORKS for Macintosh
4. Rename the file holding in the ScriptServer Queue from the temporary filename given it, to one with the name:

   ```
   SSV$TYPEFACE_DIRECTORY:font_name.PFB
   ```

   where “font_name” is the name of the font you are after.

   To do this:

- locate the file holding in the queue with the SHOW/QUEUE command:

  ```
  $ SHOW/QUEUE/FUL/ALL queue-name
  ```

  This can be found in the job list of the queue, in the final line of the job entry in a for-mat similar to:

  ```
  File: _NODE:[SYS0.MSA.MSAP$SPOOL]199507261051411974_
  -MSAP$RCVRO.PS;1/DELETE
  ```

where “NODE:” is the node name of your OpenVMS host

NOTE: *All file specifications (in bold type above) will be on one line.*

- Rename the file to the ScriptServer Directory with the following command:

```
$ RENAME _NODE:[SYS0.MSA.MSAP$SPOOL] -
_$ SSV$TYPEFACE_DIRECTORY:font_name.PFB
```

5. You can now delete the entry holding in the stopped ScriptServer queue:

```
$ DELETE/ENTRY entry-number
```

6. Using your favorite text editor, edit the renamed file.
7. FIND “beginfont”
8. FIND “endfont”
9. SELECT the area in between. This is your Font File
10. Delete all other text from the font file.
11. Convert the embedded carriage returns in the file to actual carriage returns. If you fail to do this, the file will be too large for RMS (Record Management System).
12. Save the file.

Your font is now ready for downloading.

7.7 CONTROLLING DOWNLOAD MESSAGES

When downloading permanent fonts or setup modules (see Chapter 5 above), the ScriptServer print symbiont sends messages to OPCOM with the name of the module or font being downloaded. By default, it does this on both job-specific and permanent downloads. This behavior can be modified on a system-wide or queue-specific basis by defining the appropriate system-wide logical name.

To define download message handling for all queues system-wide, use the logical name:

```
SSV$DOWNLOAD_MESSAGES.
```

For queue-specific message handling, use the name:

```
SSV$queue_DOWNLOAD_MESSAGES
```

Define the logical name to the value:

- ALL
- NONE, or
- PERMANENT

For example, if you want messages for all queues system-wide to indicate only setup modules and fonts that are permanently downloaded at queue startup, use the command:

```
$ DEFINE/SYSTEM SSV$DOWNLOAD_MESSAGES PERMANENT
```

To disable all download messages on a single queue named PSQUEUE, use the command:

```
$ DEFINE/SYSTEM SSV$PSQUEUE_DOWNLOAD_MESSAGES NONE
```

NOTE: *Similar messages are generated for downloaded Setup modules (See Chapter 5). All download messages are controlled by the above logical names*

Chapter 8

Paper and Envelope Handling

This chapter describes the methods used in the ScriptServer software to manage paper trays and envelope feeders and how it can be customized to handle specific requirements at customer sites.

8.1 QUEUE STARTUP PROCESSING

8.1.1 Determining Paper Sizes Available

At queue startup, the ScriptServer print symbiont checks for the existence of each of the supported paper sizes and declares the corresponding queue characteristics for the sizes that are found.

The symbiont determines the presence of paper types by executing the tray selection commands for each size. If the command executes without error, the paper size is present. If the command generates

an error, the paper is not available. When an error occurs, it will be either range check or undefined for PostScript printers or unknown envelope or unknown paper for PCL printers.

Early model PostScript printers (such as the Apple LaserWriter) do not support tray selection commands. On these printers, the symbiont recognizes that all paper tray selection commands fail and assumes that the only tray available contains letter-sized paper.

8.1.2 Determining Envelope Feed Capability

The symbiont determines whether envelope feed capability is present by testing two Boolean variables in the PostScript dictionary statusdict. If either variable has the value true, envelope feed capability is present. The variables checked are feedsenvelopes and envelopefeeder.

NOTE: *Envelopes are not supported when printing in PCL mode.*

8.2 JOB STARTUP PROCESSING

8.2.1 Paper Selection at Job Startup

At job startup, the ScriptServer print symbiont issues the appropriate tray selection command for the paper size specified for the print job. Rather than selecting trays based on tray number (which would be device-dependent), it selects based on the paper size desired, allowing the printer to map the desired size to a particular tray.

The specific commands used to select trays are defined in the PostScript dictionary *statusdict*, and are the name of the paper size followed by tray. For example, the command used to select legal paper is legaltray.

8.2.2 Envelope Handling at Job Startup

At job startup for an envelope print job, the process is a little more complex because of the wide range of envelope sizes (more than 20 different sizes) supported by the various PostScript printers.

Specific envelope sizes are defined through the use of the logical name:

```
SSV$queue_ENVELOPE_COMMAND
```

where “queue” is the name of the queue controlling the envelope-fed printer. If this logical name is not defined, the symbiont defaults to the envelope size 4.125 by 9.5 inches. ScriptServer supports the following specific envelope sizes as arguments to the above logical:

- COM10;
- Monarch;
- DL.

The following table indicates the appropriate PostScript operator per envelope size that may be used with most PostScript printers to customize envelope handling:

Table 8-1:Envelope PostScript Operators

Envelope Size	PostScript Operator
COM10	com10envelopetray
Monarch	monarchenvelopetray
DL	dlenvelopetray

The following is an example which defines the Envelope Command Logical for use with COM10 envelopes:

```
$ DEFINE/SYSTEM SSV$queue_ENVELOPE_COMMAND "com10envelopetray"
```

where “queue” is the name of the queue to which you wish to associate the envelope command.

NOTE: *PostScript operators are case-sensitive and need to be in lowercase. Additionally, some printers do not support these versions of these operators, in which case you will want to follow the instructions in the following section.*

Variations in Envelope Commands

Because some printers do not support the “tray” version of the PostScript operators mentioned above (i.e. com10envelopetray) the paper tray selection PostScript operator may be necessary as well. In this case, the PostScript should read as indicated in Table 8-2: on page 135:

Table 8-2:Variations of Envelope Operators

Envelop Size	PostScript Operator
COM10	com10envelopetray 3 setpapertray
Monarch	monarchenvelopetray 3 setpapertray
DL	dlenvelopetray 3 setpapertray

NOTE: *In the above table the paper tray selection portion of the operator given is particular to the QMS1725. It will be necessary to consult printer documentation to determine the appropriate PostScript for your particular printer.*

The Envelope Command for Unspecified Sizes

The logical name must be defined as the PostScript operator that invokes the envelope feeder and establishes the correct graphic state on the printer for the envelope size to be used. Unless the logical name is defined with one of the specific envelope sizes mentioned above, it will need the PostScript operator in the following format:

```
envelope_heightxenvelope_widthenvelopetray
```

where “envelope_height” is the height of the envelope and “envelope_width” is the width of the envelope.

NOTE: *ScriptServer assumes these measurements to be given in inches if there is a decimal (.) in the number. If there is no decimal, ScriptServer assumes the measurement is metric and given in millimeters.*

The following example defines the logical for an envelope measuring 4.125 inches in height and 9.5 inches in width:

```
$ DEFINE/SYSTEM SSV$queue_ENVELOPE_COMMAND "4.125x9.5envelopetray"
```

where “queue” is the name of queue to which you wish to associate the envelope command.

8.3 CUSTOMIZING PAPER AND ENVELOPE HANDLING

Under certain circumstances, it may be necessary to modify the commands used by ScriptServer for paper tray selection. This is frequently the case when using the envelope feeder due to the variety of supported envelope sizes

.

NOTE: *Customizing the PostScript commands used by the print symbiont requires a basic knowledge of the PostScript language. You must provide error-free commands to the symbiont or unexpected results will occur. Make certain that the commands work by using the PostScript interactive mode invoked with the operator **executive**. These printer-specific commands are generally explained in the document distributed with the printer called the **PostScript Language Supplement**.*

8.3.1 Customizing Paper Tray Selection

When modifying tray selection commands:

1. determine the command that would be used by ScriptServer; and
2. create a logical name that dictates what will replace the command.

The logical name format for alternate tray selection is:

```
SSV$device_TRAY_tray-command
```

where “device” is the device name of the port that the queue is on, and “tray-command” is the command you are replacing. The symbiont always places the dictionary *statusdict* on the stack before executing tray selection commands. You need not include *statusdict* begin in your redefined command.

For example, suppose you want to redefine the command used to invoke the A4 paper tray on a queue connected to the device LTA9. You have determined that the command required:

```
2 setpapertray a4
```

The command used by ScriptServer would be a4tray, so the logical name definition would be:

```
$ DEFINE/SYSTEM SSV$LTA9_TRAY_A4TRAY "2 setpapertray a4"
```

NOTE: *Because PostScript is case-sensitive, the definition of the command is in lower case and is enclosed in quotation marks.*

Or, suppose a printer without backchannel (see Section 4.2.1 *"Setting Up a Parallel Connection*” on page 66) does not support paper tray selection commands at all. The symbiont has no way of knowing what commands generate errors because there is no communication path back from the printer. In this case, the symbiont would use the command letter tray which would generate the error undefined. In this case, the appropriate logical definition (still assuming a device name of LTA9) is:

```
$ DEFINE/SYSTEM SSV$LTA9_TRAY_LETTERTRAY " "
```

This completely disables the use of tray selection commands, avoiding the undefined error.

8.3.2 Accessing Secondary Paper Trays

On PostScript printers that support secondary paper trays, it is possible to define a form in conjunction with a setup module that allows jobs printed on that form to access a secondary tray automatically. Trays are accessed using the operator *setpapertray* preceded by the tray number desired. It may also be necessary to execute a paper setup operator that sets up the PostScript graphics state for the different page size.

For example, suppose that you want to define a form named LETTERHEAD that automatically loads paper from a secondary tray. You have determined that the correct tray number to use is 2 by consulting the documentation for your printer.

1. Create a form setup module with your favorite editor, giving it the filename LTRHD.PS,containing:

   ```
   statusdict begin 2 setpapertray end letter
   ```

2. Place the module in the device control library as follows:

   ```
   $ LIBR/INSERT SYS$LIBRARY:SSV$DEVCTL.TLB LTRHD.PS
   ```

3. Define a form that references the setup module:

   ```
   $ DEFINE/FORM/SETUP=LTRHD/STOCK=DEFAULT LETTERHEAD 99
   ```

8.3.3 Customizing Envelope Handling

The problem of envelope setup commands is slightly different than that for paper trays. By default, the symbiont uses the command 4.125x9.5*envelopetray* to both select the envelope feeder and set up the graphics state for printing 4.125 by 9.5 inch envelopes. To use a different command, a system-wide logical name must be defined in the format:

```
SSV$queue_ENVELOPE_COMMAND
```

where “queue” is the name of the ScriptServer queue.

Some printers that support envelope feeders do not support simultaneous tray selection within the same command. In this case, two commands must be executed in the logical name definition. For example, the Dataproducts LZR-1260 requires both a tray selection command that tells the printer to load envelopes from the feeder, and an envelope setup command that sets up the graphics state. If such a printer is connected on a queue named PSQUEUE and the desired envelope size is an 8 by 11 inch letter-sized envelope, the command to define the logical name is:

```
$ DEFINE/SYSTEM SSV$PSQUEUE_ENVELOPE_COMMAND -
_$ "3 setpapertray letterenvelope"
```

Chapter 9

Using the SCRIPT Facility

SCRIPT is the menu-driven print request facility provided with the ScriptServer software. It is used to describe the various printing parameters used by ScriptServer when translating ASCII files into PostScript.

NOTE: *PCL support is not available from the SCRIPT menu. ScriptServer will submit ASCII text files as PCL via the PRINT command based on the value of the logical:*

```
SSV$TEXT_LANGUAGE
```

described in Chapter 10. The formatting of the file, however, will depend on the setting established via the printer's control panel.

SCRIPT is implemented as a standard DCL command and is designed to behave like the PRINT command, in that it supports many of the same command qualifiers. The SCRIPT facility operates in two modes:

- command-line and
- menu.

To invoke the menu, use the command:

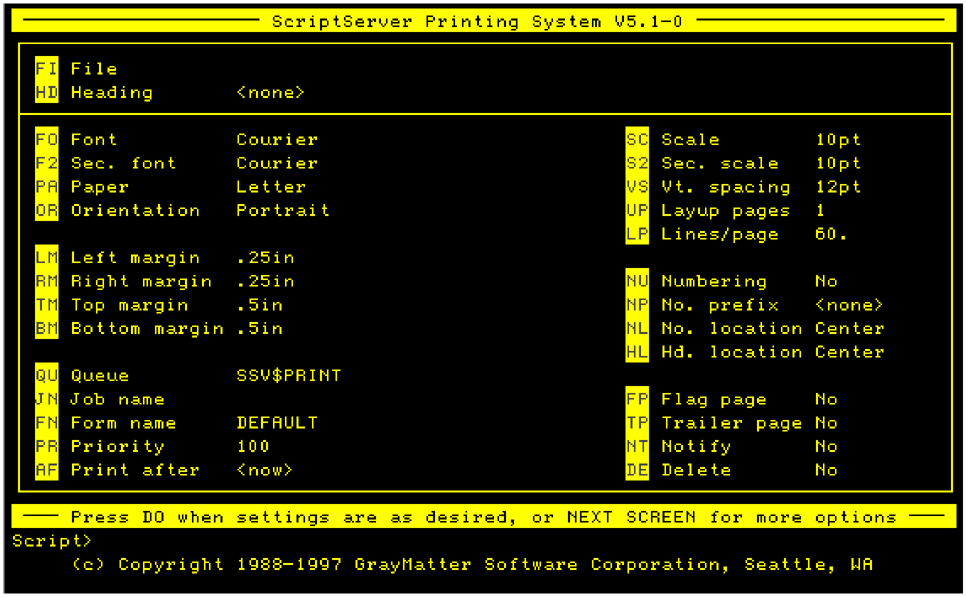
```
$ SCRIPT file-spec
```

where "file-spec" is the name of the file that you want to print. To bypass the menu, use the /GO qualifier, optionally with any other SCRIPT qualifier. For example:

```
$ SCRIPT/GO/AFTER=18:00/FORM=GREEN-BAR file-spec
```

9.1 USING THE MENU

The SCRIPT Facility menu consists of the main menu (illustrated in Figure 9-1), a secondary screen with additional formatting options (Figure 9-2), and several pop-up menus (one of which is illustrated in figure 9-3).



```
 ──────────────────── ScriptServer Printing System V5.1-0 ────────────────────

  FI File
  HD Heading        <none>

  FO Font           Courier                       SC Scale        10pt
  F2 Sec. font      Courier                       S2 Sec. scale   10pt
  PA Paper          Letter                        VS Vt. spacing  12pt
  OR Orientation    Portrait                      UP Layup pages  1
                                                  LP Lines/page   60.
  LM Left margin    .25in
  RM Right margin   .25in                         NU Numbering    No
  TM Top margin     .5in                          NP No. prefix   <none>
  BM Bottom margin  .5in                          NL No. location Center
                                                  HL Hd. location Center
  QU Queue          SSV$PRINT
  JN Job name                                     FP Flag page    No
  FN Form name      DEFAULT                       TP Trailer page No
  PR Priority       100                           NT Notify       No
  AF Print after    <now>                         DE Delete       No

 ─── Press DO when settings are as desired, or NEXT SCREEN for more options ───
Script>
     (c) Copyright 1988-1997 GrayMatter Software Corporation, Seattle, WA
```

Figure 9-1: SCRIPT Facility Main Menu

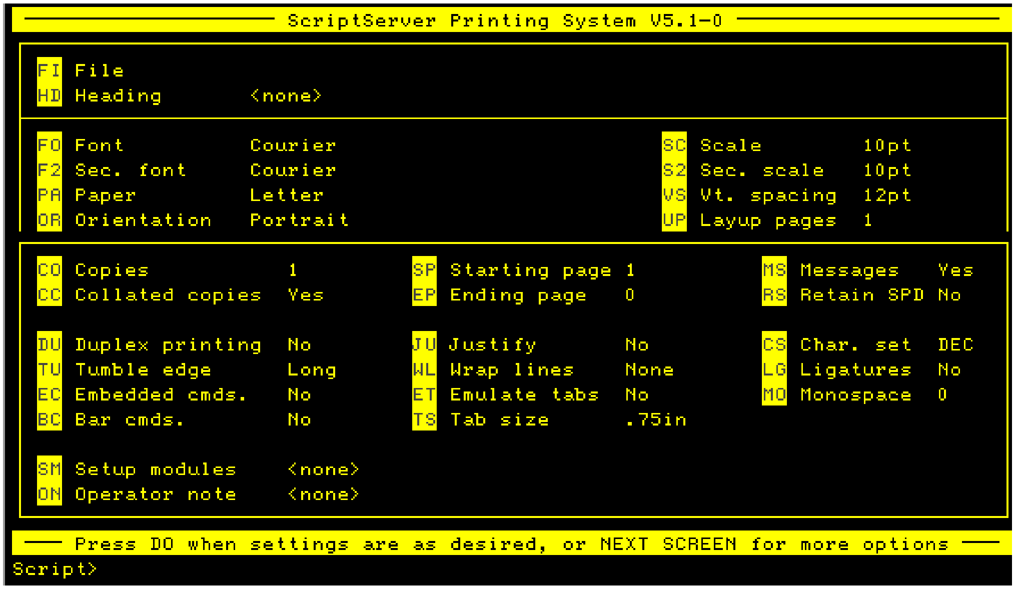
The command format within the SCRIPT menu consists of a two-letter command, represented by the bold text in the menu, followed by an optional parameter which is the new setting for that command. The following example changes the file to be printed to GL.LIS:

```
Script> FI GL.LIS
```

Menu items that have Yes / No values may be toggled to their opposite values by entering the two-letter command without a parameter. See the below for a detailed description of each of the commands supported in the menu.

Help may be obtained for any command by typing HELP followed by the command for which help is desired.

After the various settings in the menu have been adjusted to their desired values, press the DO key to submit the file(s) for printing. On terminals without a DO key, use the GO command instead. A secondary menu window containing additional print settings is available by pressing the NEXT SCREEN key. On terminals without this key, it may be accessed by pressing CTRL / N.



```
 ────────────────────  ScriptServer Printing System V5.1-0  ────────────────────

  FI File
  HD Heading       <none>

  FO Font          Courier                         SC Scale        10pt
  F2 Sec. font     Courier                         S2 Sec. scale   10pt
  PA Paper         Letter                          VS Vt. spacing  12pt
  OR Orientation   Portrait                        UP Layup pages  1

  CO Copies           1         SP Starting page 1          MS Messages   Yes
  CC Collated copies  Yes       EP Ending page   0          RS Retain SPD No

  DU Duplex printing  No        JU Justify       No         CS Char. set  DEC
  TU Tumble edge      Long      WL Wrap lines    None       LG Ligatures  No
  EC Embedded cmds.   No        ET Emulate tabs  No         MO Monospace  0
  BC Bar cmds.        No        TS Tab size      .75in

  SM Setup modules    <none>
  ON Operator note    <none>

 ─── Press DO when settings are as desired, or NEXT SCREEN for more options ───
Script>
```

Figure 9-2: SCRIPT Facility Secondary Menu

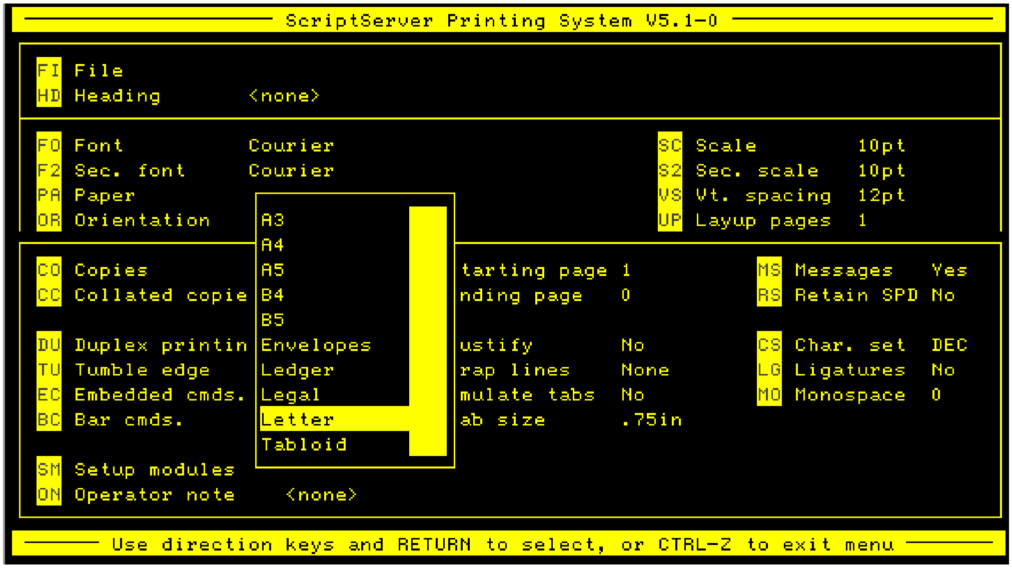
Functional Overview

The SCRIPT facility is essentially a front-end menu to the ScriptServer print symbiont. When you press the DO key, it creates a temporary ScriptServer Page Description (SPD) file in the directory identified by the logical name SYS$SCRATCH. This file contains the various selections you make that establish the parameters for how your file(s) are to be printed.

The print request is created referencing the SPD file using the ScriptServer-specific print parameter SPD$FILE. Once the print job is complete, the SPD file is deleted by the symbiont (unless directed to keep it using the /RETAIN qualifier or the RS command in the menu).

Pop-Up Menus

For certain commands, the SCRIPT facility creates pop-up menus (illustrated in Figure 9-3) allowing selection from a list of valid parameters.



```
 ──────────── ScriptServer Printing System V5.1-0 ────────────

 FI File
 HD Heading           <none>

 FO Font              Courier                    SC Scale         10pt
 F2 Sec. font         Courier                    S2 Sec. scale    10pt
 PA Paper                                        VS Vt. spacing   12pt
 OR Orientation      A3                          UP Layup pages   1
                     A4
 CO Copies           A5          tarting page 1         MS Messages    Yes
 CC Collated copie   B4          nding page   0         RS Retain SPD  No
                     B5
 DU Duplex printin   Envelopes   ustify       No        CS Char. set   DEC
 TU Tumble edge      Ledger      rap lines    None      LG Ligatures   No
 EC Embedded cmds.   Legal       mulate tabs  No        MO Monospace   0
 BC Bar cmds.        Letter      ab size      .75in
                     Tabloid
 SM Setup modules
 ON Operator note      <none>

 ──── Use direction keys and RETURN to select, or CTRL-Z to exit menu ────
```

Figure 9-3: Pop-Up Paper Selection Menu

For example, by typing “PA” at the SCRIPT prompt, you will be presented a list of the valid paper sizes on which you may print this file. Pop-up menus are available for the following commands:

- queue names (QU command)
- font names (FO and F2 commands)
- library settings (LS and DS commands)
- form names (FN command)
- heading and numbering locations (HL and NL commands)
- paper sizes (PA command)
- n-up (UP command)

NOTE: *The pop-up menu for queues may be disabled with the ScriptServer logical definition:*

```
$ DEFINE/SYSTEM SSV$SCRIPT_QUEUE_LOOKUP INHIBIT
```

*or with the **/SUPRESS** qualifier described below.*

Additionally, the Queue Pop-up menu will not display any PCL-only queues.

Font Selection Menu

When ScriptServer has been installed with queue characteristics enabled (see Section 2.4 "*ScriptServer Files*” on page 47), the font selection menu accurately represents the fonts available on the currently selected queue, as well as those available for downloading. Fonts tagged with an asterisk (*) will need to be downloaded prior to the job. If a downloadable font is selected, the SCRIPT facility submits the job with the “PRELOAD= font-name” parameter to request that the symbiont download the font.

This capability extends to generic queues. If the currently selected queue is a generic queue servicing several ScriptServer queues, the fonts displayed in the pop-up menu list are all of the fonts available on those queues targeted by the generic queue.

9.2 SETTINGS LIBRARIES

The SCRIPT facility allows you to save groups of settings in a file called the settings library. Once you have established a frequently-used setting, it may be stored under a setting-name in the settings library using the SS (save settings) command. Settings can then be invoked by setting-name using the LS (load settings) menu command, or the /SETTING command qualifier. There are five commands in the SCRIPT menu for controlling the settings library. They are:

- SS save settings
- LS load settings
- DS delete settings
- SL show library
- LI change library

The settings library is located by SCRIPT using the logical name SPD$LIBRARY. If this logical name doesn't exist, SCRIPT uses the file:

```
SYS$LOGIN:SPD_LIBRARY.DAT
```

NOTE: *If the settings library doesn't exist when a setting-related command is used, it is created automatically.*

Default Settings in SCRIPT

SCRIPT supports automatic loading of default settings from the settings library. On invocation, SCRIPT attempts to resolve the logical name SSV$SCRIPT_DEFAULTS, which is used to define a setting-name in the library as the default setting. For example, if your settings library contained a setting named MY-DEFAULTS, you could make it your default setting by defining the following logical name in your LOGIN.COM:

```
$ DEFINE SSV$SCRIPT_DEFAULTS MY-DEFAULTS
```

You can also change the default queue in the SCRIPT menu system-wide by defining the following logical to any existing ScriptServer queue:

```
SSV$SCRIPT_DEFAULT_QUEUE
```

The following logicals allow you to associate specific SCRIPT settings for use when printing ASCII files with the PRINT command:

```
SSV$DEFAULT_SETTINGS

SSV$queue_SETTINGS

SSV$FORM_form_SETTINGS
```

SSV$SCRIPT_OVERRIDE

When defined as follows, this logical allows you to override the SCRIPT setting value for headers and page numbering via the PRINT command through use of the /HEADING and /PAGE_NUMBER qualifiers:

```
$ DEFINE/SYSTEM SSV$SCRIPT_OVERRIDE “HEADINGS,PAGE_NUMBERING”
```

For example, if the SCRIPT setting associated with a particular queue does not include a heading value, by having this logical defined, you can add a heading to the print job by printing the file through the PRINT command with the /HEADING qualifier.

This logical name can be set to either HEADINGS or PAGE_NUMBERING specifically. For example, to override HEADING values only, define the logical as follows:

```
$ DEFINE/SYSTEM SSV$SCRIPT_OVERRIDE “HEADINGS”
```

9.3 SCRIPT COMANDS

Table 9-1: on page 147 summarizes the menu commands available in the SCRIPT facility. Each of the commands is described in detail below:

Table 9-1:Script Menu Command Summary

Command	Controls
AF	*Print after*
BC	*Bar commands*
BM	*Bottom margin*
CC	*Collated copies*
CO	*Copies*
CS	*Character set*
DE	*Delete file after printing*
DS	*Delete settings from library*
DU	*Duplex (two-sided) printing*
EC	*Embedded commands*
EP	*Ending page*
ET	*Emulate tabs*
FI	*File specification*
FN	*Form name*
FO	*Primary font*
FP	*Flag page*
F2	*Secondary font*
HD	*Page heading*
HL	*Heading location*
JN	*Job name*
JU	*Justification*

Table 9-1:Script Menu Command Summary (Continued)

Command	Controls
LI	*Change settings library*
LP	*Lines per page*
LM	*Left margin*
LG	*Ligatures*
LS	*Load settings from library*
MO	*Monospace*
MS	*Messages*
NL	*Page numbering location*
NP	*Page numbering prefix*
NU	*Page numbering*
NT	*Completion notification*
ON	*Operator note*
OR	*Page orientation*
PA	*Paper size*
PR	*Job priority*
QU	*Queue name*
RM	*Right margin*
RS	*Retain SPD file*
S2	*Secondary font scale*
SC	*Primary font scale*
SPAWN	*Spawn subprocess*
SS	*Save settings in library*
TS	*Tab size*
TM	*Top margin*
TP	*Trailer page*
TU	*Tumble edge*
UP	*Number up*
VS	*Vertical line spacing*
WL	*Wrap lines*

9.3.1 AF - Print after

Format: AF time expression

Default: <now>

Equiv. Qualifier: /[NO]AFTER

This command holds a job until the specified time. The time must be specified as either an absolute time or as delta time. If the specified time has passed, the job is queued for printing immediately.

Absolute Time Expressions

The acceptable format for an absolute time expression is as follows:

```
[dd-mmm-yyyy hh:mm]
```

dd	*day of the month; an integer in the range of 1-31*
mmm	*month specified as JAN, FEB, MAR, etc.*
yyyy	*year; as a four digit integer*
hh	*hour of the day; in the integer range of 0-23*
mm	*minute of the hour; integer range of 0-59*

The following restrictions also apply:

- When specifying both the date and time, you must put a space between them.
- Use hyphens when specifying the date.
- You can omit any of the fields within the date or time, as long as you type the punctuation marks that separate the fields.
- If you omit the date, the default is today's.
- If you omit the time, the default is 00:00.
- You may also use certain keywords:
    - TODAY
    - TOMORROW
    - NOW

Example of an absolute time specification:

```
31-DEC-1988 13:30 1:30 PM on December 31, 1988
```

Delta Time Expressions

An expression in delta time signifies a change in time, not necessarily a specific date. You would use delta time if, for example, you wanted to delay a print job for several hours or days. The format for entering delta time expressions is as follows:

```
[dddd-][hh:mm:ss.cc]
```

dddd	*number of days; an integer in the range of 0-9999*

The following restrictions also apply:

- (the time expression follows the format described above, however you must remember that it is a change in time, not absolute time)
- If you specify both the number of days and the time, type a hyphen between them.
- If you specify the number of days only, you must include the hyphen (-).
- You can truncate the time on the right, and can omit any fields within the time so long as you type the punctuation marks that separate the fields.

Examples of a delta time expression:

+30	*thirty days into the future*
+21::45	*twenty-one hours, 45 seconds into the future*
+10-15:30	*ten days, fifteen hours, thirty minutes into the future*

NOTE: *Refer to your OpenVMS documentation for more detailed information on time expressions.*

9.3.2 BC - Bar commands

Format: BC { YES | NO }

Default: No

Equiv.Qualifier: /[NO]BAR

This command is used to indicate whether embedded PostScript commands are contained in the print file. These commands are lines that start with a vertical bar character (|) in the first print position of the record. When being handled by the ScriptServer symbiont, bar command lines are sent to the printer without translation. The vertical bar character is stripped before sending the command to the printer.

9.3.3 BM - Bottom Margin

Format: BM size-expression

Default: 5 inch

Equiv. Qualifier: /BOTTOM_MARGIN=size-expression

The bottom margin command specifies the lowest position on the page on which text may fall. Any attempt to place text from the input file below the bottom margin results in starting a new page. Page numbering text (if requested) is the only text that is printed below the bottom margin. Additionally, because the bottom margin affects the length of the page, it also affects the number of lines per page which is determined by the vertical spacing. See the LP command below for more details on how the one affects the other.

9.3.4 CC - Collated copies

Format: CC { YES | NO }

Default: Yes

Equiv. Qualifier: /[NO]COLLATED

This command determines whether or not multiple copies of the file are grouped by page. Printing uncollated copies is considerably faster than collated copies (the default) because each page is transmitted to, and imaged on, the printer only once.

9.3.5 CO - Copies

Format: CO copy-count

Default: 1 copy

Equiv. Qualifier: /COPIES=copy-count

The CO command allows you to specify the number of copies that are to be printed. It works in conjunction with the CC (collated copies) command (see above) to determine whether or not the printed output is grouped by page.

9.3.6 CS - Character set

Format: CS { DEC | PC | CST}

Default: DEC

Equiv. Qualifier: /CHARACTER_SET

This command controls which extended ASCII character set is to be used to print the file: DEC, PC or CST. The default value of “DEC” causes the DEC Multinational Character Set (DEC-

MCS) to be used. “PC” allows the use of IBM PC-compatible character set, excluding the line-drawing characters, for which there is no direct equivalent in PostScript fonts. CST utilizes a custom character set. Refer to Section 10.3.13 *"ENCODING Parameter*” on page 192 for instructions on customizing your own character set.

9.3.7 DE - File deletion after printing

Format: DE { YES | NO }

Default: No

Equiv. Qualifier: /[NO]DELETE

The DE command is used to specify whether the input file is to be deleted after printing.

9.3.8 DS - Delete settings

Format: DS setting-name

Default: none

Equiv. Qualifier: none

The DS command specifies a setting-name to be deleted from the settings library. The use of this command requires write access to the current settings library.

9.3.9 DU - Duplex control

Format: DU { YES | NO }

Default: No

Equiv. Qualifier: /[NO]DUPLEX

The DU command controls whether the output file is to be printed on both sides of the paper, which requires a printer with duplexing capability. This command is related to the TU (tumble edge) command, which specifies whether the image on the second side of the page should be rotated 180° for binding on the short edge of the page.

9.3.10 EC - Embedded commands

Format: EC { YES | NO }

Default: No

Equiv. Qualifier: /[NO]EMBEDDED_COMMANDS

This command indicates whether the print file contains any embedded commands. Embedded commands are lines that start with the backslash (\) character. Supported embedded commands are:

\CENTER	centers the next line between margins
\FONT	modifies the current font
\LEFT_MARGIN	modifies the left margin setting
\RIGHT_MARGIN	modifies the right margin setting
\TOP_MARGIN	modifies the top margin setting
\BOTTOM_MARGIN	modifies the bottom margin setting
\RESTART_PAGE	moves current position to top of page
\VERTICAL_SPACING	modifies the vertical line spacing
\[NO]TABS	modifies the tab emulation setting
\[NO]JUSTIFY	changes current justification setting
\WRAP=LONG	wraps lines that exceed the right margin
\WRAP=ALL	wraps short lines up and long lines down
\WRAP=NONE	disables line wrapping

Refer to Chapter 19 for more information on using embedded commands with ScriptServer.

9.3.11 ET - Emulate tabs

Format: ET { YES | NO }

Default: No

Equiv. Qualifier: /[NO]EMULATE_TABS

This command controls how tab characters are handled. By default, tabs are converted to the correct number of spaces to advance to the next tab stop. In proportionally-spaced fonts, this scheme fails because all characters are different sizes. When printing with tab emulation enabled, tabs move the current print position to the next tab stop as determined by the TS (tab size) command.

9.3.12 EP - Ending page

Format: EP page-number

Default: Last page

Equiv. Qualifier: /END_PAGE=page-number

The EP command is used to specify the last page of the file to be printed. This command may be optionally used with the SP (starting page) command to print a subset of pages from the print file.

9.3.13 FI - File specification

Format: FI file-spec

Default: none

Equiv. Qualifier: Command parameter

The FI command is used to alter the file specification of the print file(s). The FI command allows wildcards, but not lists. Unless otherwise specified, the file type defaults to .LIS.

9.3.14 FN - Form name

Format: FN form-name

Default: DEFAULT

Equiv. Qualifier: /FORM=form-name

This command specifies the OpenVMS form name that the file(s) are to be printed on. The form name entered here must be a valid existing OpenVMS form name.

9.3.15 FO - Primary font

Format: FO font-name

Default: Courier

Equiv. Qualifier: /FONT=font-name

The FO command specifies the font in which the body of the text from the input file is to be printed. The font name specified must be a supported printer-resident font (see Table 3-1: on page 59) or a site-defined font (see Chapter 7).

Fonts in the pop-up menu that are tagged with an asterisk (*) are not resident in the printer but are available for downloading.

9.3.16 FP - Flag page

Format: FP { YES | NO }

Default: No

Equiv. Qualifier: /[NO]FLAG

This command is used to specify whether flag pages are to be generated by ScriptServer for this print job.

9.3.17 F2 - Secondary font

Format: F2 font-name

Default: Courier

Equiv. Qualifier: /S_FONT=font-name

Specifies the font for the heading and numbering text. Fonts in the pop-up menu that are tagged with an asterisk (*) are not resident in the printer but are available for downloading.

9.3.18 HD - Page heading

Format: HD heading-text

Default: none

Equiv. Qualifier: /[NO]HEADING=heading-text

The HD command is used to specify text that is to be printed at the top of every page for the job. The heading is centered vertically between the top of the page and the top margin, and is set in the secondary font (specified with the F2 command) at the secondary scale (specified with the S2 command). The horizontal positioning (left, right, inside, outside, or center) of the heading may be specified using the HL command. There are two keywords that may be used here to have the

ScriptServer print symbiont automatically insert the appropriate text:

- $FILE uses the full file specification as the heading
- $DATE uses the date and time of printing as the heading

Heading keywords may be inserted into other text strings, such as:

```
File: $FILE - Date: $DATE
```

9.3.19 HL - Heading location

Format: HL location-keyword

Default: Center

Equiv. Qualifier: /HLOCATION=location-keyword

This command determines the horizontal positioning of the heading text, if any. The keywords for positioning the text are listed in Table 9-2: on page 156.

Table 9-2:Heading/Numbering Location Keyword

Keyword	Text Placement
CENTER	centers the text between the left and right margins
LEFT	justifies the text on the left margin
RIGHT	justifies the text on the right margin
INSIDE	justifies the text on the left margin for odd-numbered pages and on the right margin for even-numbered pages
OUTSIDE	justifies the text on the right margin for odd-numbered pages and on the left margin for even-numbered pages

9.3.20 JN - Job name

Format: JN job-name

Default: File name

Equiv. Qualifier: /NAME=job-name

This command specifies the name of the print job.

9.3.21 JU - Justify

Format: JU { YES | NO }

Default: No

Equiv. Qualifier: /[NO]JUSTIFY

This command is used to invoke full text justification, creating smooth left and right margins. Enabling justification automatically enables full line wrapping.

9.3.22 LG - Ligatures

Format: LG { YES | NO }

Default: No

Equiv. Qualifier: /[NO]LIGATURES

The LG command causes the ASCII-to-PostScript translator to change all occurrences of the “fl” and “fi” character combinations to their ligature counterparts.

9.3.23 LI - Change current settings library

Format: LI file-spec

Default: SYS$LOGIN:SPD_LIBRARY.DAT

Equiv. Qualifier: /LIBRARY=file-spec

The LI command is used to change the current settings library. If the library specification given as the argument to this command does not exist, you will be prompted for whether it should be created. This command is handy for migrating settings between libraries, i.e. between user libraries, or from a user library to the symbiont settings library, where settings can be accessed using the /PARAM= SCRIPT_SETTINGS=setting-name parameter to the PRINT command.

9.3.24 LM - Left margin

Format: LM size-expression

Default: .25 Inch

Equiv. Qualifier: /LEFT_MARGIN=size-expression

This command is used to specify the distance of the left margin from the left edge of the paper.

9.3.25 LP - Lines per page

Format: LP { 1-99 }

Default: 60.

Equiv. Qualifier: /LINES_PER_PAGE

The number of lines per page depends on the vertical spacing between the lines of text on the page. When you change the number of lines per page, both the scale and the vertical spacing change to accommodate that new number. A value for the lines per page that has been calculated according to the vertical spacing and font scaling will be followed by a decimal (For details on how these numbers are determined, see the descriptions of SC and VS below). In order to accommodate the spacing between the lines when you change the vertical spacing, the number of lines per page changes as well.

Additionally, any command that affects the length of the page, affects the number of lines per page. For example, if you change the
of the page from portrait to landscape, the page length shrinks and because the vertical spacing remains the same, the lines per page must necessarily change to accommodate. The same is true for the same reason with regard to both bottom and top margins.

9.3.26 LS - Load settings from library

Format: LS setting-name

Default: none

Equiv. Qualifier: /SETTING=setting-name

The LS command is used to load a previously stored (with the SS command) setting-name from the current settings library. If the command is used without an argument, a pop-up menu is presented with all of the existing settings in the library from which a setting may be selected.

9.3.27 MO - Monospace

Format: MO { 0 - 999 }

Default: 0 (no monospace)

Equiv. Qualifier: /MONOSPACE

The MO command converts proportionally spaced fonts to monospace, with the character width corresponding to the number used as the argument to the command. This value represents the new width for each character in the font expressed in the character coordinate system, and is in the range 0-999. Typical values are 600 and 700.

There is risk involved in changing a proportional font to monospace because while characters in monospaced fonts are carefully and intentionally designed to look correct when evenly spaced, proportionally spaced fonts are not. Generally these converted fonts will look awkward when printed, however this feature is very useful when printing tabular reports.

9.3.28 MS - Messages

Format: MS { YES | NO }

Default: Yes

Equiv. Qualifier: /[NO]MESSAGES

The MS command is used to control whether a Print Log is generated if messages are received from the printer while the job is being printed. For more information about the Print Log, see Section 3.4 *"Error Messaging*" on page 57.

9.3.29 NL - Page numbering location

Format: NL location-keyword

Default: Center

Equiv. Qualifier: /NLOCATION=location-keyword

The NL command is used to determine the horizontal placement of the page numbering text. See Table 9-2: on page 156 above for a list of the placement keywords that may be used with this command.

9.3.30 NP - Page numbering prefix

Format: NP prefix-text

Default: none

Equiv. Qualifier: /[NO]PREFIX=prefix-text

The page numbering prefix allows you to specify a string of text to be prefixed to the page number (if page numbering is in effect).

9.3.31 NU - Page numbering

Format: NU { YES | NO }

Default: No

Equiv. Qualifier: /[NO]NUMBERING

The NU command is used to enable / disable page numbering for the print job. The page number is set with the prefix (see above) midway between the bottom margin and the bottom edge of the paper.

9.3.32 NT - Completion notification

Format: NT { YES | NO }

Default: No

Equiv. Qualifier: /[NO]NOTIFY

The NT command is used to specify whether a completion status message is desired for the print job. It is functionally equivalent to the /NOTIFY qualifier on the PRINT command.

9.3.33 ON - Operator note

Format: ON text-string

Default: none

Equiv. Qualifier: /[NO]NOTE=text-string

The ON command allows you to specify a string of text to be displayed on the flag page (if specified) under a NOTE heading.

9.3.34 OR - Page orientation

Format: OR { PORTRAIT | LANDSCAPE }

Default: Portrait

Equiv. Qualifier: /ORIENTATION={ PORTRAIT | LANDSCAPE }

The OR command specifies the page orientation of the text. In portrait orientation, the text is set parallel to the short edge of the paper, and landscape orientation sets the text parallel to the long edge of the paper. The default orientation is portrait, except for paper type Ledger, which has a default orientation of landscape. Additionally, because the page orientation affects the length of the page, it also affects the number of lines per page which is determined by the vertical spacing. See the LP command above for more details on how the one affects the other.

9.3.35 PA - Paper size

Format: PA paper-keyword

Default: Letter

Equiv. Qualifier: /PAPER=paper-keyword

The PA command specifies the desired paper size for the print job. See Table 3-2: on page 63 for a list of the supported paper sizes.

9.3.36 PR - Job priority

Format: PR priority-value

Default: 100

Equiv. Qualifier: /PRIORITY=priority-value

This command determines the priority (from 0 to 255) of the print job within the print queue.

9.3.37 QU - Queue name

Format: QU queue-name

Default: SSV$PRINT

Equiv. Qualifier: /QUEUE=queue-name

The QU command selects the desired output queue. The queue specified must be either a generic print queue or a ScriptServer print queue. If the default queue, SSV$PRINT does not exist, then the symbiont will use SYS$PRINT as it’s default.

9.3.38 RM - Right margin

Format: RM size-expression

Default: .25 inch

Equiv. Qualifier: /RIGHT_MARGIN=size-expression

The RM command determines the location of the right margin in terms of the distance from the right edge of the paper. The right margin is used to determine the locations of centered and right-justified headings or page numbering text, and the point at which line-wrapping occurs, if wrap is enabled.

9.3.39 RS - Retain SPD

Format: RS { YES | NO }

Default: No

Equiv. Qualifier: /[NO]RETAIN

The RS command determines whether the SPD file created by SCRIPT is to be retained at the conclusion of the print job. If retained, the SPD may be used to requeue a print job without re-invoking the SCRIPT facility.

9.3.40 S2 - Secondary font scale

Format: S2 size-expression

Default: 10 points

Equiv. Qualifier: /S_SCALE=size-expression

The S2 command determines the size of the text used for the secondary font (established with the F2 command). The secondary font is used for setting heading and page numbering text.

9.3.41 SC - Primary font scale

Format: SC size-expression

Default: 10 points

Equiv. Qualifier: /DYNAMIC=SCALING=n

The scale of the primary font is determined by the value of the vertical spacing command below. If dynamic scaling is enabled, the primary font scale is a calculated number based on an 83% ratio to the vertical spacing between the lines of text on the page. For example: If you enter a VS value of 14, the SC value becomes 11.6 points, 83% of 14.

If however, you have a particular size you want your font to be, you can enter an SC value manually. Because of the relationship between it and the VS command, vertical spacing will change automatically if dynamic leading is enabled (for a detailed description of this command see VS below).

9.3.42 SL - Show library settings

Format: SL

Default: not applicable

Equiv. Qualifier: none

This command is used to display all currently stored settings in the current settings library along with the library's name.

9.3.43 SM - Setup modules

Format: SM module-list

Default: none

Equiv. Qualifier: /SETUP=module-list

The SM command specifies the name(s) of setup modules to be sent to the printer from the device control library before printing the file. Setup modules are discussed in detail in Chapter 4 of this document.

9.3.44 SP - Starting page

Format: SP page-number

Default: Page 1

Equiv. Qualifier: /START_PAGE=page-number

The SP command is used to specify the first page of the file to be printed. This command may be used optionally with the EP (ending page) command to print a subset of pages from the input file.

9.3.45 SPAWN - Spawn subprocess

Format: SPAWN

Default: not applicable

Equiv. Qualifier: none

The SPAWN command is used to create a subprocess from which standard DCL commands may be entered. Use this command when you want to issue a DCL command (such as the DIRECTORY command) without losing the currently-displayed settings.

9.3.46 SS - Save settings in library

Format: SS setting-name

Default: none

Equiv. Qualifier: none

The SS command saves the currently displayed settings in the settings library for future retrieval (with the LS command) under the name setting-name.

9.3.47 TM - Top margin

Format: TM size-expression

Default: .5 inch

Equiv. Qualifier: /TOP_MARGIN=size-expression

The TM command determines the location of the top margin, where the first line of text on each page is printed. Additionally, because the top margin affects the length of the page, it will affect the number of lines per page which is determined by the vertical spacing (VS command) below. See the LP command above for a more detailed description of the relationship between the two commands.

9.3.48 TP - Trailer page

Format: TP { YES | NO }

Default: No

Equiv. Qualifier: /[NO]TRAILER

The TP command determines whether a trailer page is to be printed at the conclusion of the print job.

9.3.49 TU - Tumble edge

Format: TU { LONG | SHORT }

Default: Long

Equiv. Qualifier: /TUMBLE_EDGE

The TU command controls image orientation of the second side when duplexing (printing on both sides on the paper). Tumble edge specifies whether the image on the second side of the page should be rotated 180° for binding on the short edge of the page.

9.3.50 UP - Layup pages

Format: UP n

Default: 1

Equiv. Qualifier: /[NO]LAYUP_PAGES

The UP command determines how many logical pages are printed on each physical page. If this command is used without an argument a pop-up menu will appear from which to choose a number of logical pages to print on each physical page. Only certain numbers appear in this menu: 1, 2, 4, 6, 8, 9, 16, 25, 36, 49, 64, 81, and 100. This is because the SCRIPT Facility will only format pages as correspond to one of these numbers. If a number is entered that does not conform to one of these numbers, that number will automatically be rounded to the next higher acceptable number. If, however, a number greater than 100 is entered that number will be rounded down to 100. The physical page on which the file will be printed is then formatted according to that new number. Layup pages is convenient for condensing large reports, or for proofing documents.

9.3.51 VS - Vertical line spacing

Format: VS size-expression

Default: 12 points

Equiv. Qualifier: /DYNAMIC=LEADING=n

The size of the vertical spacing is determined by the scale of the primary font command above. If dynamic leading is enabled, the vertical spacing is a calculated number based on a 120% ratio to the scale of the primary font. For example: If you enter an SC value of 20, the VS value becomes 24 points, 120% of 20.

If, however, you have a particular size you want your vertical spacing to be, you can enter a VS value manually. Because of the relationship between it and the SC command, the scale of the primary font will change automatically if dynamic scaling is enabled (for a detailed description of this command see SC above).

The vertical spacing also has a direct affect on the lines per page command above. Because the number of lines per page is determined by the distance between those lines, if that value changes so must the number of lines per page. For a description of the relationship between these two commands, see the LP command above.

9.3.52 WL - Wrap lines

Format: WL { ALL | LONG | NONE }

Default: None

Equiv. Qualifier: /WRAP

This command controls the action taken when lines exceed the right margin. WRAP=LONG causes long lines to be wrapped to the next line at the nearest word break. WRAP=ALL performs a fill-like procedure where all lines in a paragraph are broken at the points that come closest to the right margin without exceeding it. When full justification is enabled (using the JU command), wrap is set to ALL.

9.4 QUALIFIER DESCRIPTIONS

The SCRIPT facility's qualifiers are designed to behave like those for the PRINT command. The SCRIPT facility qualifiers are summarized in the Table 9-3: on page 166 and are described in detail below.

Table 9-3:Script Qualifier Summary

Qualifier	Description
AFTER	print after a certain time
BAR	embedded PostScript commands
BOTTOM_MARGIN	bottom margin
CHARACTER_SET	extended ASCII character set
COLLATED	collation of copies
COLOR	color printer requirements
COPIES	number of copies
DELETE	delete print file(s)
DUPLEX	duplexing mode
DYNAMIC	controls auto-leading and auto-scaling
EMBEDDED_COMMANDS	embedded formatting commands
END_PAGE	last page to print
FLAG	flag page
FONT	primary font
FORM	form name
GO	bypass menu
GENERIC	include/exclude generic queues
HEADING	specify heading text
HLOCATION	heading location
JUSTIFY	full justification
LAY_UP_PAGES	prints n logical pages per physical page
LEFT_MARGIN	left margin
LIBRARY	settings library name
LIGATURES	convert fi and fl to ligatures

Table 9-3:Script Qualifier Summary (Continued)

Qualifier	Description
LINES_PER_PAGE	controls number of lines per page
MESSAGES	controls generation of print log
MONOSPACE	converts proportionally spaced fonts to monospacing
NAME	specify job name
NLOCATION	numbering location
NOTE	operator note
NOTIFY	completion notification
NUMBERING	page numbering
ORIENTATION	page orientation
OUTPUT SPD	file name
PAPER	paper size
PREFIX	numbering prefix
PRIORITY	job queue priority
QUEUE	queue name
RETAIN	retain SPD file
RIGHT_MARGIN	right margin
SCALE	Primary Font Scale
SETTING	library setting-name
SETUP	setup modules
START_PAGE	first page to print
S_FONT	secondary font
S_SCALE	secondary font size
SUPPRESS	allows suppression of queue lookup command
TABS	emulate tab stops
TAB_SIZE	tab stop size
TOP_MARGIN	top margin
TRAILER	trailer page

Table 9-3:Script Qualifier Summary (Continued)

Qualifier	Description
TUMBLE_EDGE	specify tumble mode
TWO_UP	two-page layup
VERTICAL_SPACING	line spacing
WRAP	line wrapping

9.4.1 /AFTER

Format: /[NO]AFTER= time

Default: /NOAFTER

This qualifier holds a job until the specified time which can be specified as an absolute time, delta time or a combination thereof. If the specified time has passed, the job is queued for printing immediately. See Section 9.3.1 *"AF - Print after*" on page 149 above for details on time expression.

9.4.2 /BAR

Syntax: /[NO]BAR

Default: /NOBAR

This qualifier is used to indicate whether embedded PostScript commands are contained in the print file. These commands are lines that start with the vertical bar (|) character.

9.4.3 /BOTTOM_MARGIN

Syntax: /BOTTOM_MARGIN=size-expression

Default: /BOTTOM_MARGIN=.5in

Specifies the distance from the bottom edge of the paper to the last line to be printed on the page. Additionally, because the bottom margin affects the length of the page, it also affects the number of lines per page. Thus if you adjust the bottom margin, you need also pay attention to the lines per page. The qualifier entered last will be the last to take affect.

9.4.4 /CHARACTER_SET

Syntax: /CHARACTER_SET={ DEC | PC | CST}
Default: /CHARACTER_SET=DEC

This qualifier controls the ASCII extended character set that is to be used. CHARACTER_SET=DEC causes the DEC Multinational Character Set (DMCS) to be used. CHARACTER_SET=PC allows the use of the IBM PC-compatible character set. CHARACTER_SET=CST utilizes a custom character set. Refer to Section 12.9 *"Additional Options for PostScript Encoding*" on page 212 for instructions on customizing your own character set.

9.4.5 /COLLATED

Syntax: /[NO]COLLATED
Default: /COLLATED

The /COLLATED qualifier is used in conjunction with the /COPIES qualifier. It determines whether the copies generated are grouped by page. For example, the command:

```
$ SCRIPT/COPIES=3/NOCOLLATED
```

would generate three copies of the first page followed by three copies of the second page, and so on through the end of the file. Printing uncollated copies is faster than printing collated copies because the file is sent to the printer (and imaged) only once.

9.4.6 /COLOR

Syntax: /[NO]COLOR
Default: /NOCOLOR

Use this command qualifier to explicitly route this print job to or away from a color PostScript printer. The primary use of this qualifier is to avoid printing text-oriented jobs on the higher cost-per-page color printers. For compatibility with previous releases of ScriptServer, the default is no color preference.

9.4.7 /COPIES

Syntax: /COPIES=n
Default: /COPIES=1

This qualifier specifies the number of copies of the file to be printed, which may be any value from 1 through 255. The /COPIES qualifier works in conjunction with the /COLLATED qualifier for determining collation of the output. See Section 9.4.5 *"/COLLATED*” on page 169 for more information.

9.4.8 /DELETE

Syntax: /DELETE
Default: /NODELETE

Specifies that the input file is to be deleted after printing.

9.4.9 /DUPLEX

Syntax: /[NO]DUPLEX
Default: /NODUPLEX

The DUPLEX qualifier controls whether the output file is to be printed on both sides of the paper, which requires a printer with duplexing capability.

9.4.10 /DYNAMIC

Syntax: /DYNAMIC=([NO]LEADING=ratio, [NO]SCALING=ratio)
Default: /DYNAMIC=(LEADING=120, SCALING=83)

Dynamic leading automatically determines the vertical line spacing in relation to the scale of the primary font. The default of 120 indicates that the line spacing is 120% of the primary font's size. For example, the default scale of the primary font is 10 points, thus the default dynamic leading ratio of 120% yields a vertical line spacing of 12 points. With dynamic leading enabled, the vertical spacing is adjusted whenever the primary font's scale is changed. Additionally, the leading of the page determines the number of lines per page. Thus, as the leading changes, so must the number of lines per page. The converse is true when LINES_PER_PAGE is specified (see the discussion of the LINES_PER_PAGE qualifier below for a more detailed account of the relationship between these two qualifiers) With DYNAMIC= NOLEADING, the vertical spacing becomes a constant of 12 points.

NOTE: *If you use both the LINES_PER_PAGE and the DYNAMIC=(LEADING=n) qualifiers at the same time, the LINES_PER_PAGE qualifier will always override the leading in order to accommodate the request, with the leading reverting to the default of 120%.*

Dynamic scaling automatically determines the size of the primary font in relation to the document's vertical line spacing. The default value of 83 indicates that the size of the primary font is 83% of the distance of the vertical spacing. For example, the default leading of the page is 12 point vertical spacing. 83% of that 12 point spacing results in a 10 point font. With dynamic scaling enabled, the font size is adjusted whenever the leading changes. With DYNAMIC=NOSCALING, the font size becomes a constant of 10 points.

NOTE: *From the command line level, you can only use one or the other of these qualifiers. If you attempt to use both DYNAMIC=SCALING=n and DYNAMIC=LEADING=n, the last entered in the line will override the first, forcing the first to revert to its default.*

9.4.11 /EMBEDDED_COMMANDS

Syntax: /[NO]EMBEDDED_COMMANDS
Default: /NOEMBEDDED_COMMANDS

This qualifier indicates whether the print file contains any embedded commands. Embedded commands are lines that start with the backslash (\) character. Supported embedded commands are:

\CENTER	centers the next line between margins
\FONT	modifies the current font
\LEFT_MARGIN	modifies the left margin setting
\RIGHT_MARGIN	modifies the right margin setting
\TOP_MARGIN	modifies the top margin setting
\BOTTOM_MARGIN	modifies the bottom margin setting
\RESTART_PAGE	moves current position to top of page
\VERTICAL_SPACING	modifies the vertical line spacing
\[NO]TABS	modifies the tab emulation setting

\[NO]JUSTIFY	changes current justification setting
\WRAP=LONG	wraps lines that exceed the right margin
\WRAP=ALL	wraps short lines up and long lines down
\WRAP=NONE	disables line wrapping

9.4.12 /END_PAGE

Syntax: /END_PAGE=page-number
Default: /END_PAGE=last-page

This qualifier allows you to specify a page other than the last page of the file as the last page of the print job. This qualifier may be used in conjunction with the /START_PAGE qualifier to set upper and lower limits for pages to be printed.

9.4.13 /FLAG

Syntax: /FLAG
Default: /NOFLAG

Specifies that a flag page is to be printed for the job.

9.4.14 /FONT

Syntax: /FONT=font-name
Default: /FONT=Courier

Identifies the primary font for the output file. The font name specified must be a valid PostScript font that is resident or that has been downloaded to the printer. If a font name is abbreviated, SCRIPT selects the first match for the font entered; there is no test for ambiguity.

9.4.15 /FORM

Syntax: /FORM=form-name
Default: /FORM=DEFAULT

Specifies the OpenVMS form name for the job.

9.4.16 /GENERIC

Syntax: /[NO]GENERIC

Default: /NOGENERIC

This qualifier is used to explicitly request whether generic queues are to be included in the print queue pop-up menu.

9.4.17 /GO

Syntax: /[NO]GO

Default: /NOGO

Directs the SCRIPT facility to bypass the menu and submit the print job with the parameters selected by the command qualifiers.

9.4.18 /HEADING

Syntax: /HEADING=heading-text

Default: /NOHEADING

Specifies heading text to be placed at the top of each page. Headings can be placed at the center, left, right, inside, or outside of each page as determined by the / HLOCATION qualifier, whose default value is CENTER.

9.4.19 /HLOCATION

Syntax: /HLOCATION=header-location

Default: /HLOCATION=CENTER

Specifies the location of the heading text. Valid header-locations are CENTER, LEFT, RIGHT, INSIDE, and OUTSIDE.

9.4.20 /JUSTIFY

Syntax: /[NO]JUSTIFY

Default: /NOJUSTIFY

This qualifier is used to invoke full text justification. Justification automatically enables WRAP=ALL.

9.4.21 /LAYUP_PAGES

Syntax: /[NO]LAYUP_PAGES=n
Default: /[NO]LAYUP_PAGES

This qualifier determines how many logical pages are printed on each physical page. Any number may be entered, but if a number is entered as the argument which is not one of the following: 1, 2, 4, 6, 8, 9, 16, 25, 36, 49, 64, 81, and 100, that number will automatically be rounded to the next higher acceptable number. If however, that number is greater than 100, it will be rounded down to 100. The physical page is then formatted by that number and the file will be printed accordingly. Layup-pages are convenient for condensing large reports, or for proofing documents.

9.4.22 /LEFT_MARGIN

Syntax: /LEFT_MARGIN=size-expression
Default: /LEFT_MARGIN=.25in

Specifies the distance from the left edge of the paper to the first print position on each line.

9.4.23 /LIBRARY

Syntax: /LIBRARY=file-spec
Default: /LIBRARY=SYS$LOGIN:SPD_LIBRARY.DAT

Specifies the settings library filename. The SS (show settings), SL (show library), DS (delete settings), and LS (load settings) commands all reference this file.

9.4.24 /LIGATURES

Syntax: /[NO]LIGATURES
Default: /NOLIGATURES

This qualifier is used to specify whether the “fi” and “fl” character combinations are converted to their ligature counterparts.

9.4.25 /LINES_PER_PAGE

Syntax: LINES_PER_PAGE=line-count

Default: none

The number of lines per page depends on the vertical spacing between the lines of text on the page. When you change the number of lines per page, the value of the leading is overridden and changed to accommodate the new number of lines per page. If you have dynamic scaling enabled, the size of the primary font adjusts itself according to the new vertical spacing. (For details on how these numbers are determined, see the description of the DYNAMIC qualifier above) When you change the vertical spacing, in order to accommodate the spacing between the lines, the number of lines per page changes as well.

NOTE: *If you use both the LINES_PER_PAGE and the DYNAMIC=(LEADING=n) qualifiers at the same time, the LINES_PER_PAGE qualifier will always override the leading in order to accommodate the request, with the leading reverting to the default of 120%.*

9.4.26 /MESSAGES

Syntax: /[NO]MESSAGES

Default: /MESSAGES

This qualifier determines whether a Print Log should be generated by ScriptServer if job-related messages are received from the printer. See Section 3.4 *"Error Messaging*” on page 57 for more information about the ScriptServer Print Log.

9.4.27 /MONOSPACE

Syntax: /MONSPACE={0-999}

Default: /MONOSPACE=0 (no monospacing)

The MONOSPACE qualifier converts proportionally spaced fonts to monospace, with the character width corresponding to the number used as the argument to the command. This value represents the new width for each character in the font expressed in the character coordinate system, and is in the range 0-999. Typical values are 600 and 700.

It should be noted that characters in monospaced fonts are carefully and intentionally designed to look correct when evenly spaced, proportionally spaced fonts are not. Generally these converted fonts will look awkward when printed as monospaced, however this can be useful when printing tabular reports.

9.4.28 /NAME

Syntax: /NAME=job-name
Default: taken from file name

Specifies the name that the print job is to be submitted under.

9.4.29 /NLOCATION

Syntax: /NLOCATION=number-location
Default: /NLOCATION=CENTER

Specifies the location of the page numbering text. Valid number locations are CENTER, LEFT, RIGHT, INSIDE, and OUTSIDE. Numbering must first be turned on with the /NUMBER qualifier.

9.4.30 /NOTE

Syntax: /NOTE=text-string
Default: /NOTE=" "

This qualifier specifies the operator note to be displayed on the flag page under the heading NOTE. It is meaningless if no flag page is being printed.

9.4.31 /NOTIFY

Syntax: /[NO]NOTIFY
Default: /NONOTIFY

Specifies that the print job is to be submitted with the /NOTIFY qualifier, causing a message to be sent to users logged into the account submitting the job on completion of the job.

9.4.32 /NUMBERING

Syntax: /[NO]NUMBERING
Default: /NONUMBERING

Specifies that the page number is to be placed at the bottom of each page. Page numbers can be prefixed with a constant string of characters, using the /PREFIX qualifier. Page numbers can be placed at the center, left, right, inside, or outside of each page as determined by the /NLOCATION qualifier, whose default value is CENTER.

9.4.33 /ORIENTATION

Syntax: /ORIENTATION={ PORTRAIT | LANDSCAPE }
Default: /ORIENTATION=PORTRAIT

Selects the orientation of the printed page. In the default orientation of PORTRAIT, lines are printed along the width of the page. In LANDSCAPE, lines are printed along the length of the page. Note that “Ledger” paper is the only paper type that has a default orientation of LANDSCAPE. This qualifier has an immediate effect on the LINES_PER_PAGE qualifier above. If the argument to this qualifier is changed subsequently to entering a value for the lines per page, that number will be overridden as result of the values calculated for scaling and leading as discussed above in the section detailing the DYNAMIC qualifier. See Section 9.4.25 *"/LINES_PER_PAGE*” on page 175 above describing the LINES_PER_PAGE qualifier for more details.

9.4.34 /OUTPUT

Syntax: /OUTPUT=file-spec
Default: /OUTPUT=filename.type$SPD

Specifies the name of the ScriptServer Page Description (SPD) file that is submitted to the ScriptServer print symbiont. By default, this file has the same name as the input file with “$SPD” appended to the end. For example, the SPD file for LETTER.LIS would be LETTER.LIS$SPD. By default, SPD files are created in the directory located by the logical name SYS$SCRATCH.

9.4.35 /PAPER

Syntax: /PAPER=paper-type
Default: /PAPER=LETTER

Specifies the type of paper the job is to be printed on. See Table 3-2: on page 63 for a list of supported paper sizes.

9.4.36 /PREFIX

Syntax: /PREFIX=prefix-text
Default: /PREFIX=" "

Specifies a string to be prefixed onto the page number. Page numbering must have been turned on with the /NUMBERING qualifier. See the documentation for the qualifier /NLOCATION.

9.4.37 /PRIORITY

Syntax: /PRIORITY=queue-priority
Default: /PRIORITY=100

Specifies the queue priority at which the print job is to be submitted. Values for priorities range from 0 to 255.

9.4.38 /QUEUE

Syntax: /QUEUE=queue-name
Default: /QUEUE=SSV$PRINT

Directs SCRIPT to submit the print job to the specified queue. If the queue, SSV$PRINT, does not exist, symbiont will use SYS$PRINT as the default.

9.4.39 /RETAIN

Syntax: /[NO]RETAIN
Default: /NORETAIN

Specifies that the SPD file created by SCRIPT is not to be deleted at the conclusion of the print job. See the documentation for the qualifier /OUTPUT.

9.4.40 /RIGHT_MARGIN

Syntax: /RIGHT_MARGIN=size-expression
Default: /RIGHT_MARGIN=.25in

Specifies the distance from the right edge of the paper to the last print position on each line. Size expressions maybe in terms of points, (pt), inches (in), or centimeters (cm).

9.4.41 /SCALE

Syntax: /SCALE=size-expression
Default: /SCALE=10pt

Determines the size of characters to be used for the default font. The default value for SCALE is 10 points.

9.4.42 /SETTING

Syntax: /[NO]SETTING=setting-name
Default: /NOSETTING

Loads the predefined setting-name from the settings library.

9.4.43 /SETUP

Syntax: /[NO]SETUP=module-name[,...]
Default: /NOSETUP

This qualifier is used to direct the ScriptServer print symbiont to copy the specified module(s) from the device control library to the printer before the input file(s) are to be printed.

9.4.44 /START_PAGE

Syntax: /START_PAGE=page-number
Default: /START_PAGE=1

The qualifier allows you to specify a page other than the first page of the file as the first page of the print job. This qualifier may be used in conjunction with the /END_PAGE qualifier to set upper and lower limits for pages to be printed.

9.4.45 /SUPPRESS

Syntax: /[NO]SUPPRESS
Default: [NO]SUPPRESS

When enabled the suppression qualifier overrides the queue lookup pop-up menu of the SCRIPT Facility.

9.4.46 /S_FONT

Syntax: /S_FONT=font-name
Default: /S_FONT=Courier

Specifies the secondary font to be used in printing the input file. The secondary font and secondary scale are used for printing the page numbering text and page headings.

9.4.47 /S_SCALE

Syntax: /S_SCALE=size-expression
Default: /S_SCALE=10pt

Specifies the size of the secondary font to be used in printing the input file. The secondary font and secondary scale are used for printing the page numbering text and page headings.

9.4.48 /TABS

Syntax: /[NO]TABS
Default: /NOTABS

This qualifier controls how tab characters are handled. By default, tabs are converted to the correct number of spaces to advance to the next tab stop. In proportionally-spaced fonts, this scheme fails because all characters are different sizes. When printed with the tab emulation invoked by the /TAB qualifier, the current position is advanced to the next tab position as determined by the TAB_SIZE qualifier.

9.4.49 /TAB_SIZE

Syntax: /TAB_SIZE=size-expression
Default: /TAB_SIZE=.75in

Determines the spacing of tab stops when tab emulation is in effect. See the description for the TABS qualifier above.

9.4.50 /TOP_MARGIN

Syntax: /TOP_MARGIN=size-expression
Default: /TOP_MARGIN=.5in

Specifies the distance from the top edge of the paper to the baseline of the first line to be printed on the page. The default setting for the top margin is one-half inch. Additionally, because the top margin affects the length of the page, it also affects the number of lines per page. So if you adjust the top margin, you must pay attention to the number of lines per page. The qualifier entered last in sequence will be the last one to take affect.

9.4.51 /TRAILER

Syntax: /[NO]TRAILER
Default: /NOTRAILER

Specifies whether a trailer page is printed after the job.

9.4.52 /TUMBLE_EDGE

Syntax: /TUMBLE_EDGE={ LONG | SHORT }
Default: /TUMBLE_EDGE=LONG

Determines whether tumble mode is to be used when printing ASCII file in duplex mode. Tumble mode causes the image on the second side of the paper to be rotated 180 degrees. When TUMBLE_EDGE is LONG, the duplex job is printed for being bound on the long edge of the paper.

9.4.53 /VERTICAL_SPACING

Syntax: /VERTICAL_SPACING=size-expression
Default: /VERTICAL_SPACING=12pt

Determines the vertical spacing of lines (baseline-to-baseline) for the print job.

9.4.54 /WRAP

Syntax: /WRAP={ ALL | LONG | NONE }
Default: /WRAP=NONE

Controls the action taken when lines exceed the right margin. WRAP=LONG causes long lines to be wrapped to the next line at the nearest word break. WRAP=ALL performs a fill-like procedure where all lines in the paragraphs are broken at the points that come closest to the right margin without exceeding it. When right-justification is enabled, WRAP is set to ALL.

Chapter 10

PRINT Command and ScriptServer

10.1 ASCII FILES AND THE PRINT COMMAND

ASCII files submitted for printing with the PRINT command will be translated to PostScript for printing when possible (i.e. when submitted to a PostScript / PCL or a PostScript only printer). However, when PostScript is not available on the output device, ScriptServer will submit the text file as PCL using the printer defaults established on the printer's control panel for all formatting and page description considerations.

This behavior can be altered through use of the logical:

```
SSV$TEXT_LANGUAGE
```

Defining this logical to PCL:

```
$ DEFINE/SYSTEM SSV$TEXT_LANGUAGE PCL
```

forces printing of text files in PCL mode instead of PostScript.

To define this logical on a queue specific basis, define the logical in the form:

```
$ DEFINE/SYSTEM SSV$queue_TEXT_LANGUAGE equivalence
```

where “queue” is the name of the queue for which you want to affect text translations, and “equivalence” is the language to use in the translation.

10.2 PRINT QUALIFIER DIFFERENCES

ScriptServer allows you to use the PRINT command just as you would to print on a conventional printer. Most of the command qualifiers behave without any change. Some, however, have a different effect when used with ScriptServer. The following PRINT qualifiers behave with some differences when used in conjunction with ScriptServer.

10.2.1 /CHARACTERISTICS Qualifier

The intended use of the CHARACTERISTICS qualifier is to provide job routing by the OpenVMS queue manager as described in Section 3.5. ScriptServer also uses the characteristics originating from the PRINT command (as opposed to using the SCRIPT facility) as font and paper specifiers. For example, the command:

```
$ PRINT/CHARACTERISTICS=(TIMES_ROMAN,LEGAL) file-spec
```

results not only in routing the print job to a PostScript printer with the Times Roman font and Legal sized paper, it also invokes the desired font and (in multiple input tray printers) paper on the printer.

10.2.2 /COPIES Qualifier

The COPIES qualifier is complemented by the print parameter SHEET_COUNT, allowing you to print multiple uncollated copies of each sheet of paper. If uncollated output is acceptable for a given print job, use PRINT/PARAM=SHEET_COUNT=n instead of PRINT /COPIES=n, because uncollated copies are more efficient in transmission and imaging time.

10.2.3 /FEED Qualifier

The FEED qualifier is used to automatically insert form feeds at the end of a page, based on a bottom margin which is specified in terms of lines from the bottom of the page. When ScriptServer is processing PRINT command requests, the /FEED qualifier is used to tell the ScriptServer print symbiont to honor the bottom margin. For files submitted with /NOFEED, ScriptServer starts a new page when an attempt is made to print below the bottom edge of the paper.

10.2.4 /HEADING Qualifier

This qualifier is not supported when printing in PCL mode. In PostScript mode, however, it should function properly.

10.2.5 /RESTART Qualifier

The RESTART qualifier is used to specify that an interrupted print job is to restart at the point that it was interrupted. As this is not possible in the PostScript environment, RESTART starts interrupted jobs from the beginning.

10.2.6 /SETUP Qualifier

The SETUP qualifier is used to have device setup modules from the device control library sent to the printer before the print job. With minor exceptions, all setup modules used within ScriptServer must be written in PostScript for PostScript print jobs and PCL for PCL jobs, as no translation is performed on them.

10.2.7 /SPACE Qualifier

This qualifier is not supported when printing in PCL mode. In PostScript mode, however, it should function properly.

10.3 PRINT/PARAMETERS PARAMETERS

The ScriptServer Printing System supports several parameters that are compatible with Digital Equipment PrintServer series printers and the corresponding support software. It also supports several parameters that are specific to the ScriptServer software.

Parameters are passed to ScriptServer using the following form of the PRINT command:

```
$ PRINT/PARAMETERS=(parameter[,...]) file-spec[,...]
```

The number of parameters in the parameter list is limited to eight. If only one parameter is specified, the parentheses may be omitted. Some parameters accept more than one argument separated by a comma and enclosed in parentheses. If you are using a parameter in this format with other parameters, enclose the multiple-argument parameter in double quotes. For example:

```
$ PRINT/PARAM=("PAGE_LIMIT=(5,18)",JUSTIFY) file-spec
```

The parameters that you specify are not checked by DCL when you enter the PRINT command, so no error messages will be displayed should a spelling or typing error have occurred. The parameters supported by ScriptServer are summarized in Table 10-1: on page 186, and described in detail below. Unless noted with an asterisk (*), the following print parameters are either applicable only to ASCII files, or the data-type of the file is irrelevant to the parameter's use. Those parameters noted with an asterisk (*) function with both ASCII and PostScript files. Those noted with a dagger (†) function with PCL files, as well.

Table 10-1:PRINT Parameter Summary

Parameter	Description
AUTO_LEADING	specifies the ratio of vertical line spacing to the primary font scale
AUTO_SCALING	specifies the ratio of the primary font scale to the vertical line spacing
BAR_COMMANDS	indicates that the print file contains embedded PostScript commands
BOTTOM_MARGIN	determines distance from bottom edge of page
DATA_TYPE	specifies the type of data in the print file
DISK_DELETE	deletes file(s) from printer’s hard disk
DISK_FILE	download file(s) to printer’s hard disk
DISK_FONT	download font(s) to printer’s hard disk
DISK_RUN	execute PostScript file on printer’s hard disk
† * DUPLEX	generate two-sided output
EMBEDDED_COMMANDS	specifies that the print file contains embedded control commands
EMULATE_TABS	use special tab emulation
ENCODING	controls extended character set used
FONT	specifies primary font
JUSTIFY	use full justification
LEFT_MARGIN	determines distance from left edge of page
LINES_PER_PAGE	controls number of lines per page
MESSAGES	controls generation Print Log if messages are received from the printer
MONOSPACE	converts proportionally spaced fonts to mono-spacing
NUMBER_UP	print pages n-up
† PAGE_LIMIT	print a subset of pages

Table 10-1:PRINT Parameter Summary (Continued)

Parameter	Description
† PAGE_ORIENTATION	portrait or landscape orientation
† PAPER	specifies paper size
PRELOAD	download font(s) for job
RIGHT_MARGIN	determines distance from right edge of page
SCALE	controls primary font scale
SCRIPT_SETTINGS	use a pre-defined SCRIPT setting
† * SHEET_COUNT	requests uncollated copies
† SHEET_SIZE	specifies paper size
† * SIDES	generate two-sided output
TAB_SIZE	control tab size for tab emulation
TOP_MARGINE	determines distance from top edge of page
† * TUMBLE_EDGE	control of duplexing tumble mode
VERTICAL_SPACING	distance between lines
WRAP	controls line wrapping

NOTE: *When using those parameters marked with an asterisk with PostScript files, ScriptServer must modify PostScript code sent to the printer. If you do not want ScriptServer to alter the PostScript code, do not use these parameters with those files.*

Establishing Defaults with Parameters

You can establish defaults both system-wide and for specific queues by defining a logical name that equates to a list of parameters. To create a system-wide definition, use the format:

```
$ DEFINE/SYS SSV$SYSTEM_PARAMETERS “parameter-list”
```

For example, to establish Palatino Roman as the default font with a default scale of 12 points for ASCII translations system-wide, you could define the logical as follows:

```
$ DEFINE/SYS SSV$SYSTEM_PARAMETERS (FONT=”Palatino-Roman”,SCALE=12PT)”
```

NOTE: *Fonts are case-sensitive and must be enclosed in quotation marks.*

To set up a default that is queue-specific, define the logical name in the format:

```
        $ DEFINE/SYS SSV$queue_PARAMETERS “parameter-list”
```

10.3.1 AUTO_LEADING Parameter

Syntax: [NO]AUTO_LEADING=ratio

Default: NOAUTO_LEADING

This parameter is used to specify a ratio between the vertical line spacing and the size of the primary font. When auto-leading is enabled, the default ratio is 120%, which means that the line spacing is set to 120% of the size of the font.

10.3.2 AUTO_SCALING Parameter

Syntax: [NO]AUTO_SCALING=ratio

Default: NOAUTO_SCALING

This parameter is used to specify a ratio between the size of the primary font and the vertical line spacing. When auto-scaling is enabled, the default ratio is 83%, which means that the size of the font is set to 83% of the line spacing.

10.3.3 BAR_COMMANDS Parameter

Syntax: [NO]BAR_COMMANDS

Default: NOBAR_COMMANDS

This parameter is used to indicate whether embedded PostScript commands are contained in the print file. These commands are lines that start with the vertical bar (|) character. See Appendix D for more information about embedded PostScript commands.

10.3.4 BOTTOM_MARGIN Parameter

Syntax: BOTTOM_MARGIN=size-expression

Default: BOTTOM_MARGIN=.5in

This parameter determines the lowest position on the page on which text may fall. Any attempt to place text from the input file below the bottom margin results in starting a new page. Page numbering text (if requested) is the only text that is printed below the bottom margin. Additionally, because the bottom margin affects the length of the page, it also affects the number of lines per page which is determined by the vertical spacing.

10.3.5 DATA_TYPE Parameter

Syntax: DATA_TYPE={ ASCII | POSTSCRIPT | HPGL | PCL | AUTOMATIC | site-defined}

Default: DATA_TYPE=ASCII

This parameter determines whether or not the input file is translated from ASCII to PostScript. Data types of ASCII and ANSI are translated to PostScript. If the data type is POSTSCRIPT, the file is sent to the printer untranslated. The HPGL data type invokes the HP-GL to PostScript translator in ScriptServer. A PCL data type will send the PCL file to the printer untouched. Depending on the capabilities of printers at your site, there may also be site-defined data types that invoke printer-based emulation modes. Consult your installation manager for the names of these data types. If the DATA_TYPE parameter is not specified in a PRINT command, ScriptServer determines the data type based on the file itself. For more information, see Section 3.2.1 *"Using the PRINT Command*" on page 55.

10.3.6 DISK_DELETE Parameter

Syntax: DISK_DELETE="PostScript-filespec"

Default: None

This parameter allows you to delete file(s) from the printer's hard disk. PostScript wildcard filespecs are supported. A dummy filespec may be required to satisfy the PRINT command, but will be processed normally by the symbiont. This command requires that the user is logged into an account with a UIC group of 1. Since PostScript file specifications are case-sensitive, the parameters must be enclosed in quotes to preserve lowercase characters.

10.3.7 DISK_FILE Parameter

Syntax: DISK_FILE="PostScript-filespec"

Default: None

This parameter allows you to load a (PostScript) file directly onto a printer's hard disk. If the PostScript filespec is omitted, the symbiont will create the file with the same name (without device, directory, and version number) as the OpenVMS file. This command requires that the user be logged into an account with a UIC group of 1. Since PostScript file specifications are case-sensitive, the parameters must be enclosed in quotes to preserve lowercase characters.

10.3.8 DISK_FONT Parameter

Syntax: DISK_FONT="PostScript-filespec"

Default: None

This parameter allows you to load a font file permanently onto a printer's hard disk. If "=PostScript-filespec" is omitted, the symbiont will automatically place the file in the "fonts/" directory on the hard disk, parsing the font filename from the font file itself. This command requires that the user be logged into an account with a UIC group of 1. Since PostScript file specifications are case-sensitive, the parameters must be enclosed in quotes to preserve lowercase characters.

10.3.9 DISK_RUN Parameter

Syntax: DISK_RUN=PostScript-filespec

Default: None

This parameter allows you to run a PostScript file on the printer's hard disk. A dummy filespec may be required to satisfy the PRINT command, but will be processed normally. This command does not require that the user be logged into an account with a UIC group of 1. Since PostScript file specifications are case-sensitive, the parameters must be enclosed in quotes to preserve lowercase characters.

10.3.10 DUPLEX Parameter

Syntax: [NO]DUPLEX

Default: NODUPLEX

The DUPLEX parameter functions with PCL, ASCII and PostScript files. It controls whether the output file is to be printed on both sides of the paper, which requires a printer with duplexing capability. A related parameter is TUMBLE_EDGE, which controls the orientation of the image on the second side of a sheet of paper.

10.3.11 EMBEDDED_COMMANDS Parameter

Syntax: [NO]EMBEDDED_COMMANDS

Default: NOEMBEDDED_COMMANDS

This parameter indicates whether the print file contains any embedded commands. Embedded commands are lines that start with the backslash (\) character. The default setting is

NOEMBEDDED_COMMANDS. Embedded commands allow you to change the way a file is formatted during printing. Supported embedded commands are:

Table 10-2:Embedded Command Parameters

\CENTER	centers the next line between margins
\FONT	modifies the current font
\LEFT_MARGIN	modifies the left margin setting
\RIGHT_MARGIN	modifies the right margin setting
\TOP_MARGIN	modifies the top margin setting
\BOTTOM_MARGIN	modifies the bottom margin setting
\RESTART_PAGE	moves current position to top of page
\VERTICAL_SPACING	modifies the vertical line spacing
\[NO]TABS	modifies the tab emulation setting
\[NO]JUSTIFY	changes current justification setting
\WRAP=LONG	wraps lines that exceed the right margin
\WRAP=ALL	wraps short lines up and long lines down
\WRAP=NONE	disables line wrapping

Refer to Chapter 19 for more information on using embedded commands with ScriptServer.

10.3.12 EMULATE_TABS Parameter

Syntax: [NO]EMULATE_TABS
Default: NOEMULATE_TABS

This parameter controls how tab characters are handled. By default, tabs are converted to the correct number of spaces to advance to the next tab stop. In proportionally-spaced fonts, this scheme fails because all characters are different sizes. When printed with the tab emulation invoked by the parameter EMULATE_TABS, the current position is advanced to the next tab position as determined by the TAB_SIZE parameter.

10.3.13 ENCODING Parameter

Syntax: ENCODING={ DEC | PC | CST }

Default: ENCODING=DEC

This qualifier controls the ASCII extended character set that is to be used. CHARACTER_SET=DEC causes the DEC Multinational Character Set (DMCS) to be used. CHARACTER_SET=PC allows the use of the IBM PC-compatible character set (excluding the line-drawing characters). CHARACTER_SET=CST invokes the user-defined character set (see Section 12.9 "*Additional Options for PostScript Encoding*" on page 212 for a detailed discussion on defining your own character set).

10.3.14 FONT Parameter

Syntax: FONT=font-name

Default: FONT="Courier"

This qualifier controls the typeface that is to be used. The font name must be enclosed in quotes to preserve lowercase characters, and must be one of the fonts in Table 3-1: on page 59, or a site-defined font.

10.3.15 JUSTIFY Parameter

Syntax: [NO]JUSTIFY

Default: NOJUSTIFY

This parameter is used to invoke full text justification, for smooth left and right margins. Full justification automatically enables the parameter WRAP=ALL.

10.3.16 LEFT_MARGIN Parameter

Syntax: LEFT_MARGIN=size-expression

Default: LEFT_MARGIN=.25in

This parameter specifies the distance of the left margin from the left edge of the paper.

10.3.17 LINES_PER_PAGE Parameter

Syntax: LINES_PER_PAGE=line-count

Default: none

This parameter controls the number of lines to be placed on each page. Normally, the line count is determined by margins, font size, and vertical line spacing. This parameter, in conjunction with the AUTO_SCALING parameter, provides another technique for controlling line spacing.

10.3.18 MESSAGES Parameter

Syntax: [NO]MESSAGES

Default: MESSAGES

This parameter controls whether a Print Log is generated at the end of the job, containing the messages (if any) generated by the printer during the job.

10.3.19 MONOSPACE Parameter

Syntax: MONSPACE={0-999}

Default: MONOSPACE=0 (no monospacing)

The MONOSPACE parameter converts proportionally spaced fonts to monospace, with the character width corresponding to the number used as the argument to the command. This value represents the new width for each character in the font expressed in the character coordinate system, and is in the range 0-999. Typical values are 600 and 700.

It should be noted that characters in monospaced fonts are carefully and intentionally designed to look correct when evenly spaced, proportionally spaced fonts are not. Generally these converted fonts will look awkward when printed, however the spacing can be useful when printing tabular reports.

10.3.20 NUMBER_UP Parameter

Syntax: NUMBER_UP={ n}

Default: NUMBER_UP=1

The NUMBER_UP parameter is used to specify the number of pages to be printed on each sheet of paper. The parameter NUMBER_UP=6 indicates that six logical pages should be printed on each sheet. Values entered other than 1, 2, 4, 6, 8, 9, n 2 (to n=10) will be rounded to the next highest square, unless that number exceeds 100 in which case it will be rounded down to 100. For example:

```
$ PRINT/PARAM=(NUMBER_UP=17)
```

will format the page as if the value entered were 25.

10.3.21 PAGE_LIMIT Parameter

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

Default: PAGE_LIMIT=(first-page,last-page)

The PAGE_LIMIT parameter functions with both ASCII and PCL files. It is used to specify a subset of pages from the file to be printed, using the first and last page numbers. Either of the upper- or lower-limit values may be omitted.

For example, the command:

```
$ PRINT/PARAM=PAGE_LIMIT=5
```

prints the first five pages of the print file(s).

```
$ PRINT/PARAM="PAGE_LIMIT=(3,)"
```

prints starting at page three.

```
$ PRINT/PARAM="PAGE_LIMIT=(4,8)"
```

prints pages four through eight.

10.3.22 PAGE_NUMBERING Parameter

Syntax: [NO]PAGE_NUMBERING

Default: NOPAGE_NUMBERING

This parameter specifies that the page number will appear centered at the bottom of the page on ASCII files.

10.3.23 PAGE_ORIENTATION Parameter

Syntax: PAGE_ORIENTATION={ PORTRAIT | LANDSCAPE }

Default: PAGE_ORIENTATION=PORTRAIT

This parameter functions with both ASCII and PCL files. It specifies the orientation of the printed output on the (logical) page. The minimum abbreviation of this parameter is PAGE_O. In PORTRAIT orientation, the lines of text run parallel to the short edge of the page (not necessarily the short edge of the paper). In LANDSCAPE orientation, the lines of text run parallel to the long edge of the page.

10.3.24 PAPER Parameter

Syntax: PAPER=page-size

Default: PAPER=LETTER

This parameter functions with both ASCII and PCL files. It specifies the size of the paper desired for the print job. For a list of supported paper sizes, see Table 3-2: on page 63. This parameter is synonymous with the PAPER parameter discussed above.

NOTE: *Envelope sizes are not supported in PCL mode.*

10.3.25 PRELOAD Parameter

Syntax: PRELOAD=(font-name[,]font-name...])

Default: none

This parameter directs the ScriptServer print symbiont to download the specified font(s) before printing the file(s) specified in the print command. For a list of the fonts available for downloading at your site, use the command:

```
$ SHOW LOGICAL/TABLE=SSV$DOWNLOAD_TABLE SSV$FONTS
```

10.3.26 RIGHT_MARGIN Parameter

Syntax: RIGHT_MARGIN=size-expression

Default: RIGHT_MARGIN=.25in

This parameter determines the location of the right margin in terms of the distance from the right edge of the paper. The right margin is used to determine the locations of centered and right-justified headings or page numbering text, and the point at which line-wrapping occurs, if wrap is enabled.

10.3.27 SCALE Parameter

Syntax: SCALE=size=expression

Default: SCALE=12pt

This qualifier controls the size of the primary font. It may be specified in terms of points, centimeters, or inches.

10.3.28 SCRIPT_SETTINGS Parameter

Syntax: SCRIPT_SETTINGS=setting-name

Default: none

This parameter specifies a SCRIPT facility setting name to be loaded from the symbiont settings library. The settings stored in this settings library are controlled by the system manager. This parameter is not DEC-compatible, and is primarily intended to allow the creation of queue-specific and system-wide defaults for the ScriptServer print queues.

10.3.29 SHEET_COUNT Parameter

Syntax: SHEET_COUNT=copies

Default: SHEET_COUNT=1

The SHEET_COUNT parameter functions with ASCII, PCL and PostScript files. It specifies the number of uncollated copies of the input file(s) desired for the print job. Uncollated copies produces x copies of page 1, followed by x copies of page 2, and so on. Uncollated copies can be produced more quickly than the collated copies produced using the PRINT/COPIES qualifier. The minimum abbreviation of this parameter is SHEET_C.

10.3.30 SHEET_SIZE Parameter

Syntax: SHEET_SIZE=paper-type

Default: SHEET_SIZE=LETTER

This parameter functions with both ASCII and PCL files. It specifies the size of the paper desired for the print job. For a list of supported paper sizes, see Table 3-2: on page 63. The minimum abbreviation of this parameter is SHEET_S. This parameter is synonymous with the PAPER parameter discussed above.

NOTE: *Envelope sizes are not supported in PCL mode*

10.3.31 SIDES Parameter

Syntax: [NO]SIDES

Default: NOSIDES

The SIDES parameter functions exactly as does the DUPLEX parameter above with ASCII, PCL and PostScript files. It controls whether the output file is to be printed on both sides of the paper, which requires a printer with duplexing capability. A related parameter is TUMBLE_EDGE, which controls the orientation of the image on the second side of a sheet of paper.

10.3.32 TAB_SIZE Parameter

Syntax: TAB_SIZE=size-expression

Default: TAB_SIZE=.75in

Determines the spacing of tab stops when tab emulation is in effect. See the information on the EMULATE_TABS parameter above.

10.3.33 TOP_MARGIN Parameter

Syntax: TOP_MARGIN=size-expression

Default: TOP_MARGIN=.5in

This parameter determines the location of the top margin, where the first line of text on each page is printed. Additionally, because the top margin affects the length of the page, it will affect the number of lines per page which is determined by the vertical spacing parameter.

10.3.34 TUMBLE_EDGE Parameter

Syntax: TUMBLE_EDGE={ LONG | SHORT }

Default: TUMBLE_EDGE=LONG

The TUMBLE_EDGE parameter functions with ASCII, PCL and PostScript files. It determines whether tumble mode is to be used when printing ASCII files in duplex mode. Tumble mode causes the image on the second side of the paper to be rotated 180 degrees. When TUMBLE_EDGE is LONG, the duplex job is printed for being bound on the long edge of the paper. This parameter has no effect unless duplexing is enabled with the DUPLEX parameter.

10.3.35 VERTICAL_SPACING Parameter

Syntax: VERTICAL_SPACING=size-expression

Default: VERTICAL_SPACING=12pts

Determines the vertical spacing of lines (baseline-to-baseline) for the print job.

10.3.36 WRAP Parameter

Syntax: WRAP={ ALL | LONG | NONE }

Default: WRAP=NONE

Controls the action taken when lines exceed the right margin. WRAP=LONG causes long lines to be wrapped to the next line at the nearest word break. WRAP=ALL performs a fill-like procedure where all lines in the paragraphs are broken at the points that come closest to the right margin without exceeding it. When right-justification is enabled, WRAP is set to ALL.

Chapter 11

Printer Disk Management

The ScriptServer software provides several commands for PostScript printers that support hard disks. These commands are in the form of PRINT parameters, and allow the creation, deletion, and execution of files on the printer's hard disk.

NOTE: *ScriptServer does not support printer disk management for PCL5 or PCL/PJL-only printers.*

To use these disk management commands, some knowledge of PostScript file specifications is required. These specifications are considered implementation-dependent, so they may vary from model to model. However, the general format of a PostScript file specification is:

```
%device%directory/filename
```

NOTE: *PostScript file specifications, like the rest of the language, are case-sensitive. Since DCL will upcase all characters in commands that are not enclosed in quotation marks, you will need to place the PRINT parameters within them to preserve lowercase characters.*

Disk names are usually %disk% or %diskn%, where n is an integer representing the SCSI hardware address. If the disk name is omitted, the first disk device found is used.

Certain PostScript directories are reserved for use by the PostScript interpreter, and should not be manipulated by ScriptServer disk management commands. Examples of these are the Sys/ (system files) and FC/ (font cache) directories. Another directory of note is the font/ directory, which is used for storage of disk-resident font files.

The general format of ScriptServer commands used to manipulate disk-resident PostScript files is:

```
$ PRINT/PARAM=DISK_cmd=PS-filespec VMS-filespec
```

In certain cases, the OpenVMS filespec is not required to fulfill the disk operation, but is required by DCL for the PRINT command. In these cases, use a “dummy” file: a zero-length file with a .PS file extension. For example:

```
$ CREATE SCRIPTSERVER:DUMMY.PS
<ctrl/z>
```

NOTE: *To help preserve the integrity of the files on the printer's hard disk, all file manipulation commands except DISK_RUN require that the user be logged into an account with a UIC group of 1.*

11.1 DOWNLOADING FONTS TO DISK

Disk-resident fonts behave identically to those that reside in RAM and ROM on the printer. They are automatically found by the findfont operator. To download a font to the printer's hard disk, use the DISK_FONT parameter, for example:

```
$ PRINT/PARAM=DISK_FONT SSV$TYPEFACE_DIRECTORY:OPTIMA.PFB
```

This command will copy the font “Optima” from the ScriptServer font directory to the fonts/ directory on the printer. This command also parses the font name (if present) from the first record of the font file and renames the file accordingly.

11.2 DOWNLOADING FILES TO DISK

To download a file to the printer's hard disk, use a command of the format:

```
$ PRINT/PARAM="DISK_FILE=PS-filespec" VMS-filespec
```

PostScript files that have been copied to hard disk can be executed much faster using the DISK_RUN command (see below) than by retransmitting them to the printer. For example, suppose you want to download the printer status report that is distributed with ScriptServer to the hard disk for faster execution, and that you want to call the file PtrStat.ps on the printer. To download the file, use the command:

```
$ PRINT/QUEUE=queue-name/PARAM="DISK_FILE=PtrStat.ps" -
_$ SCRIPTSERVER:PRINTER_STATUS.PS
```

You may then execute the PtrStat.ps file without retransmission using the DISK_RUN command (see example below).

11.3 EXECUTING POSTSCRIPT FILES ON DISK

To execute a PostScript file on the printer's hard disk, use the DISK_RUN command. When the file to be executed is to be run without any subsequent file sent from the VAX, a dummy file must be used as described above.

For example, to execute the printer status report that was downloaded in the previous example, use the command:

```
$ PRINT/QUEUE=queue-name/PARAM="DISK_RUN=PtrStat.ps"-
_$ SCRIPTSERVER:DUMMY.PS
```

11.4 DELETING DISK-RESIDENT FILES

To delete a file from the printer's hard disk, use the command DISK_DELETE. For example, to delete the PtrStat.ps file that was used in the previous examples, issue the command:

```
$ PRINT/QUEUE=queue-name/PARAM="DISK_DELETE=PtrStat.ps"-
_$ SCRIPTSERVER:DUMMY.PS
```

11.5 OBTAINING A DIRECTORY OF THE DISK

A PostScript program that generates a directory listing of the printer's hard disk is distributed in the SCRIPTSERVER directory. To generate a listing, simply submit the file DISK_DIRECTORY.PS to the printer, for example:

```
$ PRINT/QUEUE=queue-name SCRIPTSERVER:DISK_DIRECTORY.PS
```

Chapter 12

Customizing ScriptServer

12.1 ESTABLISHING DEFAULTS

It is possible to establish defaults for ScriptServer print queues on a queue-specific, form-specific or system-wide basis. This is accomplished using a symbiont settings library which is maintained through the SCRIPT facility. The symbiont settings library is functionally the same as SCRIPT settings libraries, but is accessed directly by the symbiont at print time, rather than by the SCRIPT facility at print request time. The file SCRIPTSERVER:SPD$LIBRARY.DAT is the symbiont settings library.

NOTE: *The procedures described here are used for controlling the ASCII-to-PostScript translation settings on files submitted using the PRINT command, and have no effect on defaults in the SCRIPT Facility.*

After creating settings in this library (with the SPD_MGR command procedure described below), the settings can be associated with a particular form name, with all ScriptServer queues system-wide, or with a given queue. This is done by creating a logical name that equates to the new default setting-name.

System Wide Defaults

To have a setting-name in the symbiont library apply as the default for the entire system, define the logical as:

```
$ DEFINE/SYS SSV$DEFAULT_SETTINGS setting-name
```

where “setting-name” is the name of the SCRIPT setting in the symbiont settings library that you wish to use as the system-wide default.

Queue-Specific Defaults

To have a setting-name in the symbiont library apply as the default for a specific queue, the logical name must be defined as follows:

```
$ DEFINE/SYS SSV$queue-name_SETTINGS setting-name
```

where “queue-name” is the name of the queue to which the settings apply, and “setting-name” is the name of the settings as saved in the symbiont settings library.

For example, suppose you have created a setting in the symbiont library named NEW-SETTING, and you want it to apply to a print queue named LASER. The DCL command to set up the correct logical name is:

```
$ DEFINE/SYSTEM SSV$LASER_SETTINGS NEW-SETTING
```

ScriptServer supports the use of the queue settings for generic queues. For example, if you have a generic queue named GENQUEUE feeding two ScriptServer queues named PSQUEUE1 and PSQUEUE2, the symbiont will first look for a default settings definition for PSQUEUE, and if none is found, it will check for a definition for GENQUEUE.

Form-Specific Defaults

To have a setting-name in the symbiont library apply as the default for a specific form name, you must first define a form to which you will apply the settings. Once that has been done, the logical name must be defined as follows:

```
$ DEFINE/SYS SSV$FORM_form-name_SETTINGS setting-name
```

where “form-name” is the name of the form to which the settings apply, and “setting-name” is the name of the settings as saved in the symbiont settings library.

For example, suppose that you have a setting stored in the symbiont library called TWO-UP that invokes two-up printing. Whenever a job is printed on the form “2UP”, you want to print automatically using two-page layup. To create the proper definitions, enter the commands:

```
$ DEFINE/FORM/STOCK=DEFAULT 2UP 99

$ DEFINE/SYS SSV$FORM_2UP_SETTINGS TWO-UP
```

Provided the form name “2UP” and form number “99” are not currently in use on your system.

12.1.1 Using SPD_MGR to Manage Defaults

The command procedure, SPD_MGR.COM simplifies the customization of the settings stored in the symbiont settings library. To invoke this procedure from the command line, use the command:

```
$ @SCRIPTSERVER:SPD_MGR
```

You can also access this procedure from the SSV$CONFIG command file. The top most menu of this command file contains six prompts, the last of which:

```
6 Create ScriptServer Default Settings
```

activates the SPD_MGR command procedure.

The procedure:

- invokes the SCRIPT facility using the command:

  ```
  $ SCRIPT/LIBRARY=SCRIPTSERVER:SPD$LIBRARY.DAT
  ```

  to allow manipulation of settings in the symbiont settings library.
- allows you to define system-wide logical names to activate a setting-name as the default for a queue, the system, or an OpenVMS form

For example, suppose you want to set up a system-wide default that uses the font *Times Roman* instead of ScriptServer's default of *Courier.*

1. Invoke SPD_MGR, which places you in the SCRIPT facility with the symbiont settings library as the current library.
2. Use the FO command to change the font to *Times Roman*.
3. Save the settings (with the SS command) under the name SYS-DEFAULTS.
4. Exit SCRIPT by pressing CTRL / Z.
5. The procedure will prompt you for whether or not you want to create a logical name definition. Respond YES to this prompt. (It is this logical name which provides the link to the setting-name for the ScriptServer print symbiont).
6. SPD_MGR then prompts you for the type of default you want to establish. Enter S for system-wide.
7. You will then be asked for the name of the setting that you want to define as the system-wide default. Respond to this prompt with SYS-DEFAULTS.

8. The procedure will create the logical name definition and display it at your terminal so you can add it to your system start-up file. Once the logical name is defined, the symbiont will use the SYS-DEFAULTS setting as the default settings for subsequent PRINT requests.

SPD_MGR and /DYNAMIC

The /DYNAMIC qualifier does not function with the SPD_MGR procedure. In order to create SCRIPT defaults requiring the functionality provided by the /DYNAMIC qualifier, it is necessary to access the SCRIPT menu directly and then manipulate the settings from within the system-wide library.

1. With the LI command, change the library to:

   ```
   SCRIPTSERVER:SPD$LIBRARY.DAT
   ```

2. Create the settings required and save them with the SS command.

Now you can use the logical:

```
SSV$SETTINGS setting-name
```

to manipulate your various default settings. Remember you have to STOP and RESTART your queues for these settings to take affect.

SPD_MGR and SSV$CONFIG.SSV

If you are using the SSV$CONFIG.SSV configuration file for management of your ScriptServer logicals, the SPD_MGR command procedure will automatically place any logicals created in this file. If you are not using the configuration file, the logicals will be defined into the system table and will work as normal.

12.2 CONTROLLING THE PRINT LOG

You can control the circumstances under which the ScriptServer print symbiont generates a Print Log using the logical name SSV$PRINTLOG_LEVEL. It may be defined so that it applies to all queues system-wide, or made specific to a given queue. To specify a particular queue, use the form SSV$queue_PRINTLOG_LEVEL. The equivalence of this logical name may be defined as follows:

ALL	causes a Print Log to be generated whenever a message is received from the printer (default)
FATAL	only jobs that cause the message Flushing: *rest of job (to end-of-file) will be ignored*
	cause the symbiont to generate Print Logs
NONE	disables Print Log generation

For example, to cause ScriptServer to generate Print Logs only when a fatal condition is detected for all queues, use the command:

```
$ DEFINE/SYSTEM SSV$PRINTLOG_LEVEL FATAL
```

NOTE: *Because the error message:*

```
Flushing: rest of job (to end-of-file) will be ignored
```

is a PostScript error, defining this logical to FATAL when in PCL mode results in the same behavior as if the logical had been defined to ALL

12.3 CONTROLLING THE FLAG PAGE

In addition to the SCRIPT command, FP and qualifier, /FLAG, there are 3 logical names available for controlling the output of a flag page on a queue by queue basis. The SCRIPT command and qualifier let you control whether or not a flag page is printed, the following logical names control the tray from which the flag page will feed. The logical names are as follows:

```
$ DEFINE/SYS SSV$queue_FLAG_TRAY_COMMAND “trayselection command”

$ DEFINE/SYS SSV$queue_FLAG_PAPER_SIZE paper-size
```

The paper-size logical should only be used in conjunction with the tray-command logical. If left undefined, the paper-size logical will default to the international default setting for the tray chosen. The tray-selection logical can be defined to equal the PostScript command for the desired tray (consult your printer documentation for the exact sytnax of this command) or to the appropriate PCL-based escape sequence for the tray (consult your printer documentation for this escape sequence).

There is also a logical available specifically for PCL5 compatible printers:

```
SSV$queue_PCL_FLAG_COMMAND
```

which allows you to select a non-default paper tray for flag and trailer pages with PCL-based escape sequences. See the discussion of these particular logicals in Chapter 23 for a more detailed discussion of the use of PCL-based escape sequences to manipulate flag pages.

12.4 CHECKPOINTING

In the event of a system or symbiont crash, the software will restart the print job at the last page checked before the failure occurred.

NOTE: *This feature is only applicable to PCL print jobs and text-to-PostScript translated print jobs.*

By default, the symbiont checks the print job at every other page. However, you can control the interval between checks with the following logical:

```
SSV$CHECKPOINT_FREQUENCY
```

How often the symbiont performs the check is determined by defining the logical as follows:

```
$ DEFINE/SYSTEM SSV$CHECKPOINT_FREQUENCY #
```

where “#” is an integer corresponding to the number of pages between checks. For example, if you wish the symbiont to check the printjob every third page you would use the following definition:

```
$ DEFINE/SYSTEM SSV$CHECKPOINT_FREQUENCY 3
```

NOTE: *By limiting checkpointing to every other page, the symbiont avoids excessive overhead. Decreasing checkpointing frequency will decrease overhead correspondingly and vice-versa.*

12.5 CONTROLLING DEVICE INACTIVE SENSITIVITY

ScriptServer uses the CTRL/T status request on queue start-up to verify the presence and status of the PostScript printer. This is also done at the start of a print job. By default, if the Script-

Server symbiont is unable to elicit a response from the printer after several attempts, the symbiont will abort the job with the completion status DEVINACT, device inactive.

This behavior can be controlled on a system-wide basis with the logical:

```
SSV$DEVINACT_OPTION argument
```

And on a queue-specific basis with the form of the logical:

```
SSV$queue_DEVINACT_OPTION argument
```

where “queue” is the name of the particular ScriptServer queue, and “argument” is one of the following:

- FAIL (default) causes the job to be aborted with the error
- DEVINACT, device in-active
- SILENT sends no messages to the operator, continually retries to connect to printer
- WARNING sends a warning message to the operator on the first unsuccessful attempt to connect to the printer

NOISY sends a warning message to the operator on each unsuccessful attempt to connect to the printer

The time interval between attempts to connect can be controlled system wide using the logical name:

```
SSV$DEVINACT_INTERVAL time-value
```

And on a queue specific basis with:

```
SSV$queue_DEVINACT_INTERVAL time-value
```

where “queue” is the name of the particular ScriptServer queue. The default value is 120 seconds.

NOTE: *Time values must be given in seconds.*

12.6 ACCOUNTING DATA PROVIDED

Provided that there is a backchannel or the queue is using the QMS protocol for TCP / IP, the ScriptServer print symbiont provides the information for the following fields in an accounting record for each print job:

- GETs from source the number of RMS input operations from the print file(s) performed to print the job
- QIOs to printer number of write operations performed to the PostScript printer
- Pages printed the number of actual sheets of paper printed for the job. When using two-page layup, this is roughly half of the logical pages printed.

12.7 SCRIPTSERVER QUEUE CHARACTERISTICS

As previously noted, ScriptServer uses queue characteristics to control the routing of ScriptServer print requests. These characteristics are defined at the time ScriptServer is installed. If a new system queue data file (JBCSYSQUE.DAT, in OpenVMS 5.4 and earlier, and QMAN$MASTER.DAT in VMS 5.5 and later) is created, all defined queues, forms, and characteristics are lost. In this case, the queue characteristics need to be redefined for the job routing function to behave properly.

ScriptServer is distributed with a command file (CHAR_DEF) that may be used to redefine queue characteristics. To execute the command procedure, use the command:

```
$ @SCRIPTSERVER:CHAR_DEF
```

If your site is using downloaded fonts, further characteristic definitions may be made in SITE_DEF.COM, which is automatically executed by CHAR_DEF.

12.7.1 Characteristics with Foreign Queues

When ScriptServer is running in foreign queue mode, it copies the characteristics set on the foreign queue to itself. For example, if ScriptServer queue HYDRO is feeding a foreign queue named LPS20_QUEUE that uses characteristics 9, 10, 20 and 21, HYDRO will set those characteristics on itself as well.

NOTE: *Through the use of the logical name:*

```
SSV$queue_CHARACTERISTICS
```

you can have ScriptServer add a list of characteristics to those it sets on itself. See below for more information.

12.7.2 Characteristics on Queues without Backchannel

When a backchannel is not available, ScriptServer has no way to obtain device-specific information from the printer to determine its abilities. In this case, the symbiont will set the characteristics that correspond to an Apple LaserWriter plus all paper characteristics for the paper system in use. If international defaults are in effect, metric paper sizes (A3, A4, B4, etc.) are used, otherwise North American paper sizes (Letter, Legal, etc.) are used. You may explicitly specify additional characteristics using a logical name as described in Section 12.7.3 "*Specifying Additional Queue Characteristics*" on page 211.

12.7.3 Specifying Additional Queue Characteristics

Normally, queue characteristics are automatically handled by the ScriptServer print symbiont. However, in certain circumstances it is necessary to explicitly specify a list of characteristics to be used in addition to those that the symbiont determines are needed. This is a useful feature when the communication link to the printer does not provide a backchannel.

To specify additional characteristics to those as determined by the print symbiont, define a logical name that equates to the list of characteristics to be added. For example, to add characteristics 93, 94, and 99 to queue LASER, define a logical name as follows:

```
$ DEFINE/SYS SSV$LASER_CHARACTERISTICS “93,94,99”
```

12.8 SCRIPTSERVER AND THE exitserver OPERATOR

The PostScript language provides the *exitserver* operator to exit the PostScript server loop. The server loop provides a context that allows the printer to reclaim virtual memory and restore the default PostScript graphics state at the conclusion of each print job.

Changes that are made outside of the server loop are persistent from job to job. For example, permanently-downloaded fonts must be downloaded outside the server loop so that they remain available for all subsequent jobs (until the printer is turned off). Depending on the nature of the change, it may become permanent, surviving even power-off. This is desirable behavior in cases such as setting the printer name or communications characteristics.

Because the contents of printer memory may be permanently changed (with potentially disastrous effects) outside the server loop, the *exitserver* operator is protected by a password so that access to it is controlled. However, almost all PostScript printers use 0 as the password to the *exitserver* operator, which is rarely changed (although it is possible to do so with the *set password* operator). To prevent unauthorized access to the *exitserver* operator, it is disabled by ScriptServer for all PostScript print jobs, except for those that are submitted by users with a UIC group of 1.

If a PostScript print file attempts to exit the server loop when the exitserver operator is disabled, the following message is generated by the printer and reported on the Print Log: ScriptServer: Unauthorized attempt to execute *exitserver*. If this behavior is undesirable at your installation, you can enable access to *exitserver* for all users by defining a logical name as follows:

```
$ DEFINE/SYS SSV$ENABLE_EXITSERVER TRUE
```

NOTE: *This protection mechanism is a security feature of the ScriptServer software. There is at least one PostScript virus that exploits the fact that most PostScript printers have a password of 0 and can permanently damage the printer. Therefore, disabling ScriptServer's protection of the PostScript operator* ***exitserver*** *is strongly discouraged.*

12.9 ADDITIONAL OPTIONS FOR POSTSCRIPT ENCODING

The options for PostScript Encoding Schemes in ScriptServer's ASCII to PostScript translation are DMCS (the default), PC, and CST. CST provides a template, allowing modification without losing one of the other schemes. This could prove helpful when a language other than English is in use on a system.

To effect the change you will need to extract the prologue (SSV$PROLOG) from the ScriptServer device control library (SYS$LIBRARY:SSV$DEVCTL.TLB):

```
$ LIBRARY/EXTRACT=SSV$PROLOG/OUT=SSV$PROLOG.PS -
_$ SYS$LIBRARY:SSV$DEVCTL.TLB
```

Modify the encoding array (called CUCS) in the SSV$PROLOG. This array contains the name of the character description indexed by ASCII value. Do this with a text editor:

```
$ EDIT/TPU SSV$PROLOG.PS
```

NOTE: *The names of the characters corresponding to the ASCII values can be found in an appendix in the PostScript Language Reference Manual.*

After saving your changes to the prologue, you will need to stop all of your ScriptServer queues in order to place the modified module back into the device control library using the following command:

```
$ LIBRARY/REPLACE SYS$LIBRARY:SSV$DEVCTL.TLB SSV$PROLOG.PS
```

Restarting your ScriptServer queues will allow the CST encoding scheme to be selected through the SCRIPT menu or via the appropriate PRINT Parameter.

NOTE: *Future releases of the software may provide a new version of the SSV$PROLOG module, so this procedure may need to be repeated after an upgrade.*

Chapter 13

ScriptServer Soft Forms

The Soft Forms utility, SSV$EPS2FORM, is an electronic form capability for the creation of "backdrop" page descriptions (containing graphics, text, images, etc.) that can be used with text files printed through ScriptServer. Page descriptions must be in the Encapsulated PostScript file format and can be created with any software that generates EPS files.

SSV$EPS2FORM turns this EPS file into a PostScript Level 2 form. The print symbiont then calls the form at the beginning of every page. Once the page has been described according to the EPS file, straight ASCII text can be printed across it. Additionally, the resulting “soft form” can be automatically generated simply by associating it with an OpenVMS printer form accessed with the PRINT command.

13.1 REQUIREMENTS

There are two requirements for basic use of soft forms:

- an EPS file (which must be located on your OpenVMS system);
- a PostScript Level 2 printer.

However, if you want to download the form permanently to the printer (see below), the printer must have enough memory to hold the PostScript file, and double sided forms (also discussed below), require a printer with duplex capability.

13.2 SSV$EPS2FORM

ScriptServer soft forms consist of two files: a PostScript language form description which defines the form in the printer, and a PostScript language file that invokes the form at the desired size and location on the page. Both of these files are drawn from the original EPS file.

The executable used to convert an EPS file for use as a ScriptServer soft form is:

```
SCRIPTSERVER:SSV$EPS2FORM.EXE
```

Activate the program with the RUN command:

```
$ RUN SCRIPTSERVER:SSV$EPS2FORM
```

The program consists of a series of prompts which create and name the files and forms required for use, and then places and scales the image on your page:

```
EPS file to convert to a PostScript form?

Form module name to create [filename_FORM.PS]?

PageSetup module name to create [filename_SETUP.PS]?

What do you want to name the PostScript form [filename]?

Distance from left edge of page for left edge of form [0in]?

Distance from bottom edge of page for bottom edge of form [0in]?

Form image width #.# points [ #.#in; #.#cm ]

Form image height #.# points [ #.#in; #.#cm ]

Scale factor for form[1]?
```

NOTE: *The naming conventions (based on the filename of the original EPS file) suggested by the default responses to the second, third, and fourth prompts ensure consistent naming of the files and forms created in this process. This may prove helpful when maintaining the forms and files.*

```
EPS file to convert to a PostScript Form?
```

Respond to this prompt with the name and directory specification of the EPS file you will use in the soft form. EPS is the default file type if none is specified.

Form module name to create [filename_FORM.PS]?

This is the definition file. This file contains the actual PostScript code from your EPS file in addition to mechanisms that place it in a PostScript Level 2 forms context. The default filename created by this prompt is *filename*_FORM.PS where “filename” is taken from the original .EPS file. The .PS file extension, however, is required no matter what you choose to call the file.

PageSetup module name to create [filename_SETUP.PS]?

This is the *setup file*. This file contains the PostScript code ScriptServer requires for placing the image contained in the definition file on every page. The default response to this prompt appends "_SETUP.PS to the original .EPS filename. The .PS file extension is required no matter what you choose to call the module.

What do you want to name the PostScript form [filename]?

This refers to the PostScript Level 2 form that will reside on the printer either permanently or per job depending on configuration (these two scenarios are discussed in detail below). ScriptServer references this form when describing the page for printing. The default response uses the original EPS filename without a file extension. We recommend that no file extension be used when naming this form.

Distance from left edge of page for left edge of form [0in]?

Distance from bottom edge of page for bottom edge of form [0in]?

These two prompts locate the image at a specific point on the page.

Responses must be given in points (PT), inches (IN), or centimeters (CM). If no measurement specification is given and international measurements are in use, values will default to CM. Otherwise, values will default to IN.

NOTE: *A soft form uses the page size of the file printed with the form as its paper size, hence distances should be entered accordingly. For example, placing an image nine inches from the bottom of a letter size page on a form printed with a landscape oriented file, will result in the image being placed off the page. Thus, it may be necessary that separate forms be created for separate paper sizes and/or orientations.*

Scale factor for form? [1]

Prior to this prompt, SSV$EPS2FORM displays two informational lines showing the dimensions of the EPS image given in points, inches, and centimeters:

```
Form image width #.# points [ #.#in; #.#cm ]
Form image height #.# points [ #.#in; #.#cm ]
```

The scale factor allows uniform scaling of the image based on this image size. A value of one (the default) leaves the image size unchanged. A positive integer increases the image size by a factor of that number. For example, a response of 2 doubles the dimensions, 3 triples them, and so on. A positive number less than one, decreases the image size accordingly. For example, a response of .5 shrinks the image to one half the original size, .25 to one fourth, and so on.

NOTE: *Responses to the scaling prompt must be given as positive numbers.*

Pressing <RETURN> without entering a new scaling factor will exit the procedure without changing the image size. If you want to change the size, enter the desired value. SSV$EPS2FORM will display the new dimensions of the image again followed by the prompt:

```
Scale factor for form [x]?
```

where “x” is the scale factor previously entered. This allows you to continue to adjust the size of the image in order to get the exact dimensions required.

NOTE: *Subsequent scaling done by SSV$EPS2FORM as a result of subsequent responses to the scale factor prompt will be done according to the ORIGINAL dimensions of the image and NOT the dimensions of the image resulting from previous scalings.*

Exit the scaling factor loop by pressing <RETURN> without entering a new number. This will exit the procedure, adjusting the image size according to the factor indicated within the brackets of the prompt. Entering another value will start the process again, adjusting the image size according to the new value.

13.3 USING YOUR SOFT FORM

Once you have created the files and named the forms required of the program, it is ready for use. Before you print with your soft form, however, you should decide whether the form should be permanently resident on the printer, or downloaded on a per-job basis.

13.3.1 Preparing a Soft Form for Use

Both permanently downloading a soft form and downloading the form on a job-by-job basis require that you first place the setup and definition files created by SSV$EPS2FORM in the ScriptServer Device Control Library:

1. Make a copy of the ScriptServer Device Control Library:

   ```
   $ COPY SYS$COMMON:[SYSLIB]SSV$DEVCTL.TLB SYS$COMMON[SYSLIB]
   ```

   This allows you to insert the required modules into the library without having to stop all of your ScriptServer queues before doing so. When ScriptServer queues are stopped and then restarted, they will use the newest version of the ScriptServer Device Control Library, in this case, the one containing the necessary files.

2. Place the setup file in the Device Control Library:

   ```
   $ LIBRARY/INSERT SYS$COMMON:[SYSLIB]SSV$DEVCTL.TLB filename_SETUP.PS
   ```

   where “filename_SETUP.PS” is the name of the setup file created and named during the SSV$EPS2FORM procedure discussed above.

3. Insert the definition file:

   ```
   $ LIBRARY/INSERT SYS$COMMON:[SYSLIB]SSV$DEVCTL.TLB filename_FORM.PS
   ```

   where “filename_FORM.PS” is the name of the definition file as named during the SSV$EPS2FORM procedure discussed above.

4. Stop your ScriptServer Queues:

   ```
   $ STOP/NEXT queue-name
   ```

   Pause five to ten seconds while the queue shuts down.

5. Re-start your queues:

   ```
   $ START/QUEUE queue-name
   ```

Your queues and the ScriptServer soft form are now ready for use. As stated above, soft forms can be downloaded to the printer permanently or on a job-by-job basis. Each of these scenarios is discussed below.

13.3.2 Permanently Downloading SoftForms to the Printer

Permanently downloading a soft form to the printer will dramatically increase throughput due to the fact that the file is not sent to the printer with every job. To permanently download the soft form you will need to add the definition file:

```
filename_FORM.PS
```

to the ScriptServer Printing System file:

```
SCRIPTSERVER:DOWNLOAD.COM
```

1. Stop a queue associated with the printer to which you want to permanently download this soft form:

   ```
   $ STOP/NEXT queue-name
   ```

2. Using a text editor, open DOWNLOAD.COM in the SCRIPTSERVER directory and add the following command to the file:

   ```
   $ DEFINE/TABLE=SSV$DOWNLOAD_TABLE SSV$queue-name_MODULES -
   _$ filename_FORM.PS
   ```

   where “queue-name” is the name of a queue pointing at the printer to which you want to download the form. The file, DOWNLOAD.COM, contains comments explaining this procedure.

3. Run the download procedure:

   ```
   $ @SCRIPTSERVER:DOWNLOAD
   ```

4. Re-start your ScriptServer queue:

   ```
   $ START/QUEUE queue-name
   ```

Once you have added the above line from step number 2 to DOWNLOAD.COM, you will need to define an OpenVMS form referencing the file as a ScriptServer PageSetup module:

```
$ DEFINE/FORM/STOCK=DEFAULT/SETUP=filename_SETUP form-name form-number
```

Where “filename_SETUP” is the setup file inserted into the Device Control Library earlier in this procedure, “form-name” is any name you wish to name your OpenVMS form, and “form-number” is any unused form number for referencing the form.

NOTE: *With printer-resident modules, turning off the power will clear the printer’s memory. However, because the module has been added to DOWNLOAD.COM, simply stopping and restarting the queue will reload the module.*

13.3.3 Non-printer Resident Soft Forms

You do not have to download soft forms permanently to the printer. To prepare a soft form for downloading on a job-by-job basis, you need only define the following logical:

```
$ DEFINE/FORM/STOCK=DEFAULT/SETUP=(filename_SETUP, -
_$ filename_FORM) form-name form-number
```

Where “filename_SETUP” and “filename_FORM” are the setup and definition files previously inserted into the ScriptServer Device Control Library, “form-name” is any name you wish to call your OpenVMS form, and “form-number” is any unused form number that will reference the form.

13.3.4 Printing with Your Soft Form

Use the soft form from either the command line with either the SCRIPT or PRINT commands, or from the SCRIPT menu by simply calling the form in the manner appropriate to the printing method. The following example demonstrates the proper syntax for use of the soft form with the OpenVMS PRINT command:

```
$ PRINT/QUE=queue-name/FORM=form-name filename.txt
```

For permanently downloaded soft forms, “queue-name” is the name of a queue associated with the printer to which you downloaded the soft form, “form-name” is the name of the form you just defined and “filename.txt” is the text file you wish to print across the soft form. ScriptServer will reference the form that is resident on the printer.

If you have not downloaded the form permanently to your printer, ScriptServer will download the form along with the file, thus "queue-name" must simply refer to a queue associated with a PostScript Level2 printer.

NOTE: *See Chapter 8 above for details on using OpenVMS forms with the SCRIPT facility.*

13.3.5 Double Sided SoftForms

It is possible to print double-sided forms so long as the back side of the form does not require data. For example, purchase orders frequently require a double sided form the front side of which requires that data be printed across a background image, and the back side of which requires a static image consisting of terms of agreement or similar information. The following procedure describes creation and use of double sided soft forms.

1. Create the image for the front of your form from an EPS file using SSV$EPS2FORM as described above.
2. Create the image for the backside of your form from an EPS file using SSV$EPS2FORM as described above.
3. Using a text editor, create the setup module that will contain both the image(s) for the front of the form to be used with ASCII data and the static image for the back side of the form. Use the format described in Section 5.3.1 *"Automatic Setup Modules*" on page 99 for Multiple Page Setup Modules.

- Use a standard text editor to extract the PostScript code from the definition file created by SSV$EPS2FORM in step 1. This will be the front side of the form. Use this PostScript code as the definition of *ssv$PageSetup* as described in Chapter 5 concerning Multiple Page Setup Modules.
- For the definition of *ssv$BackSetup* (also described in Chapter Four), use the PostScript code from the definition file created by SSV$EPS2FORM in step 2. This will be the back side of the page.

4. Insert the Multiple Page Setup module into the ScriptServer Device Control Library just as you would a regular soft form.
5. Associate this double sided soft form to an OpenVMS form:

```
$ DEFINE/FORM/STOCK=DEFAULT/SETUP=filename form-name form-number
```

Where “filename” is the same as that of the file inserted into the Device Control Library in step 4, “form-name” is any name you wish to name your OpenVMS form, and “form-number” is any unused form number referencing the form.

6. Use your double sided form just as you would any other soft form, from either the command line or the SCRIPT menu. Both of these methods are described above.

PART 3

REFERENCE

Chapter 14

Troubleshooting

Please read this chapter carefully if you are having problems printing with the ScriptServer software. The majority of problems encountered by users result from improper printer or communications setup. This chapter will help you determine the specific nature of your problem, and the steps required to help you solve it as quickly as possible.

If you are unable to solve your problem by following the diagnostic procedures in this chapter, refer to Section 14.5 *"If You Still Have a Problem*" on page 233 below for information about obtaining technical support from GrayMatter Software Corporation.

14.1 POTENTIAL PROBLEMS AND THEIR SOLUTIONS

Before proceeding with troubleshooting, it is important that you issue a REPLY/ENABLE command. This will cause any OpenVMS console messages to be sent to your terminal. You must have OPER privilege to issue this command.

14.1.1 Symptom: Queue Will Not Start

If backchannel is enabled when a ScriptServer queue starts, the queue will automatically go into an “idle” state. However, at this time, the queue is conducting a dialogue with the printer, the stages of which will be indicated in the status line of the queue description. The initial state will be “connecting” which is normally followed by “downloading fonts” etc. (See Chapter 4 for a detailed explanation of the queue description.)

If the queue remains in the “connecting” state for an excessive length of time, check the system console for OPCOM error messages from the ScriptServer print symbiont. If you receive the following message after starting a ScriptServer queue:

```
%SYSTEM-F-DEVINACT, device inactive
```

it indicates that ScriptServer is unable to get a response from the printer to a CTRL/T status request or PJL echo command. This symptom has a number of potential sources, however, the most common cause of this particular problem has to do with communication setup. If using serial communications, verify that they are intact as described below in Section 14.3 *"Verifying Communications*” on page 230. If the printer appears to be behaving properly and you are still unable to start the queue, contact GrayMatter Software for technical support as described in Section 14.5 *"If You Still Have a Problem*” on page 233.

14.1.2 Symptom: Queue Starts Without Characteristics

If you are using queue characteristics with ScriptServer (see Section 3.5 *"Queue Characteristics in ScriptServer*” on page 58), the queue should have several characteristics associated with it the first time it is started. Verify this by issuing a SHOW QUEUE/FULL command after starting a newly created queue. You should see the /CHAR qualifier, followed by a list of characteristic numbers. For example:

```
/CHAR=(9,10,20,21,22,23,24,25,26,27,28,29,30)
```

Failure to establish characteristics on a queue is almost always caused by a communication problem. Another possibility is that the printer is not in PostScript mode. These are discussed in Section 14.2 *"Checking the PostScript Printer*” on page 229 and Section 14.3 *"Verifying Communications*” on page 230.

NOTE: *If your printer is connected with a parallel interface (or another configuration without backchannel) ScriptServer will automatically set the queue characteristics as though connected to an Apple LaserWriter. See Chapter 4 for more details on the default characteristic set.*

14.1.3 Symptom: Problems Printing Files

Several problems can cause files to print improperly, or not at all. When troubleshooting file-printing problems:

- Include the /NOTIFY qualifier on your print command, or set the NT option to “Yes” in the SCRIPT menu. This will usually provide more information if the problem is related to an OpenVMS error.

- If you are having problems with a PostScript printer, include the EHANDLER setup module when submitting the file for printing. This will force any PostScript error messages to be printed.

You can do this a couple of ways:

- include the /SETUP=EHANDLER qualifier on the PRINT command, or
- set the SM option to EHANDLER in the SCRIPT menu.

NOTE: *Some software products that submit files to the ScriptServer print queue require special setup procedures.*

If you are trying to print a PostScript file, and the printer is printing the actual PostScript code instead of converting it to text and graphics, it means one of three things:

- the ScriptServer thinks that the input file is an ASCII text file,
- the printer is NOT in PostScript mode,
- if it is in Automatic File Sensing mode, the printer doesn't recognize the file as PostScript.

Refer to Chapter 3 for details regarding ScriptServer file type recognition.

NOTE: *Be sure you have OPCOM turned on with the REPLY/ENABLE command. Many PCL error messages will be returned to the user via OPCOM. See Chapter 16 for a list of PCL messages.*

14.2 CHECKING THE POSTSCRIPT PRINTER

14.2.1 PRINTER_STATUS.PS

In addition to providing verification that your PostScript printer is in PostScript mode, the file, PRINTER_STATUS.PS, provides a variety of other information about the printer and its configuration. To bypass ScriptServer (in the event that you are experiencing problems) and print this file:

1. Make sure that serial or LAT connected queues are not spooled:

   ```
   $ SET DEVICE/NOSPOOL ttcx:
   ```

   where ttcx" is the port to which your printer is connected.

2. Stop the queue:

```
$ STOP/RESET queue-name
```

3. Copy the file directly to the printer:

```
$ COPY SCRIPTSERVER:PRINTER_STATUS.PS ttcx:
```

   where ttcx" is the port to which your printer is connected.

If this fails to produce a printer status page, and you are sure that the communications are set up correctly, then your printer may not be operating in PostScript mode.

Check the following items:

- Ensure that the printer is powered on and is on-line.
- Check that the communications cable is connected to the correct port on the printer.
- If your printer has a switch to select the mode of operation, make sure it is in PostScript batch mode. On some printers, this may also effect the baud rate.
- If the printer requires a PostScript cartridge, make sure the cartridge is properly installed according to the manufacturer's instructions. Verify that the cartridge contains an Adobe-compatible PostScript interpreter.
- Verify that the printer has at least two megabytes of memory.
- Turn the printer off and on to clear any emulation mode that may have been set.

If you are still having trouble, contact GrayMatter Software Technical Support.

NOTE: *Be sure you have the information listed in Section 14.5 "If You Still Have a Problem" on page 233 ready for the support person.*

14.3 VERIFYING COMMUNICATIONS

Whenever connecting a printer for the first time or setting up a printer for use with the ScriptServer Printing System, it is recommended that you first verify that system communications with the printer are working properly.

Follow the directions below depending on the connection method used for your printer.

14.3.1 Checking Serial/LAT Connections

To test the connection to your printer, make sure the queue is stopped.

1. Verify that the port is not spooled:

```
$ SET DEVICE/NOSPOOL ttcx:
```

NOTE: *"ttcx" is the port to which your printer is connected.*

2. Connect your terminal to the printer:

```
$ SET HOST/DTE ttcx:
```

You should receive the message:

```
%REM-I-TOEXIT, connection established, type ^\ to exit
```

2 a. If checking a PCL printer, type CTRL / L (hold down the control key and press L). This will send a form feed to the printer and, if communications are set properly, force the printer to eject a blank page.

2 b. If checking a PostScript printer, type a CTRL / T (hold down the control key and press T). This requests a status message from the PostScript printer. If the printer is not busy and communications are working properly, you will receive a message like the following:

```
%%[ status: waiting; source: serial 25 ]%%
```

If you receive the above message, press CTRL / D, followed by another CTRL / T. You should then get a message similar to:

```
%%[ status: idle ]%%
```

If the printer generates these messages, the communications characteristics are set properly (with the possible exception of flow control, which is not verified by this procedure).

3. To exit SET HOST, press CTRL / \ (control / backslash).

If you do not receive status messages like those shown above from a PostScript printer, or if you are unable to force a blank page from a PCL printer, make sure:

- You are using the correct cable, and it is connected to the correct port on the printer.

- The baud rate (for serial connections) is the same on the printer, the OpenVMS terminal settings, and the terminal server port (if applicable).

If you are able to communicate with the printer using SET HOST/DTE, but are still having trouble starting the queue or printing files, you may need to switch the printer's flow control from DSR/DTR to XON/XOFF. Generally, this is done via the printer's control panel.

14.3.2 Checking TCP/IP Connections

To verify communications on TCP/IP-connected printers, use the PING command of your TCP/IP networking software. This command is used to “bounce” a network packet off a remote node to verify that it responds at the established address. For example, to verify that a printer at internet address 192.100.10.5 is responding when using Digital’s TCP/IP Services for OpenVMS software, use the command:

```
$ UCX PING 192.100.10.5
```

You should get a response similar to the following:

```
%UCX-I-LOOPACT, 192.100.10.5 is alive
```

14.4 SCRIPTSERVER SNAPSHOTS

By default, the ScriptServer print symbiont will create the snapshot file:

```
SCRIPTSERVER:SSV$SNAPSHOT.LOG
```

every time the symbiont encounters a fatal error. This text file contains information about the symbiont at the time of the error. Included in this file is information like queue name, device name, logical name definitions, and printer type. If you have REPLY/ENABLE turned on at the time of the fatal error the file will also indicate the cause of the error.

If desired, you can increase the frequency with which ScriptServer creates this file. The following logical definition:

```
$ DEFINE/SYSTEM SSV$SNAPSHOT “STREAM”
```

will cause the symbiont to create this snapshot file every time a queue starts. Defining the logical to “TASK” will create the file every time a job starts.

NOTE: *You should not define this logical unless you have been directed to do so by GrayMatter Software Technical Support.*

14.5 IF YOU STILL HAVE A PROBLEM

If you are still having a problem, call the GrayMatter Software technical support hotline at (206) 329-8800. Before calling, please have the following information available:

- OpenVMS version
- Your machine is VAX or AXP
- ScriptServer version
- Printer manufacturer and model
- How the printer is connected to the VAX or AXP (TCP / IP, LAT, DECnet, etc.)
- Version of TCP / IP software, if applicable
- Model of terminal server, if applicable
- Description of symptoms
- Commands that produced the error
- Error messages, if any
- A ScriptServer Print Log, if applicable
- A copy of the ScriptServer snapshot file, SCRIPTSERVER:SSV$SNAPSHOT.LOG, if available

Chapter 15

ScriptServer Messages

15.1 POSTSCRIPT ERRORS IN THE PRINT LOG

The Print Log is used to communicate messages received from the printer during the processing of a job back to the originator of the job. Occasionally, these are PostScript-specific software errors, which are generated in the format:

```
%%[ Error: error-name; OffendingCommand: bad-command ]%%
```

When these errors prevent the printer from completing the print job (a fatal error), the following message is also generated:

```
%%[ Flushing: rest of job (to EOF) will be ignored ]%%
```

When this message is detected by the ScriptServer print symbiont, the job is terminated immediately, and a Print Log is generated (unless overridden with the print parameter NOMESSAGES).

15.2 SCRIPTSERVER COMPLETION STATUS ERRORS

The ScriptServer Printing System uses several completion status messages (reported to the originating user if the /NOTIFY qualifier is in effect) when reporting error conditions. These completion messages are specific to ScriptServer, and are summarized below.

15.2.1 DEVCMD, device command error

The DEVCMD error is generated when ScriptServer detects the fatal error described above in Section 15.1 *"PostScript Errors in the Print Log*" on page 235.

15.2.2 DEVINACT, device inactive error

The DEVINACT error is generated when the ScriptServer print symbiont is unable to generate a status message (using the CTRL/T character) from the printer after 10 attempts. This error is generally the result of the communications cable becoming unattached, the printer having been placed in an emulation mode, or set to an incorrect baud rate.

15.2.3 TIMEOUT, device timeout error

The TIMEOUT error is generated by ScriptServer when a LAT (Local Area Transport) connection to a remote printer is lost. This error is generally caused by a loss of connection to the remote terminal server. Another potential cause of TIMEOUT errors over LAT connections is when a printer error condition such as paper jam or paper out is not remedied promptly. Either of these may cause an inactivity timeout error on the terminal server port, resulting in a TIMEOUT error on the host. Inactivity timeouts can be disabled on a per-port basis on the terminal server.

15.3 SCRIPTSERVER OPCOM MESSAGES

OPCOM messages are of four types, Informational, Warning, Error, and Fatal. To receive ScriptServer OPCOM messages, you must have REPLY/ENABLE turned on. The following lists all possible ScriptServer OPCOM messages.

%SSV-E-BACKCHANERR, Error on backchannel device for queue
<OpenVMS error message text>

An I/O error occurred when accessing the backchannel device. The specific OpenVMS error that occurred is included in the message.

%SSV-E-BACKLATERR, LAT connection error on backchannel for queue <OpenVMS error message text>.

A LAT-related error occurred during ScriptServer's attempt to connect to a LAT serial backchannel. The OpenVMS error text is included in the message sent to the operator.

%SSV-W-BADSEGHDR, Invalid segment header byte

An error was detected while unpacking a PC-format packed font file. This indicates that the file has become corrupted.

%SSV-I-BCSTATUS, Backchannel is disabled

An informational message appearing only at queue start-up and only when backchannel is disabled. No message will appear if backchannel is turned on.

%SSV-I-BCSTATUS, Alternate serial backchannel set on device

An informational message appearing at queue start-up to indicate that an alternate serial backchannel is in use by this device.

%SSV-I-BCSTATUS, Backchannel is disabled on parallel printer

An informational message appearing only at queue start-up to indicate that backchannel is disabled on this device because it is connected to a parallel connection device.

%SSV-W-CHARDECERR, Error on characteristics declaration for queue <OpenVMS error message text>

An error occurred while attempting to declare characteristics on the queue. The corresponding OpenVMS error text is included in the message.

%SSV-F-CONERROR, can’t establish connection on queue
<OpenVMS error message text>

This is a general message indicating that ScriptServer was unable to establish a connection with the device. The OpenVMS error text is included in the message.

%SSV-F-CONTIMEOUT, Unable to establish connection on queue

After forty attempts, the symbiont was unable to connect to the printer. This may be due to a LAT configuration error. This message is sometimes accompanied by OpenVMS error message text.

%SSV-I-DOWNLOAD, Downloading font to device

This message indicates that a font is being downloaded to a printer. Unless otherwise specified via logical name, this message is sent for all font downloads.

%SSV-W-DOWNLOADERR, Error downloading font file
<OpenVMS error message text>

An error has occurred while downloading a font. This message is accompanied by an additional message explaining the error condition.

%SSV-F-DTVALERR, Error validating system date on ScriptServer queue
<OpenVMS error message text>

The system date is invalid or the license key is invalid in relation to it. An instance where this may occur is if the start date on the ScriptServer license key is sometime in the future, thus it has not yet expired, but it is still invalid. The OpenVMS error text is included in the message.

%SSV-FATALERR, Fatal exception in ScriptServer symbiont
<OpenVMS error message text>

The symbiont has crashed. The OpenVMS error text is included in the message. See the discussion in the Troubleshooting section of this document for details on creation of the Snapshot file.

%SSV-F-OUTIOERR, fatal output I/O error on device, queue, in module
<OpenVMS error message text>

An I/O error occurred while attempting to print the job. The specific OpenVMS error that occurred is included in the message.

%SSV-W-FONTACCERR, Error accessing font file
<OpenVMS error message text>

An I/O error occurred while accessing a downloaded font file. The OpenVMS error text is included in the message.

%SSV-W-FONTOPNERR, File open error on font file
<OpenVMS error message text>

An error occurred while attempting to open a font file for downloading. The OpenVMS error text is included in the message.

%SSV-F-FORQSTARTERR, Error creating start-up job on foreign queue
<OpenVMS error message text>

ScriptServer was unable to create its queue start-up job on the foreign queue. The OpenVMS error text is included in the message.

%SSV-F-HOSTSPECERR, error on remote host Internet address
<OpenVMS error message text>

The remote host internet address is unrecognized by the network. ScriptServer cannot make connection. The OpenVMS error text is included in the message.

%SSV-W-IGNBACKCHAN, Ignoring backchannel specification

An error has occurred which prevents access to a user-specified backchannel device. This message indicates that ScriptServer will proceed as though there were no backchannel available.

%SSV-F-IOWRITEERROR, Error writing data to printer on queue <OpenVMS error message text>

The printer is not accepting ScriptServer's request to write data to it. This is generally means the printer has dropped the connection.The OpenVMS error text is included in the message.

%SSV-F-IVCPUCLASS, This ScriptServer license does not support this type of CPU <OpenVMS error message text>

The currently installed ScriptServer license key does not support this CPU. Contact GrayMatter Software for a new ScriptServer license key. The OpenVMS error text is included in the message.

%SSV-F-IVPRINTER, Printer not supported by ScriptServer/LE

The PostScript printer in use is not supported by this software license. Contact GrayMatter Software to obtain a new license key.

%SSV-F-LATERROR, LAT connection error on queue <OpenVMS error message text>

A LAT-related error has occurred when trying to connect to a terminal server port over the Local Area Transport. A corresponding OpenVMS error accompanies this error message.

%SSV-F-LCKTBLDMG, License management table has been damaged <OpenVMS error message text>

This message will only occur when using per queue licensing provided by GrayMatter Software and the license management table has become corrupted. The OpenVMS error text is included in the message.

%SSV-I-LICENSEE, ScriptServer software licensed to <Company>

This is an informational message that will only appear at queue start-up and indicates to whom the software is licensed.

%SSV-F-LCSEXPIR, ScriptServer software license expired
<OpenVMS error message text>

The software license or demo has expired. Contact GrayMatter Software to renew your software license key. The OpenVMS error text is included in the message.

%SSV-F-LCSVALERR, Error validating ScriptServer license on queue
<OpenVMS error message text>

The information on the license key does not correspond to the information on the node. The OpenVMS error text is included in the message.

%SSV-F-LIBRERROR, Library open error on queue
<OpenVMS error message text>

An error occurred while trying to open the device control library (usually SSV$DEVCTL.TLB). The corresponding OpenVMS error text accompanies this message.

%SSV-E-LOADPP, Please load <paper size> in <paper tray>

Paper is out on the PCL printer to which the job is being printed. See the list of PCL / PJL error codes and messages for further explanation.

%SSV-F-LPDACKERR, Error receiving lpd acknowledgment on queue

A violation of the lpd protocol has occurred. ScriptServer was expecting an acknowledgment returned from the device and received nothing.

%SSV-F-LPDACKNOZERO, Error receiving a non-zero on queue

A violation of the lpd protocol has occurred. The device returned an acknowledgment but it was an error.

%SSV-F-LSOCKETBNDERR, local socket bind error on queue
<OpenVMS error message text>

ScriptServer tried to establish a TCP / IP communication channel and failed. The OpenVMS error text is included in the message.

%SSV-F-LUDPBNDERR, local UDP socket bind error on queue
<OpenVMS error message text>

ScriptServer tried to establish a TCP / IP communication channel to a QMS printer and failed. The OpenVMS error text is included in the message.

%SSV-F-NOANSWER, Printer not responding to status query on queue

The symbiont was unable to get the printer to respond to a CTRL / T status request inquiry after five attempts. This usually indicates a communications problem. See Chapter 21 for help debugging communications problems.

%SSV-F-NODQSSUPP, This ScriptServer license does not support DQS, aborting print job

A DQS print job was submitted to a ScriptServer queue without the DQS license option. Contact GrayMatter Software to obtain a software key supporting DQS.

%SSV-F-NONETWORKOPT, this ScriptServer license does not support network printers
<OpenVMS error message text>

The ScriptServer license does not support network communications. Please contact GrayMatter Software for a new license key. The OpenVMS error text is included in the message.

%SSV-W-NOSFDSUPP, This ScriptServer license does not support fax
<OpenVMS error message text>

The ScriptServer license does not support fax. Please contact GrayMatter Software for a new license key. The OpenVMS error text is included in the message.

%SSV-F-NOSTARTERR, Printer error prevents start-up on queue

An error condition on the printer has prevented ScriptServer from activating the queue in question.

%SSV-F-OUTIOERR, Fatal output I/O error on queue

The ScriptServer symbiont has crashed due to a fatal error either writing to or reading from the device to which the queue is pointing.

%SSV-I-PERMOD, Downloading permanent module to destination

This is an informational message appearing only at queue start-up indicating a permanent module has been downloaded to the printer hard disk.

%SSV-E-PERMMODERR, Permanent module download error
<OpenVMS error message text>

An error occurred while trying to permanently download a device control library setup module. The corresponding OpenVMS error message provides more detail about the error that occurred.

%SSV-E-PJLERR, PJL error status code

See Chapter 16 on PJL error codes for details on the specific status code.

%SSV-F-PKMSERR, License key error starting queue

The print symbiont was unable to validate the software license key. Contact GrayMatter Software to obtain a valid key.

%SSV-I-PROTOCOL, TCP/IP protocol compatibility mode set to protocol

This is an informational message indicating to which supported TCP / IP protocol the TCP / IP connection has been set.

%SSV-E-PTRERR, Printer error received from device
<PostScript message text>

A message was received from the printer containing the text “PrinterError”. These messages alert operators to error conditions such as paper out and paper jam. The exact message sent is printer-dependent.

%SSV-F-QLIMITXCD, Licensed active queue limit exceeded

This message indicates the number of queues allowed by your ScriptServer per queue license has been exceeded.

%SSV-E-REQMODERR, Error locating module in library
<OpenVMS error message text>)

An error occurred while attempting to locate a required module in the device control library. In release V3.2 and above, the required modules are SSV$PROLOG and SSV$HPGL_PROLOG.

%SSV-E-SYNCERR, Job synchronization error on queue

ScriptServer received an PJL EOJ (end of job) message from a PCL / PJL printer before it was expected.

%SSV-F-TCPOPTIONSERR, error setting TCP/IP options on queue
<OpenVMS error message text>

The system will not let ScriptServer set the TCP / IP options. The OpenVMS error text is included in the message.

%SSV-F-TCPIPCONNERR, TCP/IP connection error on queue
<OpenVMS error message text>

ScriptServer was unable to make a TCP / IP connection. The accompanying OpenVMS error message will further clarify the situation.

%SSV-E-UNSFORMAT, Font file is in an unsupported format

An attempt was made to download a font file that is in an unsupported format. Refer to Chapter 7 for information on the supported font file formats.

%SSV-F-UDPCONNERR, UCP/IP connection error on queue
<OpenVMS error message text>

ScriptServer was unable to make a TCP / IP connection to a QMS printer. The accompanying OpenVMS error message will further clarify the situation.

%SSV-F-UNSUPPORTED, Your ScriptServer license does not support this CPU; stopping queue

The currently installed license key does not support this CPU. Contact GrayMatter Software to obtain a license key.

%SSV-W-WHATEOJ, PJL end-of-job received before task completed

ScriptServer received an unexpected EOJ (end of job) on a PCL print job from a PCL / PJL printer.

%SSV-W-ZEROFONT, Zero length font file
<OpenVMS error message text>

When opening a font file for downloading to the printer, ScriptServer encountered an empty file. The accompanying OpenVMS error message will further clarify the situation.

Chapter 16

PCL/PJL Messages

Error messages from PCL / PJL printers are returned from the printer as five digit codes. The ScriptServer symbiont interprets all relevant codes and, depending on the nature of the error, will either print the error on a ScriptServer print log or display the error on the console via OPCOM. The format for the error reporting is as follows:

```
PJL status group PJL status code: PJL status error
```

For example, a PCL5 printer receives a PCL print job that includes a string missing a closing double-quote character. The printer will return the code, 20011, to ScriptServer which interprets this code and reports it as:

```
PJL parser error 20011: String missing closing double-quote character
```

16.1 PJL ERROR CODES

The following tables make up a comprehensive list of the PJL error codes that ScriptServer can interpret and report. Table titles correspond to PJL status groups, left columns to PJL status codes, and right columns to PJL status error text.

Table 16-1: PJL parser errors

PJL Status Code	PJL Status Error
20001	Generic syntax error; entire command ignored
20002	Unsupported command

Table 16-1: PJL parser errors (Continued)

PJL Status Code	PJL Status Error
20004	Unsupported personality/system
20006	Illegal character or UEL terminated PJL command
20007	Line terminator missing after closing quotes
20008	Invalid character in alphanumeric value
20009	Invalid character in numeric value
20010	Invalid starting character
20011	String missing closing double-quote character
20012	Numeric value starts with a decimal point
20013	Numeric value does not contain any digits
20014	No alphanumeric value after command modifier
20015	Option name missing value field
20016	More than one command modifier
20017	Command modifier encountered after option
20018	Command not an alphanumeric value
20019	Numeric value encountered when expecting alphanumeric
20020	String encountered when expecting alphanumeric
20021	Unsupported command modifier
20022	Command modifier missing
20023	Option missing
20024	Extra data received after option name
20025	Two decimal points in numeric value

Table 16-2:PJL parser warning

PJL Status Code	PJL Status Error
25001	Generic warning error; part of command ignored
25002	PJL prefix missing

Table 16-2:PJL parser warning (Continued)

PJL Status Code	PJL Status Error
25003	Alphanumeric value too long
25004	String too long
25005	Numeric value too long
25006	Unsupported option name
25007	Option name requires a value which is missing
25008	Option name requires a value of a different type
25009	Option name does not support values
25010	Same option name received more than once
25011	Ignored option name due to underflow/overflow
25012	Value for option truncated or rounded
25013	Value for option changed to range limit
25014	Value for option was out of range; ignored
25016	Option name value is not supported
25017	String empty; option ignored

Table 16-3:PJL semantic error

PJL Status Code	PJL Status Error
27001	Generic semantic error; part of command ignored
27002	Unmatched EOJ command
27004	Cannot modify a read-only variable

Table 16-4:Auto-continuable error

PJL Status Code	PJL Status Error
30010	Status buffer overflow
30016	Memory overflow; possible data loss
30017	Print overrun; image too complex
30018	Communication error; check port configuration
30027	Communication error; check port configuration
30034	Temporary printing error
30036	NVRAM recoverable error
30072	Remove manual feed page or cassette
30076	Personality memory overflow; out of memory

Table 16-5:PJL error

PJL Status Code	PJL Status Error
35029	Adaptive compression went to Lossy with data loss
35031	Printer language requested is not available
35037	Insufficient memory available for personality
35039	Low memory; Page protect:Letter, resolution:600
35040	Low memory; Page protect:A4, resolution:600
35043	Low memory; page protect:Letter, reolution:300
35044	Low memory; page protect:A4, resolution:300
35073	Check memory management
35074	Memory management not available
35075	User maintenance requested

Table 16-6:Printer error

PJL Status Code	PJL Status Error
40005	Ink cartridge alignment; setup error
40010	No ink / toner cartridge
40011	Error accessing ink cartridge
40021	Printer open
40022	Paper jam
40024	Acc. cartridge removed; replace and cycle power
40026	Replace paper cassette
40038	Low on toner
40046	Accessory cartridge removed; replace cartridge
40047	Accessory cartridge inserted; remove cartridge
40050	Int. svc. error; switch printer off for 15 minutes
40051	Printer detected temporary error condition
40052	52 ERROR
40053	SIMM configuration error
40054	Duplex unit hardware error
40055	Printer detected temporary error condition
40056	Cannot duplex envelopes; disengage duplex uni
40057	Int. svc. error; switch printer off for 15 minutes
40058	Int. svc. error; switch printer off for 15 minute
40059	59 ERROR
40061	Parity error on SIMM; cycle power
40062	SIMM memory board error
40063	Memory error in internal RAM
40064	Internal service error; cycle power
40065	Int. svc. error; switch printer off for 15 minutes

Table 16-6:Printer error (Continued)

PJL Status Code	PJL Status Error
40067	Int. svc. error; switch printer off for 15 minutes
40068	69 SERVICE

.

Table 16-7:SSV$-E-PJLERR, PJL error

PJL Status Code	PJL Status Error
50000	General hardware failure
50001	ROM error, ROM checksum failed
50002	RAM error, RAM test failed
50003	Engine fuser error
50004	Engine beam detect error
50005	Engine scanner error
50006	Engine fan error
50007	Engine communications error
50599	Processor error, power cycle

16.1.1 Paper Loading Messages

Paper loading messages are returned as one of two varieties:

- foreground
- background

Foreground

Foreground paper loading messages result when the paper input source is out of paper and the job cannot continue printing until the situation has been corrected. The PJL status code for all paper loading messages consists of five numbers representing the situation, the paper input device (i.e. upper cassette, multi-purpose tray), and the paper size in the following format:

```
41xyy
```

where “x” is the tray code and “yy” is the media code (both of which are detailed below)

Background

Background paper loading messages result when the paper input source is out of paper, but the job is able to continue printing through some fail-over routine. The PJL status code for all paper loading messages consists of five numbers representing the situation, the paper input device (i.e. upper cassette, multi-purpose tray), and the paper size in the following format:

```
11xyy
```

where “x” is the tray code and “yy” is the media code (both of which are detailed below).

PJL Tray and Media Codes

Table 16-8:Tray codes

PJL Status Code	PJL Status Error
0	Multi-purpose
1	Manual feed
2	Paper cassette
3	Lower cassette
4	Envelope feeder

Table 16-9:Media codes

PJL Status Code	PJL Status Error
00	Unknown paper
01	Unknown envelope
02	Letter paper
03	Legal paper
04	A4 paper
05	Exec paper
06	Ledger paper

Table 16-9:Media codes (Continued)

PJL Status Code	PJL Status Error
07	A3 paper
08	COM10 envelope
09	Monarch envelope
10	C5 envelope
11	DLenvelope
12	B4 paper
13	B5 paper
14	B5 envelope
15	Custom media

Chapter 17

PostScript Errors

The ScriptServer print symbiont reports PostScript errors by providing a Print Log when messages are received from the PostScript printer. This chapter describes the common PostScript errors that occur in the OpenVMS environment.

dictfull

This error indicates that there is no more room in a dictionary. PostScript dictionaries are created with space for a fixed number of key / value pairs. This error occurs when an attempt is made to exceed this limit. The offending command accompanying this error is usually *def* or *put*.

dictstackoverflow

The PostScript language provides a stack of fixed size onto which dictionaries are pushed using the *begin* operator. This mechanism provides the context within which names are resolved. When too many dictionaries are placed on this stack, this error occurs.

dictstackunderflow

This error occurs when an attempt is made to remove the bottom most dictionary on the dictionary stack, i.e. an end is executed for which there is no corresponding begin.

execstackoverflow

This error occurs when the execution stack has grown too large. As a procedure is called, the interpreter uses this stack to remember which point to return to within the procedure. When procedures are nested too deeply, the execution stack is exhausted and this error is generated.

invalidaccess

This error is generated when an attempt is made to reference a PostScript object (array, dictionary, file, or string object) in a way that violates its access attribute. For instance, attempting to modify a read-only dictionary will result in this error. A common cause of this error is an attempt to execute the *setdefaulttimeouts* or *setsccbatch* operators on a printer that requires that the *exitserver* operator be executed first.

invalidfont

This error occurs when a reference is made to an improperly formed font dictionary. The offending command is usually either the *findfont* operator, or the *show* operator. Another common cause is an attempt to use a non-existent font on a Digital PostScript printer (most other printers just fall back to the Courier font).

invalidrestore

This error indicates that an improper restore has been attempted. The *save* and *restore* operators are used to reclaim memory and generally enclose each page of a PostScript document. When this error occurs, at least one of the stacks contains a composite object whose values were created more recently than the save whose context is being restored.

ioerror

Some input/output error has occurred, which may happen as a result of any I/O operation, such as when accessing a printer's hard disk. The most common cause of this error is a serial communications configuration problem, usually when the printer is configured for DTR flow control when the host computer is using XON/XOFF. Another less common cause is unmatched data bits and/or parity between the printer and the host.

limitcheck

This error occurs when an implementation limit has been exceeded. Implementation limits are imposed on a device-dependent basis. For more information on implementation limits, refer to Appendix B in either edition of the PostScript Language Reference Manual (the “Red Book”).

nocurrentpoint

This error occurs when an operation requires a current point when none is defined, such as with the *lineto*, *curveto*, *currentpoint*, and *show* operators). The most common cause of this error is neglecting to perform an initial *moveto*.

rangecheck

This error occurs when an operand is out of the range of legal values for a given operator. An example of a situation that would cause this error is executing “3 setpapertray” when there are only two paper trays available.

stackoverflow

This error occurs when the operand stack has grown too large. Too many objects have been pushed onto the stack and not popped off. The size of the operand stack (generally 500 elements) is determined by a given PostScript implementation’s limits.

stackunderflow

This error occurs when an attempt is made to remove an object from the operand stack when it is empty.

syntaxerror

This error occurs when PostScript program text has been encountered by the interpreter that does not conform to the PostScript syntax rules.

timeout

Some time limit has been exceeded, such as the interpreter had to wait longer for a sheet of paper to be manually fed than the value specified by the *manualfeedtimeout* period.

typecheck

This error occurs when an operand of the wrong data type is passed to an operator. An example of this is passing a string value to the *add* operator.

undefined

A name has been used that cannot be resolved in the current dictionary context, or found in an explicitly referenced dictionary with the *get* operator.

undefinedresult

This error occurs when a numeric computation would produce a meaningless result or one that cannot be represented as a PostScript number. The most common cause of this error is an attempt to divide by 0, but it may also be caused by numeric underflow or overflow.

unmatchedmark

This error occurs when a mark object “[“ is sought on the stack, but none is present. Offending commands are usually *counttomark* or *cleartomark*.

VMerror

Virtual memory in the printer has been exhausted. Either an attempt has been made to create an object larger than the amount of virtual memory available, or a program has failed to make effective use of the *save/restore* mechanism.

Chapter 18

ScriptServer Logicals

Management of the ScriptServer Printing System is done largely through logical name definitions. The following pages describe all of the logicals used by the software.

18.1 SCRIPTSERVER LOGICAL NAMES

18.1.1 Types of Logical Name Definitions

There are three types of logical names in ScriptServer:

- queue-specific
- device-specific
- system-wide

System-wide Logical Names

A system-wide logical name cannot be defined in a more specific form. For example, the logical:

```
SSV$CHECKPOINT_FREQUENCY
```

cannot be defined specific to a queue or a device.

Queue and Device-specific Logical Names

Queue and device-specific logicals can only be defined as one or the other of these, but not both. For example, the logical:

```
SSV$queue_FOREIGN_PASSALL
```

cannot be defined as a device-specific logical and the logical:

```
SSV$device_LPD_COALESCE
```

cannot be defined on a queue-specific basis. Both, however, can be defined on a system-wide basis by dropping the "queue" or "device" specification:

```
$ DEF/SYS SSV$FOREIGN_PASSALL FALSE

$ DEF/SYS SSV$LPD_COALESCE FALSE
```

Not all queue and device-specific logicals can be defined on a system-wide basis in this way; but these are noted in the logical descriptions below.

18.1.2 Dynamic Versus Static Logical Name Definitions

All ScriptServer logical names are either dynamic or static.

Static Logicals

A static logical requires that the queue be stopped and restarted before the logical name definition can take effect. For device-specific static logicals, queues using the device must be stopped and restarted before the new logical name definition can take affect.

Dynamic Logicals

Dynamic logicals take effect immediately and do not require stopping and restarting of queues.

18.2 SCRIPTSERVER LOGICAL NAME DESCRIPTIONS

In the following table, "s" indicates a static logical, "d" indicates a dynamic logical. The logical name is given in its most specific form. You will want to look to the actual descriptions to see if it can be defined in a more general manner. Finally, the equivalence indicates the possible values to which the logical name can be equated.

Table 18-1:ScriptServer logical summary

Type	Logical Name	Equivalence
s	SCRIPTSERVER	directory containing Script-Server files
s	SSV$device_BACKCHANNEL	FALSE or device-name
d	SSV$queue_ASSUME_PCL	TRUE / FALSE
d	SSV$BLANKET_FONTS	font list
d	SSV$BLANKET_MODULES	module list
s	SSV$queue_CHARACTERISTICS	characteristic number(s)
s	SSV$CHECKPOINT_FREQUENCY	any number (pages)
s	SSV$device_COLOR	TRUE / FALSE
s	SSV$device_CTRLT_SUPPORT	TRUE / FALSE
d	SSV$queue_DEFAULT_DATA_TYPE	POSTSCRIPT / HPGL / ANSI / ASCII / PCL / AUTOMATIC / emulation
d	SSV$DEFAULT_SETTINGS	system-wide SCRIPT setting
s	SSV$DEFAULT_SIZES	INTERNATIONAL
s	SSV$queue_DEVINACT_INTERVAL	any number (seconds)
s	SSV$queue_DEVINACT_OPTION	FAIL / NOISY / WARNING / SILENT
s	SSV$DISPLAY_PRODUCTS	PostScript product names
d	SSV$queue_DOWNLOAD_MESSAGES	ALL / NONE / PERMANENT
d	SSV$EMULATION_DELAY	any value (seconds)
d	SSV$ENABLE_EXITSERVER	TRUE / FALSE
d	SSV$queue_ENVELOPE_COMMAND	PostScript envelope commands
s	SSV$device_EXITSERVER	PostScript password
s	SSV$queue_FLAG_PAPER_SIZE	PostScript paper selection command
s	SSV$queue_FLAG_TRAY_COMMAND	PostScript tray command

Table 18-1:ScriptServer logical summary (Continued)

Type	Logical Name	Equivalence
s	SSV$queue_FONTS	font list
S	SSV$queue_FOREIGN_DELETE	TRUE/FALSE
s	SSV$queue_FOREIGN_PASS	ALL TRUE/FALSE
d	SSV$FORM_form_SETTINGS	system-wide SCRIPT setting
d	SSV$queue_HPGL_COLORS	pen colors
d	SSV$queue_HPGL_WIDTH	width in terms of size expression
s	SSV$INHIBIT_DESCRIPTIONS	TRUE/FALSE
s	SSV$device_INHIBIT_DISPLAY	TRUE/FALSE
s	SSV$INHIBIT_PAGE_COUNTER	TRUE/FALSE
d	SSV$queue_JOB_SETUP_MODULES	module list
s	SSV$LIMITED_PRODUCTS	PostScript product name(s)
s	SSV$device_LPD_COALESCE	TRUE/FALSE
s	SSV$device_LPD_READ	TRUE/FALSE
s	SSV$queue_MODULES	module list
d	SSV$queue_PARAMETERS	parameter list
d	SSV$SYSTEM_PARAMETERS	parameter list
d	SSV$queue_PASSALL_MODE	TRUE/FALSE
s	SSV$queue_PCL_FLAG_COMMAND	PCL-based escape sequence(s)
s	SSV$device_PCL_SUPPORT	TRUE/FALSE/LIMITED/NOPJL
s	SSV$device_POSTSCRIPT_SUPPORT	TRUE/FALSE
d	SSV$queue_PRINT_BY_USER	TRUE/FALSE
s	SSV$queue_PRINTLOG_LEVEL	ALL/FATAL/NONE
d	SSV$queue_PS_CLUE	PostScript data recognition string
s	SSV$queue_REPOSITORY_PROTECTION	protection code
s	SSV$queue_REPOSITORY_UIC	[group,member]
d	SSV$SCRIPT_DEFAULT_QUEUE	any ScriptServer queue

Table 18-1:ScriptServer logical summary (Continued)

Type	Logical Name	Equivalence
d	SSV$SCRIPT_DEFAULTS	any SCRIPT setting
d	SSV$SCRIPT_OVERRIDE	HEADINGS/ PAGE_NUMBERING
d	SSV$SCRIPT_QUEUE_LOOKUP	INHIBIT/ALLOW
d	SSV$queue_SETTINGS	system-wide SCRIPT setting
s	SSV$SNAPSHOT	STREAM/TASK
s	SSV$device_TCPIP_PORT	any value (port#)
s	SSV$device_TCPIP_PROTOCOL	HP/IBM/QMS/TELNET/LPD
d	SSV$queue_TEXT_LANGUAGE	PCL
d	SSV$device_TRAY_tray-command	PostScript tray selection command or PCL-based escape sequence
s	SSV$TYPEFACE_DIRECTORY	directory specification

18.2.1 SCRIPTSERVER

Equivalence: **directory containing ScriptServer files**

Default Behavior: **SYS$SYSDEVICE:[SCRIPTSERVER]**

Type: **static**

Scope: **system-wide only**

This logical points to the directory containing the ScriptServer files. If this is a new installation, the installation will place those files in SYS$SYSDEVICE:[SCRIPTSERVER]. If this is an upgrade, the installation will locate the directory in which those files are found and install the new files in the same directory defining the SCRIPTSERVER logical name accordingly.

This logical name cannot be contained within the ScriptServer configuration file SSV$CONFIG.SSV. It is discussed in more detail in Chapter 2.

18.2.2 SSV$device_BACKCHANNEL

Equivalence:	**FALSE or device name**
Default Behavior:	**dependent on device type**
Type:	**static**
Scope:	**device-specific only**

Defining this logical to FALSE turns off return communication from the printer. This is necessary for printers that do not support return communications (i.e. printers connected via a parallel connection). Turning off backchannel disables dynamic queue characteristic definition, thus if characteristics are in use you will want to set characteristics manually using the logical SSV$queue_CHARACTERISTICS described below.

Defining this logical to a device name activates that device as an alternate serial backchannel for printers that do not support backchannel directly (i.e. printers connected via the DECnet connection).

This logical is discussed in context and in more detail in Chapter 4.

18.2.3 SSV$queue_ASSUME_PCL

Equivalence:	**TRUE/FALSE**
Default Behavior:	**TRUE**
Type:	**dynamic**
Scope:	**both**

The default value for this logical is TRUE, meaning that all jobs sent to the printer via a queue affected by this logical will be understood as PCL.

Defining this logical to FALSE, allows you to send non-PCL escape-sequence-based emulations to the printer while in software switching mode.

This logical is discussed in more detail in Chapter 4.

18.2.4 SSV$BLANKET_FONTS

Equivalence:	**font list**
Default Behavior:	**<none>**
Type:	**dynamic**
Scope:	**system-wide only**

This logical should be included in the ScriptServer file:

```
SCRIPTSERVER:DOWNLOAD.COM
```

Fonts included in the font list will be permanently downloaded to all ScriptServer queues. This logical cannot be contained within the ScriptServer configuration file, SSV$CONFIG.SSV and is discussed in more detail in Chapter 7.

18.2.5 SSV$BLANKET_MODULES

Equivalence:	**module list**
Default Behavior:	**<none>**
Type:	**dynamic**
Scope:	**system-wide only**

This logical should be included in the ScriptServer file:

```
SCRIPTSERVER:DOWNLOAD.COM
```

Modules listed in the module list will be permanently downloaded to all ScriptServer queues.

This logical cannot be contained within the ScriptServer configuration file, SSV$CONFIG.SSV and is discussed in more detail in Chapter 8.

18.2.6 SSV$queue_CHARACTERISTICS

Equivalence:	**characteristic number(s)**
Default Behavior:	**<none>**
Type:	**static**
Scope:	**queue-specific only**

Characteristics specified will be added to the list of characteristics set automatically by ScriptServer.

This logical is discussed in more detail in Chapter 12.

18.2.7 SSV$CHECKPOINT_FREQUENCY

Equivalence:	**any number (pages)**
Default Behavior:	**2**
Type:	**static**
Scope:	**system-wide only**

The value of this logical definition determines both the accuracy of the page counter in the ScriptServer queue description and how frequently the symbiont performs a check on the status of a print job.

This logical is discussed in more detail in both Chapter 4 and Chapter 12.

18.2.8 SSV$device_COLOR

Equivalence:	**TRUE/FALSE**
Default Behavior:	**FALSE**
Type:	**static**
Scope:	**device-specific only**

The value of this logical determines whether or not the symbiont will check the device in order to determine its color capability. Defining the logical to TRUE disables the check and automatically declares the device capable of color printing. By default, and if you define the logical to FALSE, the symbiont will query the device for its color capability setting the queue accordingly. Manual declaration of color capability might be necessary in situations without backchannel.

This logical is discussed in more detail in Chapter 4.

18.2.9 SSV$device_CTRLT_SUPPORT

Equivalence:	**TRUE/FALSE**
Default Behavior:	**TRUE**
Type:	**static**
Scope:	**device-specific only**

By default, in a bi-directional environment the symbiont will send a CTRL / T status request to a device upon connection and will not send a print job until an IDLE status is returned. Defining this logical to FALSE, disables this status request.

This logical is discussed in more detail in Chapter 4.

18.2.10 SSV$queue_DEFAULT_DATA_TYPE

Equivalence: POSTSCRIPT/HPGL/ANSI/ASCII/PCL/AUTOMATIC/emulation
Default Behavior: determined by file
Type: dynamic
Scope: queue-specific only

By default, the symbiont will search the file in order to determine data type. Defining the logical to one of the above possible data types disables that search and forces the symbiont to treat every file printed through the specified queue as if it is of the specified type.

This logical is discussed in more detail in Chapter 6.

18.2.11 SSV$DEFAULT_SETTINGS

Equivalence: system-wide SCRIPT setting
Default Behavior: <none>
Type: dynamic
Scope: system-wide only

The equivalence to this logical name must be a system-wide SCRIPT setting and determines the formatting values for all ASCII files printed with the PRINT command to PostScript printers as well as the default values for the SCRIPT menu.

This logical is discussed in more detail in Chapter 12.

18.2.12 SSV$DEFAULT_SIZES

Equivalence: INTERNATIONAL
Default Behavior: <none>
Type: static
Scope: system-wide only

Defining this logical to INTERNATIONAL changes the default units of measure to centimeters rather than inches and paper sizes to European rather than North American. This affects the default PAPER value in the Script menu as well as the default units of measure.

This logical is discussed in more detail in Chapter 3 and Chapter 13.

18.2.13 SSV$queue_DEVINACT_INTERVAL

Equivalence:	**any number (seconds)**
Default Behavior:	**120**
Type:	**static**
Scope:	**both**

This determines how long the symbiont will wait between attempts to connect to the printer based on the response to the CTRL/T status request.

This logical is discussed in more detail in Chapter 12.

18.2.14 SSV$queue_DEVINACT_OPTION

Equivalence:	**FAIL/NOISY/WARNING/SILENT**
Default Behavior:	**FAIL**
Type:	**static**
Scope:	**both**

By default, if the ScriptServer symbiont is unable to elicit a response from the printer after several CTRL/T status requests, the symbiont will abort the job with the completion status DEVINACT, device inactive.

This logical and these values are discussed in more detail in Chapter 12.

18.2.15 SSV$DISPLAY_PRODUCTS

Equivalence:	**PostScript product name(s)**
Default Behavior:	**<none>**
Type:	**static**
Scope:	**system-wide only**

This logical assumes bi-directional communication capability in order to recognize product names that might be included in this list. The logical name definition contains a list of product names that are capable of displaying PJL messages on an LED display. Product names should be defined in exactly the same way as appears on the status page from the printer, or by using a wild card. The wild card character (*) indicates that the product name must contain merely the string within the quotation marks to be recognized by the symbiont. For example the definition:

```
DEF/SYS SSV$DISPLAY_PRODUCTS “*HP LASER JET 4”
```

includes the entire family of Hewlett Packard Laser Jet 4 printers. The equivalence value is not case sensitive, but if the string includes spaces it must be contained within quotation marks.

A product having its name included in this list is the equivalent of having the SSV$device_INHIBIT_DESCRIPTION logical set to FALSE for that device, meaning that PJL messages will be displayed on the LED display of the printer.

This logical and these values are discussed in more detail in Chapter 4.

18.2.16 SSV$queue_DOWNLOAD_MESSAGES

Equivalence:	**ALL/NONE/PERMANENT**
Default Behavior:	**ALL**
Type:	**dynamic**
Scope:	**both**

When downloading permanent setup modules or fonts, the ScriptServer print symbiont sends messages to OPCOM with the name of the module or font being downloaded. By default, it does this on both job-specific and permanent downloads. By defining the logical to PERMANENT, messages will only be sent for those modules or fonts downloaded on a permanent basis. Defining it to NONE, disables all download messages.

This logical is discussed in more detail in Chapter 5 and Chapter 7.

18.2.17 SSV$EMULATION_DELAY

Equivalence:	**any value (seconds)**
Default Behavior:	**10**
Type:	**dynamic**
Scope:	**system-wide only**

Some PostScript printers require a period of time during which the printer sets up the emulation being invoked. On these printers, data transmitted during this period will be lost, either causing incomplete documents or failing to print at all.

To handle this situation, ScriptServer provides an emulation delay time during which the symbiont stops sending data to the printer giving it time to reset. The default emulation delay is 10 seconds. If this delay period is too long or too short, it can be adjusted through the definition of this logical name.

This logical is discussed in more detail in Chapter 6.

18.2.18 SSV$ENABLE_EXITSERVER

Equivalence: TRUE/FALSE
Default Behavior: FALSE
Type: dynamic
Scope: system-wide only

If a PostScript print file attempts to exit the server loop when the exitserver operator is disabled, the following message is generated by the printer and reported on the Print Log:

```
ScriptServer: Unauthorized attempt to execute exitserver
```

If this behavior is undesirable at your installation, you can enable access to exitserver for all users by defining a logical name as follows:

```
$ DEFINE/SYS SSV$ENABLE_EXITSERVER TRUE
```

NOTE: *This protection mechanism is a security feature of the ScriptServer software. There is at least one PostScript virus that exploits the fact that most PostScript printers have a password of 0 and can permanently damage the printer. Therefore, disabling ScriptServer's protection of the PostScript operator **exitserver** is strongly discouraged.*

This logical is discussed in more detail in Chapter 12.

18.2.19 SSV$queue_ENVELOPE_COMMAND

Equivalence: PostScript envelope commands
Default Behavior: 4.125x9.5envelopetray
Type: dynamic
Scope: queue-specific only

Specific envelope sizes are defined through the use of this logical name. If this logical name is not defined, the symbiont defaults to the envelope size 4.125 by 9.5 inches, unless the SSV$DEFAULT_SIZES logical is set to INTERNATIONAL in which case the default size will be given in centimeters.

This logical is discussed in more detail in Chapter 12.

18.2.20 SSV$device_EXITSERVER

Equivalence:	**PostScript password**
Default Behavior:	**0 (unless LN03 (see below))**
Type:	**static**
Scope:	**device-specific only**

The exitserver operator requires a password which is usually an integer. By default, the password is 0 (or in the case of the LN03R ScriptPrinter, it is the string "LN03R"). Use this logical to change that password.

This logical is discussed in more detail in Chapter 5.

18.2.21 SSV$queue_FLAG_PAPER_SIZE

Equivalence:	**PostScript paper selection command**
Default Behavior:	**Letter (A4 if INTERNATIONAL defaults are in use)**
Type:	**static**
Scope:	**queue-specific only**

This logical should only be used in conjunction with the tray-command logical in Section 18.2.22 *"SSV$queue_FLAG_TRAY_COMMAND*" on page 271. If left undefined, the paper-size logical will default to the default setting of the tray chosen.

This logical is discussed in more detail in Chapter 12.

18.2.22 SSV$queue_FLAG_TRAY_COMMAND

Equivalence:	**PostScript tray command**
Default Behavior:	**<default paper tray>**
Type:	**static**
Scope:	**queue-specific only**

This logical determines from which printer input paper tray the flag and print log pages will be taken. By default, these pages will be taken from the default paper tray.

This logical is discussed in more detail in Chapter 12.

18.2.23 SSV$queue_FONTS

Equivalence:	**font list**
Default Behavior:	**<none>**
Type:	**static**
Scope:	**queue-specific only**

This logical is contained within the file:

```
SCRIPTSERVER:DOWNLOAD.COM
```

and determines which fonts are to be permanently downloaded to specific queues. The download file is commented extensively on how to accomplish this procedure.

This logical cannot be contained within the ScriptServer configuration file, SSV$CONFIG.SSV. It is discussed in more detail in Chapter 7.

18.2.24 SSV$queue_FOREIGN_DELETE

Equivalence:	**TRUE/FALSE**
Default Behavior:	**TRUE**
Type:	**static**
Scope:	**both**

The default value for this logical is TRUE which sets the default protection for foreign queue temporary files to allow WORLD:DELETE. This prevents these temporary files from being retained in the ScriptServer directory.

Defining this logical to FALSE, disables the WORLD:DELETE privileges and may require that these temporary files be deleted manually.

This logical is discussed in more detail in Chapter 4.

18.2.25 SSV$queue_FOREIGN_PASSALL

Equivalence:	**TRUE/FALSE**
Default Behavior:	**TRUE**
Type:	**static**
Scope:	**both**

This logical is applicable to foreign queues only. Normally, when left undefined, the symbiont will pass print jobs to the execution queue with the /PASSALL PRINT qualifier. If defined to FALSE, the /PASSALL qualifier is not used.

This logical is discussed in more detail in Chapter 4.

18.2.26 SSV$FORM_form_SETTINGS

Equivalence:	**system-wide SCRIPT setting**
Default Behavior:	**<none>**
Type:	**dynamic**
Scope:	**system-wide only**

This logical allows you to use a setting-name in the symbiont library as the default for a specific form name.

This logical is discussed in more detail in Chapter 12.

18.2.27 SSV$queue_HPGL_COLORS

Equivalence:	**pen colors**
Default Behavior:	**see table 5-1**
Type:	**dynamic**
Scope:	**queue-specific only**

This logical is used to change the pen color assignments for printing HPGL files.

This logical cannot be contained with the ScriptServer configuration file SSV$CONFIG.SSV and is discussed in more detail in Chapter 6.

18.2.28 SSV$queue_HPGL_WIDTH

Equivalence:	**Width in terms of size expression**
Default Behavior:	**.1 inches (unless INTERNATIONAL defaults are in use)**
Type:	**dynamic**
Scope:	**queue-specific only**

This logical is used to change the pen width for HPGL files.

This logical cannot be contained with the ScriptServer configuration file SSV$CONFIG.SSV and is discussed in more detail in Chapter 6.

18.2.29 SSV$INHIBIT_DESCRIPTION

Equivalence:	**TRUE/FALSE**
Default Behavior:	**FALSE**
Type:	**static**
Scope:	**system-wide only**

By default, ScriptServer will display a printer description in the ScriptServer queue description. If this is undesirable at your institution, define this logical to TRUE.

This logical is discussed in more detail in Chapter 4.

18.2.30 SSV$device_INHIBIT_DISPLAY

Equivalence:	**TRUE/FALSE**
Default Behavior:	**TRUE**
Type:	**static**
Scope:	**both**

ScriptServer contains mechanisms that allow it to manipulate the printer’s LED display (when applicable). By default, ScriptServer will not do this, however, by defining this logical to FALSE, you can activate ScriptServer’s ability to manipulate the display. When explicitly defined to TRUE for a specific device, this logical takes precedence over the SSV$DISPLAY_PRODUCTS logical.

This logical is discussed in more detail in Chapter 4.

18.2.31 SSV$INHIBIT_PAGE_COUNTER

Equivalence:	**TRUE/FALSE**
Default Behavior:	**FALSE**
Type:	**static**
Scope:	**system-wide only**

By default, ScriptServer will display as accurately as the SSV$CHECKPOINT_FREQUENCY logical allows the number of the page being printed in the queue description during a print job. Defining this logical to FALSE will turn off that counter.

This logical is discussed in more detail in Chapter 4.

18.2.32 SSV$queue_JOB_SETUP_MODULES

Equivalence:	**module list**
Default Behavior:	**<none>**
Type:	**dynamic**
Scope:	**queue-specific only**

With this logical you can specify that a set of modules indicated by the module-list are to be sent to the printer before each print job to the queue specified.

This logical is discussed in more detail in Chapter 4 and Chapter 5.

18.2.33 SSV$LIMITED_PRODUCTS

Equivalence:	**PostScript product name(s)**
Default Behavior:	**<none>**
Type:	**static**
Scope:	**system-wide only**

This logical assumes bi-directional communications that allow ScriptServer to see the printer product name. Product names should conform to the specifications detailed in the discussion of SSV$DISPLAY_PRODUCTS above. The products listed within the definition of this logical name however will be considered to have limited PJL capabilities corresponding to the LIMITED implementation resulting from the logical name definition of SSV$device_PCL_SUPPORT discussed in Section 18.2.55 *"SSV$device_TCPIP_PROTOCOL*” on page 285.

This logical is discussed in more detail in Chapter 4.

18.2.34 SSV$device_LPD_COALESCE

Equivalence:	**TRUE/FALSE**
Default Behavior:	**FALSE**
Type:	**static**
Scope:	**both**

One of the TCP / IP protocol options becomes packet coalescence, where the send operation will be delayed to fill up the send buffer for more efficient I / O operations. Whether or not packets are coalesced is determined by the combination of the TCP / IP stack being used and the TCP / IP card on the printer. By default, packet coalescence is turned off.

This logical is discussed in more detail in Chapter 4.

18.2.35 SSV$device_LPD_READ

Equivalence:	**TRUE/FALSE**
Default Behavior:	**TRUE**
Type:	**static**
Scope:	**both**

Defining this logical to FALSE, results in a more flexible, but non-standard implementation of lpd. In this incarnation, ScriptServer won't read and wait for acknowledgments in the lpd protocol. You would normally only use the logical for debugging purposes.

This logical is discussed in more detail in Chapter 4.

18.2.36 SSV$queue_MODULES

Equivalence:	**module-list**
Default Behavior:	**<none>**
Type:	**static**
Scope:	**queue-specific only**

This logical is used to define a module or set of modules as queue setup modules. This logical cannot be included in the ScriptServer configuration file, SSV$CONFIG.SSV and is discussed in more detail in Chapter 5.

18.2.37 SSV$queue_PARAMETERS

Equivalence: **parameter(s)**
Default Behavior: **<none>**
Type: **dynamic**
Scope: **queue-specific only**

This logical is used to create a default set of PRINT parameters associated with a particular queue. If you wish to create a default set of PRINT parameters system wide, use the logical:

```
SSV$SYSTEM_PARAMETERS
```

Described in Section 18.2.51 *"SSV$queue_SETTINGS*” on page 283.

This logical is discussed in more detail in Chapter 10.

18.2.38 SSV$queue_PASSALL_MODE

Equivalence: **TRUE/FALSE**
Default Behavior: **FALSE**
Type: **dynamic**
Scope: **queue-specific only**

By default, the symbiont passes CTRL / D’s to the printer. This poses a problem on some printers that do not support the CTRL / D function. Defining this logical to TRUE means that ScriptServer will not pass CTRL / D’s to the printer.

This logical is discussed in more detail in Chapter 3.

18.2.39 SSV$queue_PCL_FLAG_COMMAND

Equivalence: **Paper tray and/or Paper size selection PCL-based escape sequence**
Default Behavior: **<none>**
Type: **static**
Scope: **queue-specific only**

This logical allows you to select a non-default paper tray for flag and trailer pages with PCL-based escape sequences. When defining this logical, you can use <ESC> to indicate For example, if legal paper is the desired paper size for flag and trailer pages from a QMS 2425 or HP

LaserJet 5SiMX printer that uses the lower tray as the default letter paper tray and the multipurpose tray for legal size paper, the logical definition would be as follows:

```
$ DEFINE/SYSTEM SSV$queue_PCL_FLAG_COMMAND “<ESC>&l2H<ESC>&l3A”
```

where “queue” is the name of the queue for which you want to affect flag and trailer page paper selection. In this logical definition, the characters following the ampersands and preceding the numbers 2 and 3 respectively are lower case L’s (el) and not the numeric digit “one”. Additionally within this example, the first escape sequence selects the multipurpose paper tray, the second sequence selects the legal paper size.

You can also use the following notation in this logical definition to indicate Escape Characters:

```
%xx
```

where “xx” is the hexadecimal value of a character. Values range from 00 (zero zero) to FF. For instance, in the escape sequence string in the previous example would be defined as:

```
%1b&12H%1b&13A
```

The whole command would read as follows:

```
$ DEFINE/SYSTEM SSV$queue_PCL_FLAG_COMAND "%1b&12H%1b&13A"
```

NOTE: *If you wish to include a percent sign (%) in the definition string using this latter notation, you must use the notation:*

```
%%
```

This logical is discussed in more detail in Chapter 12.

18.2.40 SSV$device_PCL_SUPPORT

Equivalence:	**TRUE/FALSE/LIMITED/NOPJL**
Default Behavior:	**FALSE**
Type:	**static**
Scope:	**device-specific only**

By default, ScriptServer will attempt to determine whether or not a printer supports the PCL printer language by using PJL commands. In situations where this is impossible (i.e. no back-channel), defining this logical to TRUE indicates that the device in question fully supports PCL and PJL commands.

The LIMITED definition of this logical is intended for use with PCL printers that fully support the PCL language, but only support a limited set of PJL commands that must include at least those corresponding to:

- the universal exit command;
- the enter language command; and
- PJL comments.

NOPJL indicates that PCL is a supported data type, but that they symbiont should send no PJL commands whatsoever to the printer in question.

NOTE: *This logical takes precedence over the SSV$LIMITED_PRODUCTS logical detailed above.*

The default value of this logical is LIMITED for any network queue in QMS protocol mode.

This logical is discussed in more detail in Chapter 4.

18.2.41 SSV$device_POSTSCRIPT_SUPPORT

Equivalence:	**TRUE/FALSE**
Default Behavior:	**TRUE**
Type:	**static**
Scope:	**device-specific only**

This logical should be used to indicate that a printer does NOT support PostScript in a uni-directional environment.

This logical is discussed in more detail in Chapter 4.

18.2.42 SSV$queue_PRINT_BY_USER

Equivalence:	**TRUE/FALSE**
Default Behavior:	**TRUE**
Type:	**dynamic**
Scope:	**both**

The “queue” in this logical name refers to the execution queue to which the ScriptServer queue is pointing. Defining this logical to FALSE, submits the print job to this execution queue without the /USER qualifier. Thus the job is submitted under the system account with all of its privileges, allowing compatibility with other manufacturers’ queues.

This logical is discussed in more detail in Chapter 4.

18.2.43 SSV$queue_PRINTLOG_LEVEL

Equivalence:	**ALL/FATAL/NONE**
Default Behavior:	**ALL**
Type:	**static**
Scope:	**both**

This logical determines under what circumstances a Printlog will be printed with a print job.

This logical is discussed in more detail in Chapter 12.

18.2.44 SSV$queue_PS_CLUE

Equivalence:	**PostScript data recognition string or PS-clue module**
Default Behavior:	**%!**
Type:	**dynamic**
Scope:	**queue-specific only**

According to the Adobe standard, %! is used to indicate that a file is PostScript. By default, the ScriptServer symbiont appends this to the front of all PostScript files sent to the printer, thereby indicating that the subsequent file is PostScript. Use this logical when the destination printer requires some other string to tell it that a file is PostScript. If this can be defined within a single string it may be defined as the equivalence of the logical:

```
$ DEFINE/SYSTEM SSV$queue_PS_CLUE “PostScript String”
```

If, however, you require that more text be sent to the printer than can be contained within a string, you can place the text within a file, insert the file into the ScriptServer Device Control Library just as you would a setup module, and call the file through this logical which will append the contents of the file called to the front of the file being printed. For example, if you want to append the contents of a file called:

```
POSTSCRIPT_MODULE
```

that has been inserted into the Device Control Library to the front of a PostScript file being sent to this particular clue, you can call the module with this logical simply by defining the name of the module preceded by “@” as the equivalence of the logical:

```
$ DEFINE/SYSTEM SSV$queue_PS_CLUE @POSTSCRIPT_MODULE
```

This logical is discussed in more detail in Chapter 2 and rules governing the use of setup modules are discussed in more detail in Chapter 5.

18.2.45 SSV$queue_REPOSITORY_PROTECTION

Equivalence:	**"(protection code)"**
Default Behavior:	**original file owner's Protection code**
Type:	**static**
Scope:	**both system-wide and queue-specific**

Defining this logical sets a mandatory Protection code for all files created by the ScriptServer repository queue.

NOTE: *This logical sets an absolute protection code and not a modification to the existing file protection. For example, if the current file protection is (O:RWED,G:RW) and you define the logical as:*

```
$ DEFINE/SYSTEM SSV$REPOSITORY_PROTECTION "(S:RW)"
```

the new protection code will be (S:RW) and not (S:RW,O:RWED,G:RW). It simply changes the protection code to the value associated with the logical.

Possible protection codes are as follows:

```
(S:RWED, O:RWED, G:RWED, W:RWED)
```

The codes preceding the colon are defined as follows:

- S = SYSTEM
- O = OWNER
- G = GROUP
- W = WORLD

The codes following the colon are defined as follows:

- R = READ
- W = WRITE
- E = EXECUTE

- D = DELETE

NOTE: *See your OpenVMS documentation for further details on possible Protection codes.*

18.2.46 SSV$queue_REPOSITORY_UIC

Equivalence: “	**[group,member]”**
Default Behavior:	**original owner’s UIC**
Type:	**static**
Scope:	**both system-wide and queue-specific**

Defining this logical sets a mandatory UIC for all files created by the ScriptServer repository queue. The “group” number can be any octal number from 0 to 37777 and the “member” number can be any octal number from 0 to 177777.

NOTE: *See your OpenVMS documentation for further details on UIC.*

18.2.47 SSV$SCRIPT_DEFAULT_QUEUE

Equivalence:	**any ScriptServer queue**
Default Behavior:	**SSV$PRINT (SYS$PRINT if that doesn’t exist)**
Type:	**dynamic**
Scope:	**system-wide only**

This logical is used to change the default queue in the SCRIPT facility system-wide. This logical is discussed in more detail in Chapter 9.

18.2.48 SSV$SCRIPT_DEFAULTS

Equivalence:	**any SCRIPT setting**
Default Behavior:	**<none>**
Type:	**dynamic**
Scope:	**system-wide only**

Use this logical to set the SCRIPT defaults system-wide. This logical is discussed in more detail in Chapter 9.

18.2.49 SSV$SCRIPT_OVERRIDE

Equivalence:	**HEADINGS/PAGE_NUMBERING**
Default Behavior:	**<none>**
Type:	**dynamic**
Scope:	**system-wide only**

Use of this logical allows the user to override the HEADINGS and PAGE_NUMBERING features of the Script Facility.

This logical is discussed in more detail in Chapter 9.

18.2.50 SSV$SCRIPT_QUEUE_LOOKUP

Equivalence:	**INHIBIT/ALLOW**
Default Behavior:	**ALLOW**
Type:	**dynamic**
Scope:	**system-wide only**

Defining this logical to INHIBIT, prevents the SCRIPT queue pop-up menu from displaying the available ScriptServer queues.

This logical is discussed in more detail in Chapter 9.

18.2.51 SSV$queue_SETTINGS

Equivalence:	**system-wide SCRIPT setting**
Default Behavior:	**<none>**
Type:	**dynamic**
Scope:	**queue-specific only**

This logical allows you to associate a particular SCRIPT setting with a particular queue by default.

This logical is discussed in more detail in Chapter 12.

18.2.52 SSV$SNAPSHOT

Equivalence:	**STREAM/TASK**
Default Behavior:	**<none>**
Type:	**static**
Scope:	**system-wide only**

With this logical you can increase the frequency with which ScriptServer creates its snapshot file. Defining the logical to STREAM will cause the symbiont to create this snapshot file every time a queue starts. Defining it to TASK will create the file every time a job starts.

This logical is discussed in more detail in Chapter 14.

18.2.53 SSV$SYSTEM_PARAMETERS

Equivalence:	**parameter(s)**
Default Behavior:	**<none>**
Type:	**dynamic**
Scope:	**system-wide only**

This logical is used to create a default set of PRINT parameters system-wide. To create a default set of PRINT parameters on a queue-specific basis, use the logical:

```
SSV$queue_PARAMETERS
```

discussed in Section 18.2.37 "*SSV$queue_PARAMETERS*” on page 277.

This logical is discussed in more detail in Chapter 10.

18.2.54 SSV$device_TCPIP_PORT

Equivalence:	**any value (port #)**
Default Behavior:	**23 (or based on SSV$device_TCPIP_PROTOCOL)**
Type:	**static**
Scope:	**device-specific only**

This logical allows you to change the port number of your TCP/IP device.

This logical is discussed in more detail in Chapter 4.

18.2.55 SSV$device_TCPIP_PROTOCOL

Equivalence: HP/IBM/QMS/TELNET

Default Behavior: TELNET

Type: static

Scope: device-specific only

This value of this logical determines which TCP/IP protocol will be used for you TCP/IP connection to that device.

This logical is discussed in more detail in Chapter 4.

18.2.56 SSV$queue_TEXT_LANGUAGE

Equivalence: PCL

Default Behavior: <none>

Type: dynamic

Scope: both

By default, ScriptServer will attempt to translate all text files to PS. If attempting to print the job to a PCL-only printer that ScriptServer can recognize, it will translate those text files to PCL. Use this logical if you wish to have all text files translated to PCL all the time.

This logical is discussed in more detail in Chapter 4.

18.2.57 SSV$device_TRAY_tray-command

Equivalence: PostScript tray selection commands or PCL-based escape se-quence

Default Behavior: <none>

Type: dynamic

Scope: device-specific only

This logical is used to select alternate trays for use when printing to the specified device. If the device is PostScript, the equivalence must be the appropriate PostScript tray selection command. If the printer is PCL, the equivalence must be the appropriate PCL-based escape sequence. Consult your Printer Documentation for details on the respective language commands.

This logical is discussed in more detail in Chapter 8.

18.2.58 SSV$TYPEFACE_DIRECTORY

Equivalence:	**directory specification**
Default Behavior:	**SCRIPTSERVER**
Type:	**static**
Scope:	**system-wide only**

Fonts are downloaded from the directory located by this logical name.

This logical is discussed in more detail in Chapter 7.

Chapter 19

Embedded Commands

ScriptServer allows you to embed commands in ASCII text files to change the parameters that control the formatting of the printed document. For these commands to take effect, you must enable embedded commands using one of the following methods:

- using the EC YES command in the SCRIPT facility;
- using the /EMBEDDED_COMMANDS qualifier when invoking the SCRIPT facility; or
- using the /PARAM=EMBEDDED_COMMANDS when using the PRINT command.

Embedded commands are lines that start with the backslash character (\). The general format is:

```
\command{=argument}
```

You may also embed PostScript commands in an ASCII text file. Embedding PostScript requires fairly extensive knowledge of the PostScript language, but provides the ability to customize your output in ways that would otherwise be unavailable.

The vertical bar character (|) is used to flag embedded PostScript commands, so these commands are referred to as bar commands.

Bar commands are enabled using one of the following methods:

- using the BC YES command in the SCRIPT facility;
- using the /BAR qualifier when invoking the SCRIPT facility; or
- using the /PARAM=BAR_COMMANDS when using the PRINT command.

19.1 EMBEDDED COMMANDS

19.1.1 BOTTOM_MARGIN Command

Format: \BOTTOM_MARGIN=size-expression

Default: \BOTTOM_MARGIN=.5in

This command is used to change the location of the bottom margin on subsequent pages. Whenever a new line is started, the bottom margin value is checked. If the new line would have a baseline below the bottom margin, a new page is started.

19.1.2 CENTER Command

Format: \CENTER

Default: none

The CENTER command is used to indicate that the following line in the file is to be centered between the left and right margins.

19.1.3 FONT Command

Format: \FONT=font-name {size-expression}

Default: \FONT=Courier 10pt

This command allows you to modify the current font, and optionally, its scale. If the size-expression is omitted, the new font is scaled to the current scaling value. When specifying fonts, you should use the full PostScript font name. If an incomplete name is used, the first font found that starts with the incomplete name is used. There is no test for ambiguous font specifications.

19.1.4 JUSTIFY Command

Format: \[NO]JUSTIFY

Default: \NOJUSTIFY

The JUSTIFY command allows you to enable / disable full justification on both margins. When justification is enabled, it forces full wrap mode. See Section 19.1.4 *"JUSTIFY Command*" on page 289 for more information on wrap handling.

19.1.5 LEFT_MARGIN Command

Format: \LEFT_MARGIN=size-expression

Default: \LEFT_MARGIN=.25in

This command allows you to change the location of the left margin. The change takes effect immediately.

19.1.6 RESTART_PAGE Command

Format: \RESTART_PAGE

Default: not applicable

The RESTART_PAGE command allows you to jump back to the first line given the current margin settings. This command is useful for starting a new column after modifying the current left and right margins.

19.1.7 RIGHT_MARGIN Command

Format: \RIGHT_MARGIN=size-expression

Default: \RIGHT_MARGIN=.5in

This command allows you to change the location of the right margin. The change takes effect immediately.

19.1.8 TABS Command

Format: \[NO]TABS

Default: \NOTABS

The TABS command allows you to enable / disable tab emulation. With the default setting of NOTABS, tabs are converted to the correct number of spaces to advance to the next tab stop.

With tab emulation enabled, tabs move the current print position to the next position as defined using the TAB_SIZE parameter.

19.1.9 VERTICAL_SPACING Command

Format: \VERTICAL_SPACING=size-expression

Default: \VERTICAL_SPACING=12pt

This command is used to modify line spacing.

19.1.10 WRAP Command

Format: \WRAP={ALL | LONG | NONE }

Default: \WRAP=NONE

The WRAP command is used to control how lines that exceed the right margin are handled. When WRAP is set to NONE, no line wrapping is performed. With WRAP set to LONG, lines that exceed the right margin are wrapped at the nearest word break. WRAP=ALL causes an entire paragraph to be treated as a long line, starting a new line at the nearest word break for each line. Paragraphs are delimited by two consecutive newlines.

19.1.11 Preventing Automatic Newline

At the end of every line in the text file, a newline operation is performed. Newlines move the current print position to the left margin on the next line as specified by the current vertical spacing setting. In some circumstances, you may want to insert an embedded command within a line without causing a newline to be performed. This is accomplished using the accent grave character (`). When used as the last character in a line of text, the grave character disables the automatic newline normally performed, preserving the current print position.

19.1.12 Restrictions

It is important to note that the ScriptServer software is not intended to be a word processor or a full-function typesetter. The capabilities provided by embedded commands are intended for low-level formatting needs that may otherwise require the use of word processing software.

The following restrictions exist with embedded commands:

- multiple fonts in a paragraph with full justification enabled is not supported
- tab emulation is not supported in paragraphs with full justification enabled

19.2 EMBEDDED POSTSCRIPT

The embedded PostScript capability allows you to introduce your own commands directly into the PostScript language stream generated by the ScriptServer print symbiont. As discussed earlier, embedded PostScript commands are flagged with the vertical bar (|) character as the first character in a line. The remainder of the line is the PostScript language code to be embedded, for example:

```
|gsave 100 100 moveto 500 100 rlineto stroke grestore
```

Several guidelines need to be followed for proper handling of embedded PostScript.

- The behavior rules outlined with regards to setup modules must be observed.
- As shown in the sample above, any modifications to the graphic state should be enclosed within a *gsave/grestore* construct so the graphic state changes are undone.
- Each page generated by ScriptServer is enclosed within a *save/restore* construct, so any variable or procedure definitions made in the embedded command will be undefined at the next page break.
- If you want to define a procedure or variable for use throughout the document, define them at the top of the file, then save them into ScriptServer's *save/restore* state (named SaveState) as shown in the following code fragment for the user-written procedure MyProc:

```
|/MyProc { ... } def

|/SaveState save def
```

Chapter 20

Self-Configuring TCP/IP Queues

One of the more time-consuming tasks when managing network printers is the configuration of TCP / IP queues. This problem is compounded by the fact that printers from various manufacturers use slightly different communications protocols and IP ports. By default, the ScriptServer print symbiont now attempts to establish a connection on a variety of the commonly used IP ports. When a connection is made successfully, it will automatically establish the communications parameters based on the port being used. This automatic configuration is accomplished using AUTO as the TCP / IP protocol, which is the default setting.

For example, on Hewlett-Packard LaserJet printers using the JetDirect interface, it will automatically establish a full-duplex connection on port 9100 with full backchannel and Control-T support. On QMS printers based on Crown technology, the ScriptServer symbiont will establish a TCP connection on port 35 (without backchannel) and a UDP connection on port 35 for printer status information.

To increase flexibility when configuring network queues, the ScriptServer symbiont also supports a new logical name for the default TCP / IP protocol. In previous releases, the default network protocol was TELNET on port 23, which is the standard port used for TELNET connections. You can now specify the default protocol for your site using the logical name SSV$TCPIP_PROTOCOL. For example, to establish HP as the default protocol, you would define the logical name:

```
$ DEFINE/SYSTEM SSV$TCPIP_PROTOCOL HP
```

Please note that by doing so, you will disable the self-configuration feature unless you have explicitly set AUTO as the protocol for a specific IP address. For example:

```
$ DEFINE/SYSTEM "SSV$PRINTER1.MYDOMAIN.COM_TCPIP_PROTOCOL" AUTO
```

If your site has a variety of printers from different manufacturers, it is recommended that you allow the new self-configuration feature to automatically determine which protocol to use.

20.1 PROTOCOL, BACKCHANNEL, AND CONTROL-T FALLBACK

When a queue is started on a TCP / IP address with AUTO as the protocol, the ScriptServer symbiont will attempt to establish a connection in the following order: HP, XEROX, QMS, TELNET, LPD. Once a connection has been established, the symbiont will communicate with the printer based on the port, which may automatically enable or disable the backchannel and / or Control-T support.

If, after several tries, the symbiont is unable to get a response from the printer using Control-T, Control-T support will be disabled automatically. If the printer will not respond to any commands at all after several tries, the backchannel will be disabled automatically.

Note that many printers equipped with network capability will communicate using more than one protocol. For instance, a LaserJet with JetDirect will communicate using both HP and LPD protocols.

Chapter 21

Repository Queues

Repository queues allow you to direct output from the ScriptServer print symbiont into an OpenVMS directory. One of the benefits of repository queues is that they provide an easy means to convert ASCII files to industry-standard PostScript format. To designate that a queue is a repository queue, simply specify the name of a directory (or a logical name pointing to a directory) as the argument to the /ON qualifier in the START/QUEUE or INITIALIZE/QUEUE DCL commands.

For example, suppose that you want to create a queue named REPOSITORY that directs ScriptServer output to the directory [REPOSITORY] on device DKA400. You could use the following command to create the queue:

```
$ INIT/QUEUE/ON=DKA400:[REPOSITORY]/PROCESSOR=SSV$SMB/LIBRARY=SSV$DEVCTL -
_$ REPOSITORY
```

Files that are converted to PostScript on this queue will be generated with the file type ".PS". For instance, if a user were to print the file MYLETTER.TXT to this queue, the ScriptServer symbiont would create the PostScript language file DKA400:[REPOSITORY]MYLETTER.PS.

With the exception that output is directed to a disk file, all ScriptServer logical names and behaviors apply to repository queues. For instance, in the example repository queue from above, if the logical name SSV$REPOSITORY_TEXT_LANGUAGE were defined as PCL, the ScriptServer symbiont would create a PCL file rather than a PostScript file.

If you would like to standardize the format of the PostScript output generated in repository queues, you could use the SPD_MGR.COM command procedure to associate a SCRIPT setting with the repository queue. See Chapter 12 of the ScriptServer Printing System User Guide for more information about establishing default settings for a queue.

21.1 CONTROLLING OWNERSHIP OF REPOSITORY FILES

ScriptServer software allows you to use UICs to control ownership of files in a ScriptServer repository. To establish the UIC (User Identification Code) that owns files created in a repository, use a logical name in the format SSV$queue_REPOSITORY_UIC, where queue is the name of the repository queue whose UIC you want to designate. For example, if you want to designate ownership of files in the repository queue named SSVREP to the UIC [100,5], you do so by defining the logical name:

```
$ DEFINE/SYS SSV$SSVREP_REPOSITORY_UIC [100,5]
```

Note: *Unless you specify otherwise, files in repository queue directories that are created by the ScriptServer print symbiont will be owned by the UIC associated with the SYSTEM account, which is [1,4] by default.*

21.2 CONTROLLING ACCESS TO REPOSITORY FILES

To control the access to files created in a repository, you specify the protection code of the files using a standard OpenVMS UIC-based protection specification. These are in the format “(S:rwed, O:rwed, G:rwed, W:rwed)”, where S, O, G, and W stand for System, Owner, Group, and World, and rwed represents the level of access allowed to each type of accessor: read, write, execute, and / or delete.

To establish the protection code of files created in a repository, use a logical name in the format SSV$queue_REPOSITORY_PROTECTION, where queue is the name of the repository queue whose UIC you want to designate. For example, if you want to set the protection code of files in the repository queue named SSVREP to (S:RWED,O:RWED,G:R,W), you do so by defining the logical name:

```
$ DEFINE/SYS SSV$SSVREP_REPOSITORY_PROTECTION ”S:RWED,O:RWED,G:R,W”
```

For more information about OpenVMS UIC-based protection, see the manual OpenVMS DCL Dictionary.

Chapter 22

Line Printer Daemon (LPD) Compatibility

This release of the ScriptServer Printing System includes a new fully-conforming implementation of the LPD printing protocol, referred to as "Robust" LPD mode. Since robust mode requires an intermediate spooling process, use this mode only if the standard LPD-compatible mode does not work with the printer or printing host.

22.1 BACKGROUND INFORMATION

Specifications for the Line Printer Daemon protocol are defined in the Request for Comments (RFC) 1179 memo. This memo can be obtained from the current edition of the "IAB Official Protocol Standards," or on the World Wide Web from the site located at *http://sunsite.auc.dk/RFC/*. LPD is a TCP-based protocol with a client / server architecture that allows host computers to print files either locally or to printers on remote network hosts. The protocol was originally developed and distributed for use with the UNIX operating system, but is now commonly available on many other platforms, including OpenVMS and Windows NT.

22.2 USING STANDARD LPD SUPPORT

ScriptServer will automatically use the LPD protocol by starting the queue on a device designated with the format:

```
remote_queue@tcpip_address
```

The remote queue is the name of the LPD printer running on the remote host indicated with the TCP/IP address. For example, to create a ScriptServer queue called PSPRINTER that sends print files to a printer named LaserJet5 that is running on a remote host at TCP/IP address 192.100.10.2, use the following command:

```
$ INITIALIZE/QUEUE/START/ON=”LaserJet5@192.100.10.2”/PROCESSOR=SSV$SMB -

_$ /LIBRARY=SSV$DEVCTL/DEFAULT=FEED PSPRINTER
```

Note: *The remote queue specification is case-sensitive, and must exactly match the actual LPD printer name from the remote host. Also, the Domain Name Service (DNS) host name can be substituted for the TCP/IP address.*

You can also create LPD queues by using the SSV$CONFIG command procedure located in the ScriptServer directory, and selecting option 1 (Create Queue) followed by option 6 (LPD Queue type). The procedure will prompt you for the correct device information. To run this procedure, type:

```
$ @SCRIPTSERVER:SSV$CONFIG
```

22.3 NETWORK-ATTACHED PRINTERS

Currently, many popular printers are available with a network card that allows them to directly connect to a TCP/IP network. Virtually all network cards or print servers support the LPD protocol through the use of an “internal” Line Printer Daemon that runs on the printer. These hardware implementations usually provide two or more remote queues that perform various manipulations on the file data, or specify alternate output ports on the device. GrayMatter Software recommends using a remote queue that will not alter the data stream because ScriptServer has already formatted the file. The remote queue names can be obtained from your printer card’s documentation by looking for “Setting up a printer on a UNIX system” or a related topic. If the documentation has an /etc/printcap file example, look for the value that appears after the :rp=. For your convenience, we have included a table of the most common network printer cards and the appropriate remote queue names.

Note: *QMS printers must have SPOOLING turned on in the TCP/IP sub-menu to prevent losing data.*

Table 22-1:Printer Cards and Queue Names

Manufacturer	Card Name	Remote Queue	Supported Printers
Hewlett-Packard[a]	JetDirect	raw	HP LaserJet series
XCD[1]	XjetIV	HP_PRINTER	HP LaserJet series
Emulex[1]	NETJet	PASSTHRU	HP LaserJet series
QMS	Crown	lp	QMS network printers
Dataproducts[1]	VPT	(user-defined)	Dataproducts LZR xx80
Digital Equipment Corp.[b]	N/A	postscript	DEC PrintServer series
Canon2	N/A	xjprint	Canon CLBP 360PS
Axis Communications	Axis 560 Print Server	PR1,PR2,or PR3	All printers (via serial and parallel ports)

a. These network cards also support a “raw” TCP/IP data port that allows ScriptServer to send the file directly without using the LPD protocol. This is generally preferred over LPD. See Section 4.2.5 *"Setting up a TCP/IP Connection*” on page 72.

b. These network printers require ScriptServer to run in Robust LPD mode. See below for details.

22.4 ROBUST LPD MODE

By default, ScriptServer LPD queues do not include the accurate byte count with the protocol data file header packets. This is an optional parameter within the LPD protocol, and ScriptServer is unable to provide this information because the amount of data in the file does not represent the exact amount sent to the printer. This is due to flag pages, resource accounting, data translation and embedded carriage controls.

Therefore, while running ScriptServer on an implementation of LPD that requires an accurate byte count, you will need to configure the queue for robust LPD mode before starting it. The following logical name causes ScriptServer to spool the output file locally first, and then send it with an accurate byte count to the destination queue on the remote host. This form of the logical name applies to all LPD queues on the system.

```
$ DEFINE/SYSTEM SSV$ROBUST_LPD TRUE
```

You may also use the logical to limit robust LPD mode to a specific queue.

```
$ DEFINE/SYSTEM SSV$MYQUEUE_ROBUST_LPD TRUE
```

This logical name needs to be defined for ScriptServer to send print jobs to most UNIX and Windows NT‰ LPD implementations.

22.5 USING PRIVILEGED PORTS

Some LPD server implementations require that the ports used by the LPD client (in this case, the ScriptServer symbiont) are in the privileged range, (less than 1024). Strict compliance with RFC 1179 requires that the client port numbers used are from 721 to 731 inclusive.

To configure a ScriptServer queue to comply with these requirements, you must define the logical name SSV$PRIVILEGED_LPD as TRUE. This logical name can be defined system-wide for all LPD print queues, as follows:

```
$ DEFINE/SYS SSV$PRIVILEGED_LPD TRUE
```

or it can be defined on a queue-by-queue basis. For example. to configure a queue that is spooling to a queue named lp on an LPD server at IP address host.mydomain.com, you must define the logical name as follows:

```
$ DEFINE/SYS “SSV$lp@host.mydomain.com_PRIVILEGED_LPD” TRUE
```

22.6 CONTROLLING THE LPD DATA TYPE

To describe how a print request is to be processed, the LPD specifications require a control file in addition to the print file. Among other things, the control file contains a one-character “data type” field. How this field is used to determine the way that the print file is processed varies from implementation to implementation.

In some cases, it is necessary to change the data type used by the ScriptServer symbiont from its default value of “f”. For example, the Windows NT LPD server requires a value of “l” (lowercase “L”.) You can define the logical name so that it applies to all ScriptServer LPD queues, such as:

```
$ DEFINE/SYS SSV$LPD_DATA_TYPE “l”
```

Note: *To preserve lowercase characters in logical names, you must enclose them in double-quotes.*

To configure a queue that is spooling to a queue named Printer at IP address NTsystem.mydomain.com, you must define the logical name like this:

```
$ DEFINE/SYS “SSV$Printer@NTsystem.mydomain.com_LPD_DATA_TYPE” “l”
```

To handle the requirements of particular files, you can also specify the LPD data type at the PRINT command level. For example:

```
$ PRINT/QUEUE=LPDQUEUE/PARAM=LPD_DATA_TYPE=“l”
  MYFILE.TXT
```

22.7 CONTROLLING SPOOL FILE LOCATION AND OWNERSHIP

When using robust LPD mode, you can control the location and ownership of the intermediate spooling files. The logical name SSV$SPOOL_DIRECTORY can be used to define an alternate location for spooled files. The default location is the SCRIPTSERVER directory. Note that this logical name also determines the location of spooled files for foreign queues.

You can also control ownership of the files spooled in robust LPD mode. By default, the system UIC under which the symbiont runs (usually [1,4]) is the owner of spooled files. By defining the logical name SSV$SPOOL_OWNER as USER, these files will be created as owned by the user that submitted the print job. This logical name is intended to address problems that may arise due to disk space limitations and quotas, and therefore does not affect foreign queues. To control file ownership for foreign queues, the logical name SSV$PRINT_BY_USER is consulted.

Chapter 23

Page Limit Support For PCL Files

Printers that fully support HP-PJL (printer job language) can use the page limit parameter to print a range of pages. Early implementations of PJL do not support the PJL page ranging command used by the ScriptServer print symbiont. An example of using the PAGE_LIMIT parameter to specify a range of pages to be printed is:

```
$ PRINT/QUEUE=HP4SI/PARAMETER=(“PAGE_LIMIT=(2,5)”) REPORT.LIS
```

To select page ranges on printers that support the PCL5 printer language, but not full HP-PJL, the use of an OpenVMS form is required. ScriptServer will use the /LENGTH qualifier value in the form printed with the job to set the number of lines per page. The ScriptServer symbiont assumes that the printer is properly configured for the number of lines per page that is specified on the form. The number of lines per page is often defined on PCL5 printers as the “FORM=##” setting. Please refer to your printer documentation for the correct setting. To ensure that your printer is set for the right number of lines per page, you may wish to create a setup module that sets the “Text Length” of the job and associate this module with the form. The following example uses 50 lines per page as the desired value.

Note: *The LINES_PER_PAGE print parameter does not support PCL files.*

Create a setup module that contains the appropriate PCL escape sequences to use the desired settings on the printer. To set the text length value to 50, create a new text file in an editor, and insert the escape sequence.

```
$ EDIT PCL50.PCL

<ESC>&l50F

<CTRL-Z>
```

Define the OpenVMS form to be used for the 50 lines / page job.

```
$ DEFINE/FORM/STOCK=DEFAULT/SETUP=PCL50/LENGTH=50 50LPP 99
```

Print the job with the form and the page limit parameter.

```
$ PRINT/QUEUE=HPQUEUE/FORM=50LPP/PARAM=(“PAGE_LIMIT=(3,6)”) FILE.PCL
```

Note: *The printer’s setting for the number of lines per page must be greater than or equal to the number of lines per page requested in the job, or the printer’s value is used. If you wish to override this functionality, you must include the escape sequences to adjust VMI and Margins in your setup module.*

Chapter 24

Miscellaneous Corrections & Enhancements

24.1 VERSION V5.0 MODIFICATION SUMMARY

Version V5.0-2

The following corrections and enhancements were made in version 5.0-2:

- Allowed page accounting to be disabled with logical name SSV$queue_DISABLE_PAGECOUNT. This increases throughput significantly with the backchannel enabled, particularly on HP LaserJets using the JetDirect network card.
- Added more checks for STOP/RESET during queue startup and ensured that the printer display is cleared if a STOP/RESET is detected.
- Added tolerance for link disconnect errors when writing data to a TCP/IP printer.
- Prevented invalid symbiont messages (“INVSMBMSG”) on STOP/NEXT commands.
- Created logical name SSV$device_NETWORK_IOERROR to modify the symbiont's behavior when encountering an I/O error on a network printer. The default behavior of the logical is that the job being processed during the failure will be REQUEUEed. Also send message to OPCOM that gives the reason the I/O failed.
- Fixed SSV$device_PCL_SUPPORT logical, was only working properly for "LIMITED" mode. Also added other miscellaneous PCL file type recognition and PCL characteristics adjustments, especially with LAT protocol.
- Fixed logicals SSV$DISPLAY_PRODUCTS and SSV$LIMITED_PRODUCTS.

- Modified device control library module handling so that it allows record lengths longer than 255 bytes. The new limit is now 1024 bytes.

Version V5.0-3

The following corrections and enhancements were made in version 5.0-3:

- Fixed the alignment of the first line of subsequent pages in ASCII files when using explicit form feed characters in a file with no carriage return carriage controls.
- Prevented the printing of “No Error” Print Logs, which occasionally happened when printing PCL files on QMS printers in QMS network mode. Also removed error messages that start with a non-printable character.
- Added support for a PostScript clue (PS_CLUE) module in the device control library. When using the logical SSV$queue_PS_CLUE, it can be defined as a module in the library by preceeding it with "@". The new default clue is “%!PS-Adobe". This can be used for compatibility with QMS Document Option Commands.
- Modified the symbiont so that there is no search for “data type” clues or examination of the file extension if printing using an explicit data type. Also added the logical SSV$ASSUME_PCL to allow escape sequence based emulations such as LN03.
- The symbiont now creates temporary files for foreign queues with the print job’s UIC rather than the system UIC. This helps with a change made in VMS version 6.1 that prevents the files from getting deleted.
- The symbiont now catches device command (DEVCMD) errors on files that reach end-of-file before we can process "flushing" messages from the PostScript interpreter.
- Flush the output buffer at the end of fax setup. This makes sure that the DialCallee and setpagedevice commands get sent to the printer before we start processing the file being transmitted.
- Prevented invalid I/O channel errors in QMS mode.
- If QMS protocol mode is set, the symbiont automatically sets PJL mode to LIMITED.
- Added compatibility for a PCL or PJL flag/trailer tray selection command for non-PostScript jobs. Also sends a PJL RESET command before sending file or flag/trailer page in full PJL mode. The (static) logical name SSV$queue_PCL_FLAG_COMMAND is used to control the command sent.
- Fixed a problem that occasionally caused the last page of the job not to print when translating from ASCII to PostScript.
- The symbiont now preserves case in SSV$CONFIG.SSV for logical values that are enclosed in double quotes.

- The symbiont now honors the SHEET_SIZE parameter regardless of what it thinks is available on the printer.
- Added support to allow logical name values to contain escape sequences. The format is “%XX” where XX is the hexadecimal value of the character. The <ESC> notation is also be accepted for escape characters. A percent sign needs to be indicated with %%.
- The symbiont now ignores the printer’s echo of UEL commands (<ESC>%-12345X.)
- The job’s page count is now displayed on job and file trailer page if there is a back-channel and page counting is not disabled. This required moving trailer pages (and reset modules) past printlogs and the routine that retrieves the final page count.
- Fixed NOTE qualifier on PCL flag pages.
- The symbiont no longer clears the HP LED display between files in a job or between copies of a file.
- The symbiont now sends the job reset module from the device control library after the file trailer and before the job trailer page. Also, if the module has "_PCL" in its name, it will only be sent to the printer if printing in PCL mode. The same is true for "_PS", which will only go to the printer for PostScript, Text (PostScript mode), or SCRIPT jobs.
- Device control library module handling has been modified so that carriage-return / line-feeds are not appended to each record if printing in PCL mode.
- Status messages from PJL printers are now handled properly. Job-related errors now appear only on Print Logs, and device related problems are reported to OPCOM.
- Files with the extension .EPS are now assumed to be PostScript language files.
- Entry numbers are now printed on file flag and trailer pages (for both PostScript and PCL.)

Version V5.0-4

The following corrections and enhancements were made in version 5.0-4:

- Allowed paper to be explicitly set even if the characteristic is not available on the queue. Also added A4, A5, and B4 to the /PAPER qualifier in the SCRIPT facility.
- Added LPD variables to symbiont snapshots. Also added the ability to take snapshots on a specific queue.
- Added the logical name SSV$INCLUDE_TRAILERS to keep trailers with the file rather than disconnecting and reconnecting on QMS Status1 printers. This prevents trailer pages from begin interlaced with jobs from other hosts.

- The symbiont now ignores PJL EOJ messages unless it is printing in PCL mode. This was causing “%SSV-W-WHATEOJ” messages if a another print processor (like some LaserJet drivers for Windows) placed PJL JOB and EOJ messages in the print file.
- Fixed problems with ASCII to PostScript conversion when using certain combinations of n-up printing, page limits, and duplexing.
- Added the new TCP/IP protocol XEROX.
- The symbiont now uses the correct /PARAM=DATA_TYPE when printing PCL files on foreign queues.

Version V5.0-5

The following corrections and enhancements were made in version 5.0-5:

- The symbiont now turns off unsolicited status messages on PJL printers correctly, instead of just at the end of PCL print jobs.
- Added the logical name SSV$LPD_DATA_TYPE to control the data type used in LPD control files. This can also be done at the job level using the print parameter /PARAM=LPD_DATA_TYPE=“x”, where x is the data type to be used.

Version V5.0-6

This release primarily improves reliability of LaserJet/JetDirect printers over bidirectional TCP/IP connections.

The following corrections and enhancements were made in version 5.0-6:

- The symbiont now uses a formal shutdown process to do TCP/IP disconnects from the printer.
- Timing parameters for printer status acquisition have been modified to decrease the likelihood of I/O timeouts, particularly with HP LaserJets using JetDirect network cards.
- A problem where use of the system-wide logical (as opposed to device-specific) for robust LPD could break non-LPD queues has been corrected.
- The logical names that disable CTRL/T (SSV$CTRLT_SUPPORT) and backchannel (SSV$BACKCHANNEL) can now be defined (as FALSE) on a system-wide basis.
- A problem where unsolicited status (USTATUS) messages on PJL-capable printers could confuse status acquisition has been corrected. This was reported as a device inactive (DEVINACT) error in prior releases.

Version V5.1-0

The following miscellaneous corrections and enhancements were made in the beta release of version 5.1 (V5.1-0):

- Two new command procedures are available to list queues by characteristics, and characteristics by queue. To view the characteristics associated with a queue use the procedure QUEUE_TO_CHARS as follows:

```
@SCRIPTSERVER:QUEUE_TO_CHARS queue-name [first-char] [last-char]
```

- where queue-name is the queue specification (wildcards are allowed) you want listed, and the optional first-char and last-char arguments specify the range of characteristic numbers you are interested in. For example,

```
@SCRIPTSERVER:QUEUE_TO_CHARS DORY 10 19
```

- will list all of the paper types characteristics associated with queue DORY.
- To view the queues that have a given characteristic, use the procedure CHAR_TO_QUEUES as follows:

```
@SCRIPTSERVER:CHAR_TO_QUEUES char-name
```

For example, to obtain a list of all queues that have the LEGAL characteristic associated with it, use the command:

```
@SCRIPTSERVER:CHAR_TO_QUEUES LEGAL
```

- The symbiont now supports a system-wide default TCP/IP protocol though the logical name SSV$TCPIP_PROTOCOL, whose default value is AUTO.
- Explicit port number definitions (using the logical name SSV$define_TCPIP_PORT) now properly override the port number for TCP/IP protocols HP and IBM, which was forcing it to 9100, and generating misleading OPCOM queue startup messages.
- The default printer configuration was changed from PostScript/ to PostScript/PCL/NoPJL.
- If the symbiont can't get the printer to talk to it on queue startup using PJL commands on a TCP/IP connection, it will now disconnect and reconnect before trying PostScript commands (which forces a reset on the printer).
- The symbiont now handles and interprets many more new PJL unsolicited status (USTATUS) messages, including messages associated with file systems (printer hard disks), paper jam locations, and external input and output devices, such as binders, staplers, scanners, and mailboxes.

- The symbiont now detects when a print job has been canceled from the printer control panel, and generates an appropriate Print Log indicating this has occurred.
- You can now specify the directory used for spooled files using the logical name SSV$SPOOL_DIRECTORY. The spool directory is used for both foreign queue and robust LPD spooling.
- You can now specify whether ownership of robust LPD spool files is the user that submitted the print job or the system using the logical name SSV$SPOOL_OWNER, which is defined as USER or SYSTEM. The default value is SYSTEM.
- Some fatal errors were causing the queue to stop without generating OPCOM messages. This problem has been corrected, and the snapshot file enhanced to capture more information on fatal snapshots.

Version V5.1-1

The following miscellaneous corrections and enhancements were made in the initial public release of version 5.1 (V5.1-1):

- The LPD_DATA_TYPE logical name handling was fixed so that it is also looked up as a device-specific logical (as documented)
- Unrecognized and/or unsupported print parameters are now reported on (nonfatal) Print Logs
- Handling of the PRINT_BY_USER logical name lookup was corrected. Previously, the first job printed on a queue was printed under the username SYSTEM
- The new logical name SUPPRESS_LF was added to allow control over where the first line of the first page lands. This fixes a problem with 2-byte print carriage control files that shift the first line of the first page down.

Version V5.1-2

In this release, particular attention has been paid to increasing job processing performance. In some configurations, as much as 20 to 30 seconds of elapsed time (per print file) has been eliminated due to optimizations in print job management.

The following miscellaneous corrections and enhancements were made in the second release of version 5.1 (V5.1-2):

- Added support for connection debugging when either the logical name SSV$VERBOSE_CONNECTIONS or SSV$queue_VERBOSE_CONNECTIONS is defined as TRUE

- Add a delay after LPD jobs according to logical SSV$device_LPD_DELAY. This improves reliability for those printer implementations that require a short delay at the conclusion of a print job before it is ready to accept another connection for a new job. The delay is expressed in seconds. In the testing done at GrayMatter Software, we have found that a delay of two to three seconds is adequate.
- The ScriptServer symbiont now supports the use of generic queue names in logical names that use queue names to manipulate software features. Since generic queue names are only known to the symbiont when a job is printed, this capability is restricted to dynamic logical names only. Refer to Appendix
- Occasionally in previous versions, TCP/IP connections would time out without giving the printer enough time to complete a connection, resulting in the need for re-establishing a network connection. This problem has been corrected.
- On some PCL language printers with only one input tray (notably the HP Color LaserJet 5M), the queue startup dialog may fail when the symbiont attempted to determine paper capabilities. This problem has been corrected.
- The number of queues supported by the SCRIPT facility queue pop-up menu has been increased from 300 to 750.
- The new logical name SSV$PS_INDICATOR has been implemented to allow a system administrator to specify what indicates to the symbiont that a file is PostScript. The default value is %!PS.
- The symbiont will no longer set the node an username on an HP display unless full PJL support is present.
- The symbiont now supports printing PostScript files with embedded binary data. In prior releases, the binary data could be interpreted as underlying control information, such as XON/XOFF, Control/C, and Control/T rather than as data. Symptoms of this problem could range from PostScript errors such as undefined and timeout to garbled bitmap images. By using the new print parameter OUTPUT_FILTER, this data is sent to the printer by way of a PostScript (Level 2) filter called ASCII85. To print a file using the new filter, append /PARAM=OUTPUT_FILTER=ASCII85 to the PRINT command, for example:

```
$ PRINT/PARAM=OUTPUT_FILTER=ASCII85 MYFILE.PS
```

Alternately, if a given queue will be printing PostScript files with embedded binary data on a regular basis, you can have the filter invoked using the queue-specific logical name that specifies default parameter, SSV$queue_PARAMETERS.

Note that using this output filter expands the amount of data sent to the printer by approximately 25%.

- You can now select the output tray to which a print job's output is directed using the print parameter OUTPUT_TRAY. To learn the names of the output trays to use, print the newly-distributed file SCRIPTSERVER:PAPER_HANDLING.PS. It will list all of the paper-handling capabilities of the printer it is printed on, including the names of each output tray. Note that PostScript tray names are case-sensitive, so the print parameter must be enclosed in quotes, for example:

```
$ PRINT/QUEUE=HP6/PARAM="OUTPUT_TRAY=Side" MYFILE.TXT
```

- Error messages reporting negative acknowledgements (NAKs) from LPD print servers now include the error code in the message.
- License support was expanded to handle the new AlphaServer 1200.
- The Network Console Agent (SSV$NCAGENT) was corrected so that rights and class identifiers are no longer case-sensitive.
- The Network Console Agent (SSV$NCAGENT) was corrected so that rights defined via the SSVNC_USER, SSVNC_OPERATOR, SSV$NC_CONTROL logical are properly asserted.

Chapter 25

Appendix A —Logical Name Summary

This section contains a revised list of all logical names used in ScriptServer software as of the release of version 5.1. In the following table, the type column indicates whether the logical name is static or dynamic. Static logical names require that the affected queue(s) must be restarted before taking effect. Dynamic logical names take effect the next time a print job is started on the queue.

Below the logical name is the section of the documentation or of this Addendum that discusses the use of the logical name in detail. Logical names that are tagged with an asterisk (*) denote that they are new, or are documented for the first time here.

Table 25-1:Logical Name Summary

Logical Name	Equivalence	Description	T
SCRIPTSERVER Section 2.3 "*After Software Installation*” on page 45	directory specification	This directory contains the files distributed with ScriptServer software that don’t have to be stored in SYS$SYSTEM or SYS$LIBRARY	s
SSV$queue_ASSUME_PCL*	TRUE / FALSE	When defined as TRUE, this logical indicates that when ScriptServer software is unable to determine the data type of a print file, it should assume that the data type is PCL.	d

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$device_BACKCHANNEL Section 4.2.1 *"Setting Up a Parallel Connection*" on page 66	FALSE or device name	Defining this logical as FALSE tells the ScriptServer symbiont that the printer is not able to transmit data to the host. This disables the ability to determine the printer's capabilities at queue startup and to get page counts for system resource accounting. Defining this logical as a device name indicates that the device is to be used as an alternate serial backchannel.	s
SSV$BLANKET_FONTS Section 7.3 *"Downloading Permanent Fonts*" on page 126	font list	Fonts in this list are permanently downloaded to all ScriptServer queues.	s
SSV$BLANKET_MODULES Section 5.6 *"Permanent Setup Modules*" on page 106	module list	Modules defined in this module list will be permanently downloaded to all ScriptServer queues.	d
SSV$queue_CHARACTERISTICS Section 12.7 *"ScriptServer Queue Characteristics*" on page 210	characteristic number(s)	Characteristics defined in this list will be added to the list set automatically by the ScriptServer sym-biont.	s

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$CHECKPOINT_ FREQUENCY Section 12.4 *"Check-pointing*" on page 208	pages between check-points	The value of this logical determines both the accuracy of the page counter in the ScriptServer queue description, and how frequently the symbiont check-points the job with the OpenVMS Queue Manager.	s
SSV$device_COLORS ection 4.2.1 *"Setting Up a Parallel Connection*" on page 66	TRUE / FALSE	Defines color capabilities. Use this logi-cal name on color printers without backchannels.	s
SSV$CONSOLEx* Chapter 1	network console specification(s)	This logical name is used to define the TCP / IP address of computers running the ScriptServer Network Console application, and the rights available to each machine.	s
SSV$device_CTRLT_ SUPPORT Section 4.2.2 *"Setting Up a Serial Connection*" on page 69	TRUE / FALSE	By default, on PostScript-capable print-ers with a back-channel, the symbiont will send a Control-T status request to the printer on connection and will not send a print job until the printer returns an IDLE status. By defining this logical as FALSE, Control-T status requests are disabled and the printer is always assumed to be idle.	s

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$queue_DEFAULT_DATA_TYPE Section 6.2.4 *"Defining Default Data Types*" on page 123	POSTSCRIPT / HPGL / ANSI / ASCII / PCL / AUTOMATIC / emulation	Normally, the symbiont scans the first few records of each print job to deter-mine its data type. Defining this logical to one of these data types disables that search and forces every file printed to be treated as that data type.	d
SSV$DEFAULT_SETTINGS Section 12.1 *"Establishing Defaults*" on page 203	system-wide SCRIPT setting name	This logical name must be a system-wide SCRIPT setting and determines the formatting values used for all ASCII files printed with the PRINT command to PostScript printers as well as the default values used in the SCRIPT menu.	d
SSV$DEFAULT_SIZES Section 3.7 *"Paper Sizes Supported*" on page 62	INTERNATIONAL	Defining this logical name as INTER-NATIONAL changes the default units of measure to metric values rather than North American. This affects the default PAPER value in the SCRIPT menu as well as the units of measure.	s
SSV$queue_DEVINACT_INTERVAL Section 12.5 *"Controlling Device Inactive Sensitivity*" on page 208	number of seconds	This logical name determines the length of the delay between attempts to connect to the printer based on the response to Control-T status requests.	s

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$queue_DEVINACT_OPTION Section 12.5 *"Controlling Device Inactive Sensitivity*" on page 208	FAIL / NOISY / WARNINGS / SILENT	This logical name determines how the symbiont handles the situation when the printer will not respond to Control-T status requests. The default value of FAIL causes the printer to abort the print job or queue startup with a "DEVINACT, device inactive" error.	s
SSV$queue_DISABLE_PAGECOUNT*	TRUE / FALSE	By default, the ScriptServer symbiont will obtain the page counter from the printer (when a backchannel is available) before and after each print job. On some printers, this slows queue proc-essing (most notably LaserJet printers using a JetDirect interface.) If your site does not require resource accounting with page counts, you can speed proc-essing by defining this logical name as TRUE.	s

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$DISPLAY_PRODUCTS Section 4.2.7 *"The Printer Display*" on page 73	PostScript product names	This logical name controls which printer products can display messages on the printer's display panel. When available, the symbiont will display the node and username of the current job during processing. Product names must be in exactly the same format as they appear on status pages from the printer, although wild-cards are allowed using asterisks. For example, defining this logical name as "*LASERJET 4*" will include the entire family of HP LaserJet 4 printers.	s
SSV$queue_DOWNLOAD_MESSAGES Section 5.7 *"Controlling Download Messages*" on page 108	ALL / NONE / PERMANENT	When downloading permanent setup modules or fonts, the ScriptServer symbiont sends messages to OPCOM about the download. By defining this logical name as PERMANENT, messages will only be sent for those downloaded on a permanent basis. To disable all download messages, define this logical name as NONE.	d

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$EMULATION_DELAY Section 6.2.2 "*Emulation Delay Time*” on page 122	number of seconds	On some printers, a period of time must elapse when switching between emulations. (This situation only arises on printers without automatic emulation switching.) This logical allows you to create a time delay when ScriptServer is switching emulations.	d
SSV$ENABLE_EXITSERVER Section 12.8 "*ScriptServer and the exitserver Operator*” on page 211	TRUE/FALSE	The exitserver operator is used to make persistent (across jobs until power cycle) changes in PostScript printers. By default, ScriptServer software pre-vents access to this operator by Post-Script language files. You can disable this protection mechanism by defining this logical name as TRUE.	d
SSV$queue_ENVELOPE_COMMAND Section 8.3.3 "*Customizing Envelope Handling*” on page 138	PostScript commands	This logical name is used to control the PostScript commands used by the sym-biont for printing envelopes.	d

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$device_EXITSERVER Section 5.6 *"Permanent Setup Modules*" on page 106	PostScript password	The exitserver operator is used to make persistent changes to the PostScript interpreter, such as permanent font downloads. This logical name allows you to specify a password when it is some-thing other than the default value of 0.	s
SSV$queue_FLAG_PAPER_SIZE Section 12.3 *"Controlling the Flag Page*" on page 207	page selection command	This logical name should only be used in conjunction with the tray-command logical below. If left undefined, the paper size will default to LETTER, or A4 if international defaults are in effect.	s
SSV$queue_FLAG_TRAY_COMMAND Section 12.3 *"Controlling the Flag Page*" on page 207	page selection command	This logical name defines a PostScript language command to select the paper tray from which flag, trailer, and print log pages are taken.	s
SSV$queue_FONTS Section 7.3 *"Downloading Permanent Fonts*" on page 126	font list	This logical name defines the fonts that are to be downloaded to the specified queue. It is defined in the command file SCRIPTSERVER:DOWNLOAD.COM.	s
SSV$FORM_form_SETTINGS Section 12.1 *"Establishing Defaults*" on page 203	system-wide SCRIPT setting name	This logical name is used to associate a setting name in the symbiont settings library with an OpenVMS form name.	d

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$queue_FOREIGN_PASSALL Section 4.3.1 *"Using the INIT/QUEUE Command*" on page 74	TRUE/FALSE	This logical name allows you to control whether jobs are passed to foreign queues with the PASSALL print quali-fier. The default value is TRUE.	s
SSV$queue_HPGL_COLORS Section 6.1.2 *"Pen Colors and Width*" on page 116	pen colors	This logical name is used to control pen color assignments when printing HPGL files. It must be a logical name; it cannot be defined in the configuration file.	d
SSV$queue_HPGL_WIDTH Section 6.1.2 *"Pen Colors and Width*" on page 116	pen width as a size expression	This logical name is used to control pen widths when printing HPGL files. It must be a logical name; it cannot be defined in the configuration file.	d
SSV$queue_INCLUDE_TRAILERS*	TRUE/FALSE	Defining this logical name as TRUE causes trailer pages to be sent with the file rather than disconnecting and reconnecting on QMS Status1 printers. This prevents trailer pages from being interlaced with jobs from other hosts.	s
SSV$INHIBIT_DESCRIPTION Section 4.5 *"Script-Server Queue Descriptions*" on page 91	TRUE/FALSE	By default, the symbiont will display printer and queue information in the queue description line. If this is not desirable in your organization, define this logical name as TRUE.	s

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$device_INHIBIT_DISPLAY Section 4.2.7 *"The Printer Display*" on page 73	TRUE/FALSE	By default, the symbiont will display host and username information on the printer's display panel (if supported by the printer.) If this is not desirable in your organization, define this logical name as FALSE. This logical name takes precedence over the SSV$DISPLAY_ PRODUCTS logical name.	s
SSV$INHIBIT_PAGE_COUNTER Section 4.6.1 *"Script-Server Queue Status Dialogue*" on page 93	TRUE/FALSE	Normally, the ScriptServer symbiont will display the number of pages printed in the queue description (to the level of accuracy determined by the logical SSV$CHECKPOINT_FREQUENCY.) Defining this logical name as FALSE will turn off that counter.	s
SSV$LIMITED_PRODUCTS Section 4.2.1 *"Setting Up a Parallel Connection*" on page 66	PostScript product names	This logical name allows you to specify a list of products that contain the limited PJL command set, such as the LaserJet 3Si. It assumes that a backchannel is available to respond to product name queries by the symbiont, otherwise use the SSV$device_PCL_SUPPORT logi-cal name to specify LIMITED support.	s

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$device_LPD_COALESCE Section 4.2.5 *"Setting up a TCP/IP Connection*" on page 72	TRUE/FALSE	This option allows you to control packet coalescence on LPD queues, where send operations are delayed to fill up the send buffer for more efficient I/O operations. By default this is FALSE, since record blocking is a SET QUEUE option.	s
SSV$queue_LPD_DATA_TYPE* V5.1 Chapter 22	LPD control file data type	This logical lets you define the LPD data type used in the control file sent to LPD servers. The default value used by ScriptServer software is “f”, but some LPD server implementations (such as Windows NT) require “l” (lowercase L) or another data type.	d
SSV$device_LPD_DELAY*V5.1 Chapter 22	number of seconds	Some printer imnplementations require a delay to reset the LPD server. This logical name specifies the delay in seconds between connections.	s
SSV$device_LPD_READ Section 4.2.5 *"Setting up a TCP/IP Connection*" on page 72	TRUE/FALSE	This logical name allows you to control whether the symbiont will read and process ACK/NAKs from the printer in LPD mode. For most implementations, it should be left at its default value.	s

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$queue_MODULES Section 5.3.4 *"Queue Setup Modules*" on page 102	module list	This logical name allows you to define a list of modules that are to permanently downloaded at queue startup. It must be defined as a logical name, not defined in the configuration file.	s
SSV$device_NETWORK_IOERROR*	REQUEUE/FAIL	This logical determines the behavior when a network connection is unexpect-edly disconnected. By default, the ScriptServer symbiont will requeue the job.	s
SSV$queue_PARAMETERS Section 10.3 *"Print/ Parameters Parameters*" on page 185	parameter list	You can use this logical name to define a set of default print PARAMETERS to be associated with a queue.	d
SSV$queue_PASSALL_MODE Section 3.4.1 *"THE SCRIPTSERVER PRINT LOG*" on page 57	TRUE/FALSE	By default, the ScriptServer symbiont will not continue to process PostScript jobs once it has received the message “Flushing: rest of job to end-of-file will be ignored” from the printer. However, some applications generate PostScript files containing multiple jobs where this is expected behavior. By defining this logical name as TRUE, PostScript files are sent in their entirety.	d

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$device_PCL_FLAG_COMMAND*	PCL command	This logical name defines a PCL com-mand to select the tray from which flag, trailer, and print log pages are taken.	s
SSV$device_PCL_SUPPORT Section 4.2.1 *"Setting Up a Parallel Connection*" on page 66	TRUE / FALSE / LIMITED / NOPJL	By default, the symbiont will attempt to determine whether or not a printer sup-ports the PCL printer language using PJL commands. In situations where this is not possible, (such as no backchannel) defining this logical as TRUE indicates that both the PJL command set and the PCL language are supported. Defining it as LIMITED means the printer can process the limited PJL command set (such as the HP LaserJet 3Si), NOPJL indicates that the PCL language is avail-able, but no PJL commands, and NONE indicate no PCL language support.	s
SSV$device_POSTSCRIPT_SUPPORT Section 4.2.1 *"Setting Up a Parallel Connection*" on page 66	TRUE / FALSE	By default, the symbiont will attempt to determine whether or not a printer sup-ports the PostScript language using Control-T commands. Where this is not possible, (such as without a back-channel) defining this as FALSE indi-cates that PostScript is not supported.	s

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$queue_PRINT_BY_USER Section 4.4.1 *"Creating SSV$CONFIG.SSV*" on page 87	TRUE / FALSE	By default, when print jobs are created on foreign queues, ScriptServer will submit the job under the original username. By defining this logical name as FALSE causes these jobs to be submit-ted under the SYSTEM username.	d
SSV$queue_PRINTLOG_LEVEL Section 3.4.1 *"THE SCRIPTSERVER PRINT LOG*" on page 57	ALL / FATAL / NONE	This logical name determines under what circumstances a Print Log is gen-erated. By default, it is generated whenever a message is received from the printer. Defining this logical name as FATAL will generate Print Logs only when a message is received followed by “Flushing: rest of job to end-of-file will be ignored.” Defining it as NONE disables the generation of Print Logs.	s
SSV$queue_PS_CLUE Section 3.2.1 *"Using the PRINT Command*" on page 55	PostScript string	The ScriptServer symbiont prefixes PostScript commands with the charac-ter string “%!” to clue printers with lan-guage sensing that the data is in Post-Script language. This logical name allows you to override this default value.	d

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$PS_INDICATOR ** Chapter 24	text string	By default, the ScriptServer symbiont identifies whether a file is PostScript by searching the file for the string “%!PS”. This logical name allows changing this indicator to support PostScript files that use a different character string (such as “%!”) to indicate PostScript content.	
SSV$queue_REPOSITORY_PROTECTION* Chapter 20	OpenVMS protection code	This logical name is used to specify the protection code of the files created by the repository queue.	s
SSV$queue_REPOSITORY_UIC* Chapter 20	OpenVMS UIC	This logical name is used to specify the UIC of the owner of files created by the repository queue.	s
SSV$queue_ROBUST_LPD* Chapter 4	TRUE/FALSE	By defining this logical name as TRUE, print jobs are spooled internally before begin sent to the LPD server to obtain accurate byte counts for the job. This is only necessary in certain environments.	s
SSV$SCRIPT_DEFAULT_QUEUE Section 9.2 *"Settings Libraries*” on page 145	ScriptServer queue	This equivalence of this logical name is used as the default queue name in the SCRIPT Facility menu. It may also be defined at the user process level.	d

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$SCRIPT_DEFAULTS Section 9.2 *"Settings Libraries*" on page 145	any SCRIPT setting	This logical name defines the default SCRIPT setting-name. It may also be defined at the user process level.	d
SSV$SCRIPT_OVERRIDE Section 9.2 *"Settings Libraries*" on page 145	HEADINGS / PAGE_NUMBERING	This logical name allows PRINT / HEADINGS and / PAGE_NUM–BERING to override the settings established in a SCRIPT setting.	d
SSV$SCRIPT_QUEUE_LOOKUP Section 9.1 *"USING THE MENU*" on page 142	INHIBIT / ALLOW	By defining this logical name as INHIBIT, you can prevent the SCRIPT facility from scanning all system queues to create its queue pop-up menu. This speeds processing on systems with large numbers of print queues.	d
SSV$queue_SETTINGS Section 9.2 *"Settings Libraries*" on page 145	system-wide SCRIPT setting name	This logical name allows you to associ-ate a setting in the symbiont settings library with a queue.	d
SSV$SNAPSHOT Section 22.4 *"Robust LPD Mode*" on page 299	STREAM / TASK / LPD / FATAL	The ScriptServer symbiont can generate snapshot files to debug printing prob-lems. Normally only generated when a fatal error occurs, you can use this logi-cal name to cause snapshots to be taken on STREAM (queue) startup and on TASK (job) startup, or for LPD job startup.	s

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$SPOOL_DIRECTORY* Chapter 22	OpenVMS directory	Determines the locations of files that are spooled for both foreign queues and robust LPD queues.	
SSV$SPOOL_OWNER*			
Chapter 22	SYSTEM / USER	Determines whether the system (UIC [1,4] by default) or the owner of the print job is the owner of files spooled in robust LPD mode.	
SSV$SYSTEM_PARAMETERS Section 10.3 *"Print/ Parameters Parameters*" on page 185	parameter list	You can use this logical name to define a set of default print PARAMETERS to be associated with all ScriptServer queues. This settings can be overridden with the queue-specific version above.	d
SSV$device_TCPIP_PORT Section 4.3.1 *"Using the INIT/QUEUE Command*" on page 74	port number	This logical name allows you to define the TCP / IP port to be used to establish the connection.	s
SSV$device_TCPIP_PROTOCOL Section 4.3.1 *"Using the INIT/QUEUE Command*" on page 74	HP / IBM / QMS / XEROX / TELNET / LPD / AUTO	This logical is used to control which TCP / IP protocol is used to communicate with a network printer. The default setting of AUTO causes the symbiont to scan through the supported protocols to determine which to use.	s

Table 25-1:Logical Name Summary (Continued)

Logical Name	Equivalence	Description	T
SSV$TCPIP_TIMEOUT*	number of seconds	This logical name is used to control how long ScriptServer will wait to receive a response to a connection attempt when using AUTO TCP/IP protocol. The default value is 15 seconds.	s
SSV$queue_TEXT_LANGUAGE Chapter 9	PCL/POSTSCRIPT	By default, the ScriptServer symbiont will translate text files to PostScript, if it is available. On printers supporting both PostScript and PCL, you can over-ride this behavior by defining this logical name as PCL. Note that many formatting features are not available in PCL mode.	d
SSV$device_TRAY_tray-command Section 8.3.1 *"Customizing Paper Tray Selection*" on page 137	PostScript tray selection command	This logical name allows you to specify the PostScript command to be used when selecting paper of a particular size.	d
SSV$TYPEFACE_DIRECTORY Chapter 7	directory specification	This logical name points to the directory containing PostScript fonts for downloading. By default, it points to the SCRIPTSERVER: directory.	s
SSV$queue_VERBOSE_CONNECTIONS* Chapter 24	TRUE/FALSE	This logical name is used to provide additional connection-related messages for troubleshooting network connections during queue startup.	s

Chapter 26

Appendix B — Troubleshooting

Version 5.1 of the ScriptServer Printing System is designed to simplify setting up and managing your print queues. There are two general categories of problems when managing ScriptServer queues: queue startup and file printing problems. Please take the time to consult this literature when attempting to resolve these problems before calling Technical Support.

When attempting to resolve software problems, it is important to get as much relevant information as possible before much progress can be made. The best clues to solving software problems comes in the form of error messages. ScriptServer software reports error messages in two ways:

1. If the error is caused by a configuration or other system-related problem, the error is reported to OPCOM, the OpenVMS Operator Communications facility. To receive OPCOM errors, you must enable them using the REPLY command (which requires the OPER privilege,) for example:

   ```
   $ REPLY/ENABLE
   ```

   Since ScriptServer errors are in the PRINTER category, you can restrict the messages that are sent to your terminal to just printer-related messages using the command:

   ```
   $ REPLY/ENABLE=PRINTER
   ```

   OPCOM messages from the ScriptServer symbiont have the general format:

   ```
   %%%%%%%%%%% OPCOM 12/16/96 11:35:30 AM %%%%%%%%%%%
   Message received from user SYSTEM on VMS-HOST
   %SSV-W-SSVID, ScriptServer error message
   %SYSTEM-F-SYSTEMID, OpenVMS error message
   ```

The first message line indicates the specific problem encountered by the ScriptServer print symbiont. The second line of the message generally provides the OpenVMS system error message associated with the problem.

2. If the error is related to processing the print file, the problem is reported through the print job Completion Status, which is visible using the /NOTIFY qualifier to the PRINT command. For example, is you are having a problem printing a file named MYFILE.TXT, to view any errors reported in the processing of the file you should print it with the command:

```
$ PRINT/QUEUE=SSVQUEUE/NOTIFY MYFILE.TXT
```

Note: *GrayMatter Software strongly recommends that you enable both OPCOM messages and completion status notification as shown above before attempting to resolve any printing problems you are encountering.*

26.1 TROUBLESHOOTING QUEUE STARTUP PROBLEMS

Queue startup problems are frequently the result of communications problems between the OpenVMS host and the printer. As you look for potential causes of queue startup problems, it is helpful if you review the following common situations where problems may occur:

- Installing a new printer for use with ScriptServer software – if you are installing another printer identical or similar to a printer already in use with ScriptServer software, review the logical names and / or configuration file (SCRIPTSERVER:SSV$CONFIG.SSV) settings for the existing printer.
- Changing a printer’s configuration – if you are modifying the way that a printer is set up, review the logical names and / or configuration file settings to verify that they are still applicable to the printer’s new configuration.

Using the SHOW QUEUE command (or the Network Console,) carefully watch the current status of the ScriptServer print symbiont during queue startup. As a ScriptServer queue starts up, it changes the queue description as shown in the following table.

Table 26-1:Queue Descriptions During Queue Startup

Queue Description	Progress of Queue Startup
<ScriptServer V5.1-1 Queue>	inital setting of queue description on startup
<ScriptServer V5.1-1 Queue [connecting]>	internal intialization is complete – the symbiont is now attempting to es-tablish a connection with the printer
<ScriptServer V5.1-1 Queue [starting up>	the symbiont has successfully connected and is beginning the startup dialog with the printer
<ScriptServer V5.1-1 Queue on product-name [startup dialog]>	the symbiont has determined the model name of the printer, indicating successful bi-directional communi-cation with the printer
<ScriptServer V5.1-1 Queue on product-name [checking fonts]>	the symbiont is verifying the fonts available on the printer, and checking for whether fonts need to be down-loaded
<ScriptServer V5.1-1 Queue on product-name [idle]>	queue startup is complete; it is now ready to process print requests

Table 26-2:Queue Startup Troubleshooting Chart

Symptom	Possible Cause	Solution
	Printer is offline	Verify that the printer is online, properly connected, and ready to accept print jobs.
	Serial port baud rate problems	Verify that the printer and the communications port are running at the same speed. Generally speaking, they should be set for the highest speed available on both the printer and port, such as 19200 baud or greater.

Table 26-2:Queue Startup Troubleshooting Chart

Symptom	Possible Cause	Solution
	Flow control problem	The flow control on both the printer and communications port should be using the same control protocol, which is either XON / XOFF (preferred in the OpenVMS environment) or DTR.
	LAT configuration problem	The terminal server port may not be configured properly. Review the settings established in LATCP and verify proper communication using SET HOST / DTE. See Section 14.3.1 *"Checking Serial/LAT Connections*” on page 231 of the ScriptServer Printing System User Guide.OPCOM Error message:DEVINACT, device inactive
This indicates the printer is not responding to status queries.	Parallel port configuration	In some environments (such as parallel ports on terminal servers,) a parallel connection may appear to be bi-directional serial connection to the symbiont. In this case you must indicate the lack of backchannel capability by defining the logical name SSV$device_BACKCHANNEL as FALSE. For example, if the device name is LTA1, you must define a command as follows: $ DEFINE / SYS SSV$LTA1_BACKCHANNEL FALSE

Table 26-2:Queue Startup Troubleshooting Chart

Symptom	Possible Cause	Solution
	Printer does not support Control-T status queries	Explicitly disable Control-T status queries. For example, if the printer is attached to LTA1, define the logical name as follows $ DEFINE/SYS SSV$LTA1_CTRLT_SUPPORT FALSE
	Printer does not support PostScript	Define a logical name that tells the symbiont that PostScript is not available. For example, it the printer is attached to LTA1, define it as follows: $ DEFINE/SYS SSV$LTA1_POSTSCRIPT_SUPPORT FALSE
	TCP/IP network address incorrect	Verify that printer is responding at the network address using the PING command, or by attempting to establish a TELNET connection.
Error message: TCPIPCONERR, TCP/IP connection error on queue	Incorrect TCP/IP protocol settings	Verify that you are using the correct protcol, port, and/or LPD queue name, or use the AUTO protocol to have the symbiont adapt automatically
	Subnet mask prevents connection	Ensure that the printer's subnet mask allows connection from the host. Class C networks should use 255.255.255.0, class B networks should use 255.255.0.0.

26.2 TROUBLESHOOTING PRINTING PROBLEMS

As you troubleshoot printing problems, look for common threads associated with problem files and/or printers. For example:

- are the same types of files (e.g. PostScript or PCL) always failing to print correctly on the same printer or type of printer?
- when you try to print a PostScript file with problems, make sure you get the error code (e.g. syntaxerror) generated if an error occurs. PostScript errors are described in Chapter 24 of the ScriptServer Printing System User Guide. If you are not getting a Print Log with the PostScript error on it, try using PRINT/SETUP=EHANDLER PROBLEM_FILE.PS to print the file with module that handles PostScript errors. Or, you can download the permanent error handler prior to printing the problem file using the command :

```
$ PRINT/QUEUE=queue-name SCRIPTSERVER:PERM_EHANDLER.PS
```

when a PostScript error is detected by the ScriptServer print symbiont, the job's Completion Status will be :

```
%SYSTEM-F-DEVCMD, device command error
```

- if you are not dealing with a PostScript error or file, there may be an error reported as the job's completion status, which is visible using the /NOTIFY qualifier, for example

```
$ PRINT/NOTIFY/QUEUE=queue-name PROBLEM_FILE.PCL
```

Table 26-3:Print File Troubleshooting Chart

Symptom	Possible Cause	Solution
PostScript error:VMerror	The PostScript interepreter has used all available memory, yet needs more to continue processing the job. This error frequently causes the printer to reboot	Reduce complexity of the document being printed, or make more virtual memory available to the interpreter. This can be done by installing additional printer RAM, adjusting VM allocation, and installing / allocating virtual memory on a printer hard disk.
PostScript error:ioerror	An i / o error has occurred; usually a flow control problem	The flow control on both the printer and com-munications port should be using the same control protocol, which is either XON / XOFF (preferred in the OpenVMS environment) or DTR.
any other PostScript error	A PostScript language error occurred, possibly due to an application error	Try to recreate the PostScript file using slightly different settings to see if the error recurs. See if error occurs without being spooled by ScriptServer using COPY or TELNET commands.
Completion Status:RNF, record not found	The settings record associated with a form, queue, or system could not be found	Verify that the setting stored in the symbiont settings library SCRIPTSERVER: SPD$LIRRARY.DAT (and associated by logical name) still exists using SPD_MGR.COM.
Completion Status:HANGUP, dataset hangup	A LAT connection was unexpectedly dropped	Verify that the LAT port is configured properly using the terminal server command port and / or the LATCP, the LAT control program

Table 26-3:Print File Troubleshooting Chart

Symptom	Possible Cause	Solution
Unusually long delays between print jobs	Unnecessary page resource accounting	If your organization does not require page accounting, you can speed processing using the SSV$DISABLE_PAGECOUNT logical name
	The SCRIPT facility was used to print a PostScript file	The SCRIPT facility is only used for ASCII to PostScript translation; if a PostScript file is printed using SCRIPT, it is treated as ASCII text and wrapped in PostScript.
PostScript language text is being printed	Using LPD protocol, the wrong control file data type is being used	Try using the PRINT / PARAMETER = LPD_DATA_TYPE="L". If successful, define it as a queue-specific logical name. See chapter 4 for details.
	No PostScript on printer, or printer configured with the PostScript language off	Verify that the printer is capable of printing PostScript and that it is software-enabled

Index