Performance Simulation System™ for OpenVMS User's Guide

Order Number: AD-130-PSS/3.4

This manual describes the installation and proper operation of both the Performance Simulation System and the External Performance Simulation System for VAX/VMS.

Revision/Update Information:	This is a revised manual. The EPSS manual no longer exists. It has been incorporated into this manual.
Operating System and Version:	VAX/VMS Version 5.4 or later
Operating System and Version:	OpenVMS Alpha Version 6.1 or later
Software Version:	PSS Version 3.4
Software Version:	EPSS Version 3.3

Advanced Systems Concepts
Parsippany, New Jersey

Third Printing: September 1995

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

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

ASCI, ASCI logo, PSS, EPSS, INTACT, LIBERATOR, PRISM, RECORDER, RQE, SHADOW, and WATCH are trademarks of Advanced Systems Concepts, Inc.

DIGITAL, VAX, VMS, FMS, TDMS, and PDP are trademarks of Hewlett Packard

Contents

INDEX

EXAMPLES

FIGURES

TABLES

Preface

The PERFORMANCE SIMULATION SYSTEM™ (PSS™) for OpenVMS, engineered by Advanced Systems Concepts, Inc., allows you to monitor and record terminal sessions and replay those sessions on one or more real or pseudo terminals. PSS provides three main services:

- Performance simulation
- Application Reliability and Testing
- Security

The EXTERNAL PERFORMANCE SIMULATION SYSTEM™ (EPSS™) for OpenVMS, engineered by Advanced Systems Concepts, Inc., allows you to replay sessions created by PSS from another computer system. In this manner the replay is free of the load imposed by the RTE software.

DOCUMENT PURPOSE

The *PERFORMANCE SIMULATION SYSTEM for OpenVMS User's Guide* describes the installation, the concepts and commands, and the proper operation of PSS and EPSS.

INTENDED AUDIENCE

The intended audience for this manual includes several potential groups of PSS users:

- The VAX System Manager
- The Security Administrator
- The PSS User

The **OpenVMS System Manager** is in charge of installing the software, ensuring that OpenVMS system resources are available, and incorporating the start-up procedure for the PSS product within OpenVMS.

The **Security Administrator** is in charge of overall, system security—necessary, because PSS requires preauthorization of all potential OpenVMS users *prior* to their using the product.

In addition this manual directly addresses the **PSS User** and provides detailed procedures for using the product.

DOCUMENT STRUCTURE

The *PERFORMANCE SIMULATION SYSTEM for OpenVMS User's Guide* consists of eleven chapters, an appendix, a glossary, and an index, as follows:

Section	Function
Chapter 1 OVERVIEW	Provides an overview of the PSS and EPSS products
Chapter 2 INSTALLATION	Itemizes the installation procedures for the PSS and EPSS products
Chapter 3 CONFIGURATION	Discusses the OpenVMS resources, required for proper operation of PSS/EPSS, as well as OpenVMS privileges and necessary quotas
Chapter 4 PRODUCT USES	Relates the many different ways in which you can use PSS
Chapter 5 USING PSS	Describes the use of the PSS product from a PSS user's point-of-view, discusses all the major PSS-user commands in detail, and illustrates their use
Chapter 6 USING EPSS	Describes the use of the EPSS product from an EPSS user's point-of-view, discusses all the major EPSS-user commands in detail, and illustrates their use
Chapter 7 FILE CONVERSION UTILITY	Explains the use of the PSS/CONVERT utility, which converts PSS internal-format files into ASCII editable files, as well as the PSS lexicals, which you can add to an input script file
Chapter 8 INTERACTIVE REAL-TIME MONITORING	Demonstrates the use of the PSS/EPSS real-time interactive monitoring facility, which also provides an interface for executing PSS/EPSS commands, on-line HELP, and the DCL SPAWN command

Chapter 9 PRODUCT SECURITY	Discusses the use of the PSS authorization and security system and—particularly for the system-security administrator—the techniques and processes for controlling access to PSS functions and terminals on your system
Chapter 10 RESTRICTIONS AND LIMITATIONS	Illustrates some of the internal mechanisms, used in PSS
Chapter 11 PRODUCT SUPPORT	Recommends the best methods for using PSS for the first time
Appendix A PSS MESSAGES	Lists all the messages that PSS might display
GLOSSARY	Defines PSS and related terms

CONVENTIONS

This document uses the following conventions:

- Square brackets ([ ]) indicate the enclosed item is optional.
- Uppercase indicates that you type text exactly as shown.
- Text in lowercase consists of data that you must supply—a device name, for example.
- All prompts and messages that the PSS and EPSS products display on your screen appear in small typewriter font, while all command syntaxes that you must enter appear in the same font, boldfaced, as follows:

```
This is a message or a prompt.
$ This is what you must enter.
```

- A comma, followed by a horizontal ellipsis (,...), indicates that you can supply more than one item, each separated by a comma.
- A vertical bar (|) indicates *one*, logical choice within a list, such as this *or* that.
- A [] indicates a key on your keyboard, for example, [Return].
- A vertical ellipsis indicates the omission of system-supplied information, as in the following example:

```
%VMSINSTAL-W-NOTSYSTEM, You are not logged in to the SYSTEM account.
%VMSINSTAL-W-DECNET, Your DECnet network is up and running.
%VMSINSTAL-W-ACTIVE, The following processes are still active:
   .
   .
   .
* Do you want to continue anyway? [NO]   YES  Return
```

SUMMARY OF TECHNICAL CHANGES

This release has the following new and changed features:

- VMS Version 5 and SMP Support
- A new start-up command procedure PSS_STARTUP.COM that replaces the previous procedure, STARTPSS.COM

  This procedure has been moved to SYS$STARTUP from SYS$MANAGER in accordance with V5 practice.

- The PSS command verb no longer requires a special SET COMMAND

  PSS command is now added directly to your system's DCLTABLES.EXE.

- A new pseudo-terminal, default-device name of PSUA

  Prior to V3.1, the default was PTA.

- Several new parameters, added to PSS_STARTUP.COM, which relate to the pseudo-terminal-device name and the number of pseudo terminals, generated for this system

  Prior to V3.1, users had to edit the STARTPSS.COM command procedure to change the device name or add additional pseudo-terminal units.

- New security authorization facility

  This system provides much more control and flexibility than the previous method of using a combination of VMS privileges and WORLD/SYSTEM passwords.

- Several new or enhanced lexicals:

  %CUR_LOOP - Substitute current value of %DO_LOOP

  %DO_LOOP - Iterate portion of input script

  %ENDDO_LOOP - Delimits extent of active %DO_LOOP

  %EXEC - New "local" parameters may be passed to input script

  %LOCAL_PARAM - Substitute value of local parameter passed in %EXEC

  %MAIL - Send message to VMS user-written program

  %RANDOM - Generate random number for value substitution in script

  %RANDOM_TIME - Generate random waiting times

  %SUSPEND - Suspend playback and wait for user input

  New PSS real-time interactive monitoring utility

- The inclusion of EPSS in one manual with PSS

- A total rewrite of this manual to provide clarity and consistency

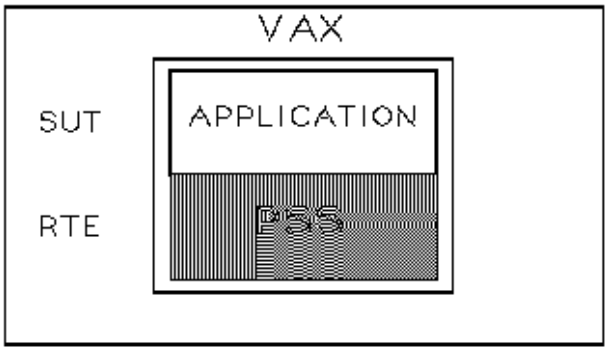
1 OVERVIEW

1.1 What Is PSS?

The **Performance Simulation System** (**PSS**) for OpenVMS allows a user to monitor and record terminal sessions and then replay those sessions on one or more real or pseudo terminals. You can categorize PSS as a **R**emote **T**erminal **E**mulator and the application to be tested, as the **S**ystem **U**nder **T**est, as depicted in Figure 1–1. PSS provides three main services:

- Performance Simulation
- Application Reliability and Testing
- Security

Figure 1–1 PSS on an OpenVMS System



1.1.1 Performance Simulation: Concepts and Terms

To begin recording a terminal session, enter the PSS/RECORD command. This commands PSS to record all input and output data, concerning a specified terminal, to disk. PSS creates separate files for input and output. Input characters that the terminal user enters are time stamped to simulate a realistic environment during a playback session. A conversion utility allows you to convert the input file from an internal-PSS format to the standard-ASCII format. This feature permits you to change the input "script" or to add special PSS lexicals for special-response and other handling.

To play back a terminal session, you enter the PSS command, PSS/PLAYBACK. This command instructs PSS to open the input script and begin to playback the session on the specified terminal. Input takes approximately the same time to replay, as it originally did to enter. You can play back sessions on real and/or pseudo terminals. A *pseudo terminal* is a software-created, terminal device. This feature allows you to test and measure your software against greater terminal loads than your system might physically have.

When the simulation finishes, PSS allows you to obtain a response-time statistics report for one or more PSS terminals. PSS computes the response time with *Service-Level* response measurement, as opposed to *Response-Level* measurement (compared in Table 1–1).

Table 1–1 Computing Responses

Type of Calculation	Interpretation
Service-Level response measurement	Begins with the termination of a READ request and ends with a new READ request
Response-Level measurement	Begins with the termination of a READ request, but ends when the first output data is sent to the user's terminal

Response-level measurements are subject to "placebo-type" interference (such as EDT's "WORKING" messages) and are not so discriminating as service-level measurements which determine the overall time the user waits to perform the next computer-related activity. In this manual, the phrase "response-time" refers to the service-level time computation.

Typically, PSS measures the response-time for each input entry which ends with a OpenVMS delimiter. However, PSS permits you to enter your own response-time checkpoints via the conversion utility. Thus, PSS can yield response-time measurements of special interest to you. During the playback, a response-time event file can be captured, which indicates the actual response-times seen by PSS for each input event. A a report can be produced which matches the input script with the response-time events so a detailed playback experiment can be produced and analyzed.

1.1.2 Application Reliability and Testing

During the recording phase PSS by default creates a file for all output data captured, per terminal. You can then issue the PLAYBACK command and capture the output of that session. By comparing the two files for differences (via DIF), you can generate an automated-audit trail. This proves particularly useful for project work in which you must ensure regression testing and reliability before implementing your system.

1.1.3 Security

Since PSS records all of the input and/or output data of any selected terminal session, an interesting application of this product lies in its system security. You can use PSS to monitor both local, dial-in, LAT, and X29 terminal sessions. This feature allows you to monitor highly sensitive applications for unauthorized access. In addition by monitoring the OpenVMS-system console you have a complete log of system events, including anything the operator enters. This feature is useful for monitoring operator interaction on production machines, when a complete record of events is necessary for problem determination.

1.1.4 PSS Authorization

PSS can be a very powerful tool, when used correctly by authorized users. PSS comes with its own authorization system which does not rely soley on OpenVMS privileges for product use. The PSS Authorization facility allows two classes of users: Administrators and Users.

An **Administrator** is one who determines the type of access an individual can have, concerning PSS, as well as the creation of other PSS administrators.

A **User** is an individual who has been authorized to enter various PSS commands, based on his/her authorization level.

In this manner a OpenVMS user who is not authorized cannot use the PSS product, regardless of the type of OpenVMS privileges held.

1.2 What is EPSS?

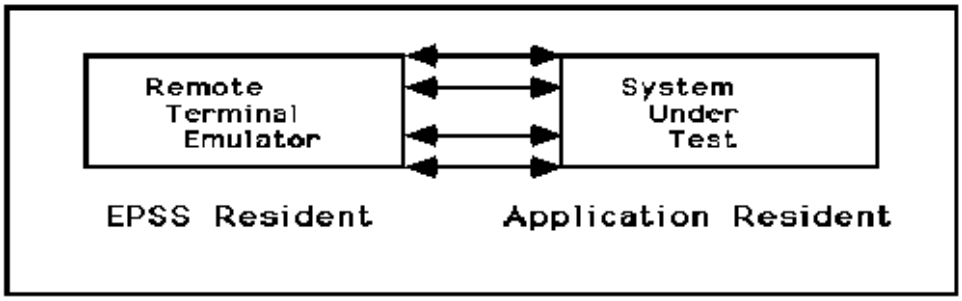
External PSS (or EPSS) is a related and separately-licensed product to PSS (Performance Simulation System), engineered by Advanced Systems Concepts. PSS provides performance simulation facilities on a single OpenVMS system, EPSS provides RTE facilities on a physically distinct OpenVMS system from the one you want to test.

This manual uses two terms to indicate the system on which EPSS resides and the system to which EPSS directs its playbacks. *RTE* refers to an OpenVMS system that provides **R**emote **T**erminal **E**mulation. This is the system that EPSS runs on.

SUT refers to the **S**ystem **U**nder **T**est. Your application program(s) will reside in this system.

By providing separate OpenVMS systems for RTE and SUT use, EPSS can perform a technically pure simulation (Figure 1–2). To determine CPU and other resource use, you can use Digital's SPM and VPA tools. You can use EPSS to provide the volume-loading facility and also to measure the service times that the RTE-user population encounters.

Figure 1–2 SUT and RTE Relationship



2 INSTALLATION

2.1 Installation of PSS

2.1.1 Release Notes

PSS provides online release notes that you can display or print.

They are available from the VMSINSTAL procedure if you use the N option. VMSINSTAL can print and/or display the release notes.

After installation the release notes will be in the file, SYS$HELP:PSS034.RELEASE_NOTES, as Example 2–1 illustrates. In addition online help also provides the location of the release-notes file.

Example 2–1 Using Help to Find PSS Release Notes

```
$ HELP PSS RELEASE_NOTES

PSS

  Release_notes

     The release notes for PSS V3.4 have been placed in the file
     SYS$HELP:PSS034.RELEASE_NOTES.
```

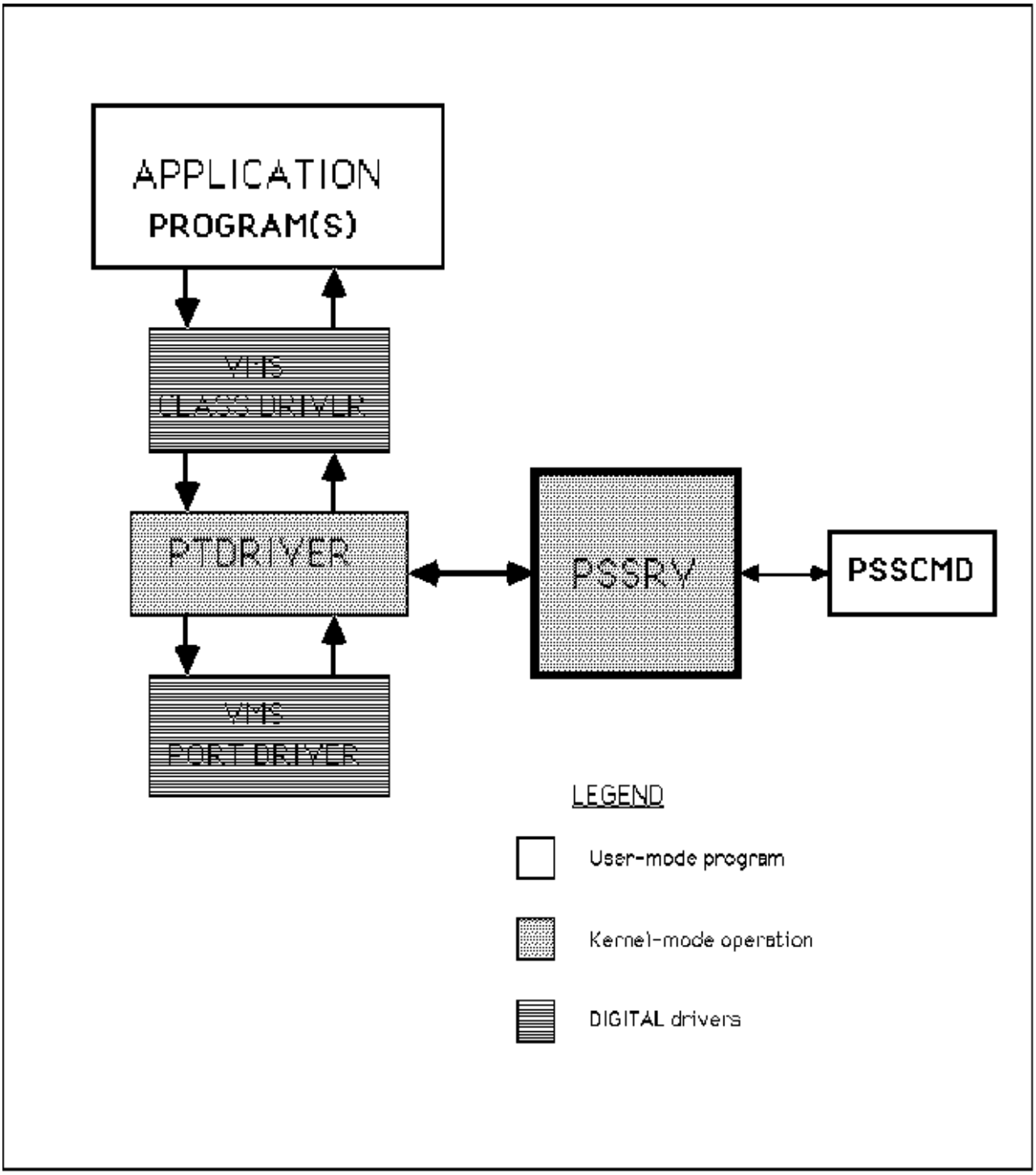
2.1.2 Contents of the Distribution Kit

PSS Product Checklist

- PSS User's Guide
- PSS Distribution Media
- PSS Product Information
- PSS Software Product License Agreement
- PSS Product Warranty Card

If the contents of this kit are incomplete, please contact Advanced Systems Concepts, Inc., Hoboken, N.J. or your local distributor.

Figure 2–1 PSS Component Flow



2.1.3 Before You Install PSS

PSS consists of five main components—-*PTDRIVER, PSSRV, PSSCMD, PSVCMD,* and *PSS_MONITOR*, as shown in Figure 2–1. The first two of these components require system resources which PSS must have available for proper product operation.

PTDRIVER

This VMS driver performs two major functions.

1. PSS uses this component during recording to intercept the input and/or output data of a specific terminal. In record mode PTDRIVER does not act as a driver, but serves instead as a collection of routines that execute in kernel mode and collect recorded data for PSSRV.

2. In playback mode PSS makes the driver aspects of PTDRIVER apparent, as it provides support for a new "pseudo" terminal device, named PSU. This feature of PSS allows you to perform simulated terminal loading.

PSSRV

This component performs processing in both kernel and user modes and writes all collected data (via PTDRIVER) to one or more files per terminal.

PSSCMD

This component performs all DCL command processing for the PSS product with the exception of the /CONVERT and /MONITOR commands.

PSVCMD

This component performs PSS file conversion. PSS uses a special type of formatting for all of its files, which it must convert through PSVCMD, to ASCII for use by an editor.

PSS_MONITOR

This component provides a real-time interface for monitor and control of PSS operations.

2.1.4 Installation Checklist

Check that you have each item in the following list before you start the installation:

1. A valid version of the VMS operating system (see Section 2.1.5)
2. A backup of your system disk (see Section 2.1.6)
3. Required privileges, quotas, disk space, and time (see Section 2.1.7)
4. A working familiarity with the VMSINSTAL procedure (see Section 2.1.8)

2.1.5 Prerequisite Software

PSS requires that a valid VMS Version 5.4 or 5.x(-n) operating system be installed on your system before installing PSS. To determine what version of VMS you have, type the following command:

```
$ SHOW SYSTEM

VAX/VMS V5.4  on node PSS  1-DEC-1990 11:33:54.06   Uptime    6 22:25:03
  Pid    Process Name    State  Pri      I/O       CPU       Page flts Ph.Mem
```

The first line of the displayed information shows the VMS Version.

VMSINSTAL checks for a valid version and aborts the procedure, if it does not find one.

2.1.6 System Disk Backup

ASCI recommends that you take the precaution of backing up your system disk before any installation. See the *OpenVMS System Manager's Reference Manual* for instructions on backing up your system disk.

2.1.7 VMSINSTAL Requirements

ASCI recommends that you run the VMSINSTAL procedure from the SYSTEM account.

The account you use for the installation must have the following privileges:

- CMKRNL
- TMPMBX

The account you use for the installation must have the following quotas:

- ASTLM = 24
- BIOLM = 18
- BYTLM = 18000
- DIOLM = 18
- ENQLM = 30
- FILLM = 20

PSS requires a net utilization of at least 500 free-disk blocks on your system disk.

The installation takes about 5 minutes from magnetic tape, operating on a VAX 8250.

2.1.8 The VMSINSTAL Procedure

VMSINSTAL is a Digital-supplied, command procedure, used for installing software products from their distribution media to your system. To invoke the procedure, issue the following command:

```
$ @SYS$UPDATE:VMSINSTAL  product  device  OPTIONS options_flag
```

You can invoke the command procedure, VMSINSTAL, without using any of the shown parameters. If so, the procedure prompts you for the necessary information.

Table 2–1 VMSINSTAL Parameters

Parameter	Description
product	The product name and version—in this case, PSS034 This parameter is optional.
device	The device where you mount the distribution volume The format is ddcu, when dd is the device code, c is the controller, and u is the unit number. You can use a logical name that translates to a valid device specification. This parameter is optional.
OPTIONS	The presence of this keyword parameter that indicates whether you have used the options_flag parameter to specify any options This parameter is optional.
options_flag	A letter which specifies an option The only valid option for this installation is N, which allows you to print and/or view the release notes before the installation continues. This parameter is optional and must follow the keyword, OPTIONS.

The PSS installation process uses the standard VMS installation procedure, VMSINSTAL. PSS is normally distributed on 9-track, 1600-bpi magtape in BACKUP format.

The next section documents the installation procedure, step by step. Future versions of VMSINSTAL might contain new prompts that are not documented here; however, the installation should proceed normally.

2.1.9 Performing the Installation

Step 1: Log In

Log into the System Manager's account (SYSTEM) and use the operator's console terminal. Make sure that you do have all of the items, listed in the checklist in Section 2.1.4.

```
Username:  SYSTEM     Return
Password:             Return
```

Step 2: Run VMSINSTAL

Invoke the installation command procedure as follows:

```
$ @SYS$UPDATE:VMSINSTAL PSS034 $2$MUA0: OPTIONS N     Return
```

PSS034 is the product name and version.

2MUA0: represents a valid device name on which you mount the distribution media. If the media is a disk, use the directory and device names.

OPTIONS N indicates that you want VMSINSTAL to prompt you for the Release Notes options.

If you do not specify a product or device, the system prompts you for it.

If you do not want to type or print the release notes before the installation, omit OPTIONS **N**. After the installation you can find the release notes in the SYS$HELP directory.

To abort the installation, type CTRL Y. The installation procedure then deletes any files you have created up to that point and exits. To restart the installation, proceed with Step 2:.

Step 3: Check Warnings

VMSINSTAL might display any or all of the following warning messages and ask if you want to continue.

```
%VMSINSTAL-W-NOTSYSTEM, You are not logged in to the SYSTEM account.
%VMSINSTAL-W-DECNET, Your DECnet network is up and running.
%VMSINSTAL-W-ACTIVE, The following processes are still active:
   .
   .
   .
* Do you want to continue anyway? [NO]   YES  Return
```

If you choose not to continue, you return to the DCL level. Correct the warning condition, and restart with Step 2:, above.

Step 4: Create a Backup

If you have not backed up your system disk, do not continue with the installation. If you need to back up the system disk, type NO. VMSINSTAL exits and returns you to the DCL prompt. Backup your system disk and start VMSINSTAL at Step 2:, above.

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

Step 5: Load the Distribution Media

VMSINSTAL asks you to load the distribution media onto the device.

```
Please mount the first volume of the set on $2$MUA0:.
* Are you ready? [YES]   YES  Return
%MOUNT-I-MOUNTED, PSS034 mounted on _$2$MUA0: (HSC000)
```

Step 6: Select the Release Notes Option

This step applies only if you specified the OPTIONS **N** in Step 2:.

```
Release Notes Options:

 1.  Display release notes
 2.  Print release notes
 3.  Both 1 and 2
 4.  Copy release notes to SYS$HELP
 5.  Do not display, print or copy release notes

* Select option [2]:   2    Return
* Queue name [SYS$PRINT]:   SYS$PRINT    Return
Job PSS034 (queue SYS$PRINT, entry 1117) started on SYS$PRINT

* Do you want to continue the installation? [YES]
   YES    Return
```

Option 1 immediately displays the release notes at your terminal. Option 2 prompts you for a print queue and spools the release notes for printing. Option 3 does both. Option 4 copies the release notes to the SYS$HELP directory, and Option 5 prevents VMSINSTAL from displaying, printing, or copying them.

```
* Queue name [SYS$PRINT]:
```

Then VMSINSTAL asks, if you want to continue the installation:

```
* Do you want to continue the installation? [YES]
```

Step 7: Purge Existing Files

The PSS system creates a new generation of these system files during installation:

- SYS$LIBRARY:DCLTABLES.EXE
- SYS$STARTUP:PSS_STARTUP.COM

VMSINSTAL displays the following prompt, asking whether you want to purge old versions of these files:

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

Step 8: Product Security

If this is the first installation of PSS V3.4 (or later) on this system, the following query appears:

```
Do you want to disable PSS security [NO]:
```

Type YES if you want to effectively disable the PSS security system. Type NO (the default) to leave the security system enabled. ASCI highly recommends that you answer *NO* to this query

Note: Regardless of how you answer this question, an administrator record for the SYSTEM account is created (the password is ACCESS). ASCI strongly recommends that you change this password (or delete the administrator record) as soon as possible. If you entered YES to this security query, the installation procedure issues the following command:

```
$ PSS/AUTH/SEC/OPER=ALL/TYPE=T/OBJECT=* ADD *
```

This command allows everyone on your system unrestricted access to the product. You can cancel this effect, after installation, by invoking the following command (using a valid PSS administrator account):

```
$ PSS/AUTH/SEC/TYPE=T/OBJECT=* DELETE *
```

Step 9: Allow the Installation to Complete

You will not be prompted until the installation completes successfully. Then it displays the following messages:

```
        Installation of PSS V3.4 completed at 12:20

 VMSINSTAL procedure done at 12:57
```

Step 10: Log out

Note that VMSINSTAL deletes or changes entries in the process symbol tables during the installation. Therefore, if you continue to use the System Manager's account and want to restore those symbols, log out and log in again.

```
$ LOGOUT   Return
SYSTEM     logged out at 27-July-1988  12:58:00.00
```

2.1.10 Error Conditions

If the installation procedure fails for any reason, VMSINSTAL displays the following message:

```
%VMSINSTAL-F-UNEXPECTED, Installation terminated due to unexpected event.
```

This unexpected event can result from any of the following conditions:

- Insufficient disk space to complete the installation
- Insufficient AST quota
- Insufficient buffered I/O-byte count
- Insufficient subprocess quota
- Insufficient open-file quota
- Insufficient process paging-file quota
- Insufficient process working-set quota
- Insufficient system maximum-working set
- Incorrect version of VMS

For descriptions of the error messages, generated by these conditions, see the *VAX/VMS System Messages and Recovery Procedures Reference Manual* and the *Guide to VAX/VMS Software Installation*.

If you are notified that any of these conditions exist, take the appropriate action, as described in the message. You might need to change a system parameter (with SYSGEN) or increase an authorized quota value (with AUTHORIZE). If the installation fails, restart the installation procedure from Step 2:.

2.1.11 A Sample Installation

A sample installation, indicating the sequence of answering the various VMSINSTAL prompts and queries, is shown in Example 2–2.

Example 2–2 Sample Installation of PSS

```
$ SET UIC [1,4]
$ SET DEFAULT SYS$UPDATE
$ @VMSINSTAL

        VAX/VMS Software Product Installation Procedure V5.0-2

It is 15-JUN-1989 at 15:43.
Enter a question mark (?) at any time for help.

* Are you satisfied with the backup of your system disk [YES]? YES
* Where will the distribution volumes be mounted: $2$MUA0:

Enter the products to be processed from the first distribution volume
set.
* Products: PSS034
* Options:
Please mount the first volume of the set on  $2$MUA0:.
* Are you ready? YES

The following products will be processed:

  PSS V3.4

 Beginning installation of PSS V3.4 at 10:51

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

******************************************************************

Attention -  System Manager / Installer:

1. Please remember to add SYS$STARTUP:PSS_STARTUP
 to your VMS Startup procedures.

2. PSS Release Notes can be found in SYS$HELP.

3. PSS DCL commands will be added to your DCL Tables.

4. PSS Help will be added to your VMS Help library.
```

Example 2–2 Cont'd on next page

Example 2–2 (Cont.) Sample Installation of PSS

```
*****************************************************************

* Do you want to purge files replaced by this installation [YES]? YES
* Do you want to disable PSS security [NO]? NO
%PSSCMD-I-SECNOTFND, security file not found.
%PSSCMD-I-CRESEC, security file VMI$ROOT:[SYSUPD.PSS034]PSS_AUTHORIZE
.DAT;1 created.
%PSS-S-CRESECFIL, PSS Authorization file created.
%VMSINSTAL-I-MOVEFILES, Files will now be moved to their target
directories...

 Installation of PSS V3.4 completed at 10:56

Enter the products to be processed from the next distribution volume
set.
* Products: EXIT

        VMSINSTAL procedure done at 10:56
```

After you install PSS, ASCI recommends that you include the following command procedure as part of your system start up:

```
$ @SYS$STARTUP:PSS_STARTUP
```

2.2 Installation of EPSS

2.2.1 Release Notes

EPSS provides online release notes that you can display or print.

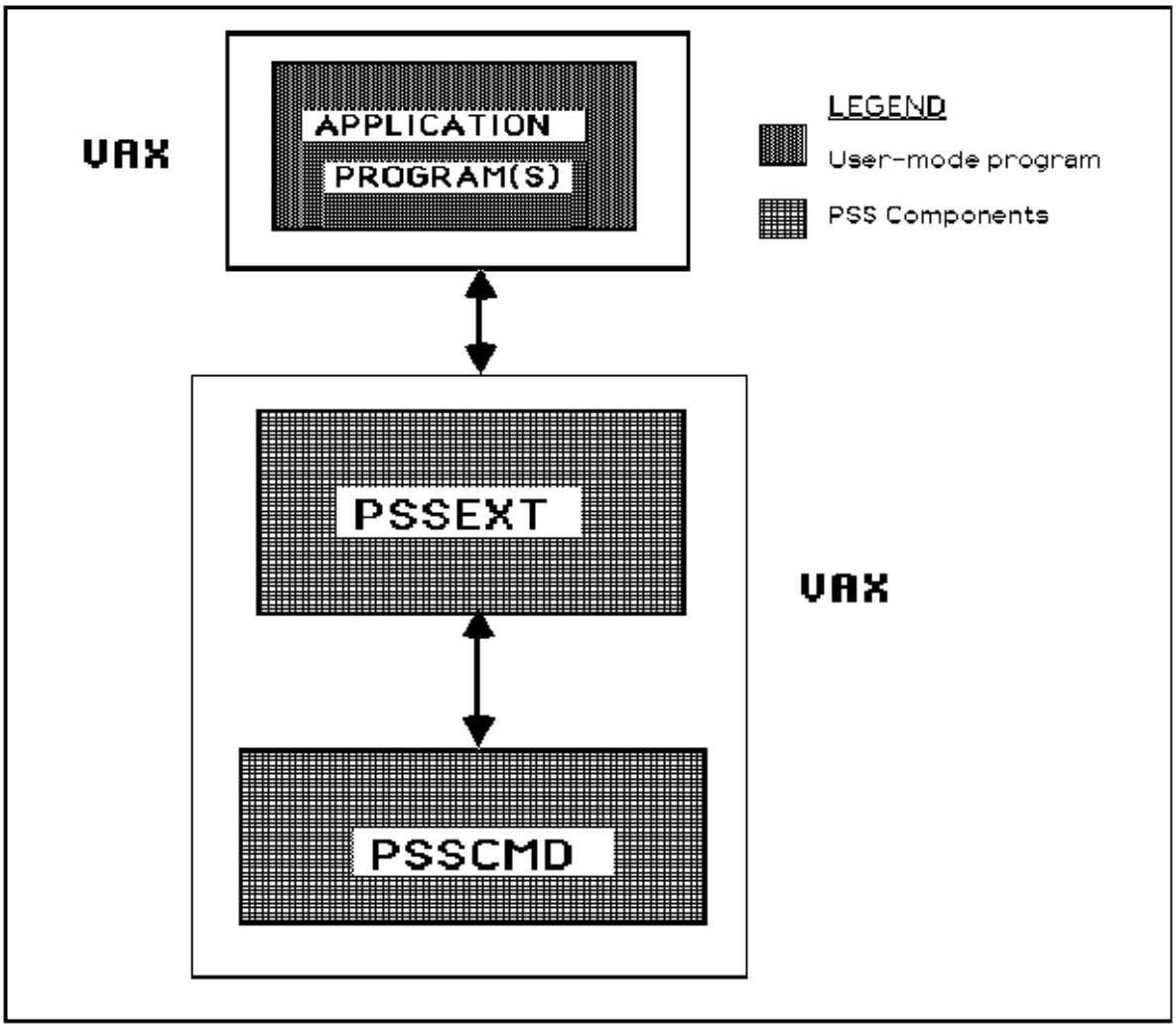
After printing and/or displaying the release notes, they are available from the VMSINSTAL procedure if you use the N option.

After installation the release notes will be in the file, SYS$HELP:EPSS033.RELEASE_NOTES, as Example 2–3 illustrates. In addition online help also provides the location of the release-notes file.

Example 2–3 Using Help to Find EPSS Release Notes

```
$ HELP EPSS RELEASE_NOTES

EPSS

  Release_notes

     The release notes for EPSS V3.3 have been placed in the file
     SYS$HELP:EPSS033.RELEASE_NOTES.
```

Figure 2–2 EPSS Component Flow



2.2.2 Before You Install EPSS

EPSS consists of one main component—-*PSSEXT*. The first of these components requires system resources, which EPSS must have available for proper product operation, as shown in Figure 2–2.

PSSEXT

This component performs the actual playback requests. Input scripts are opened and data is sent, through the specified terminal ports, to the System Under Test. PSSEXT is a normal user-mode, non-privileged program.

2.2.3 Installation Checklist

Check that you have each item in the following list before you start the installation:

1. A valid version of the VMS operating system (see Section 2.2.4)
2. A backup of your system disk (see Section 2.2.5)
3. Required privileges, quotas, disk space, and time (see Section 2.2.6)
4. A working familiarity with the VMSINSTAL procedure (see Section 2.2.7)

2.2.4 Prerequisite Software

EPSS requires that you install a valid VMS Version 5.0 or 5.x(-n) operating system on your system before the installation of PSS. To determine what version of VMS you have, type the following command:

```
$ SHOW SYSTEM

VAX/VMS V5.0  on node PSS  1-DEC-1988 11:33:54.06   Uptime    6 22:25:03
  Pid    Process Name    State  Pri      I/O       CPU       Page flts Ph.Mem
```

The first line of the displayed information shows the VMS Version.

VMSINSTAL checks for a valid version and aborts the procedure if it does not find one.

2.2.5 System Disk Backup

It is recommended that you take the precaution of backing up your system disk before any installation. (See the *VAX/VMS System Manager's Reference Manual* for instructions on backing up the system disk.)

2.2.6 VMSINSTAL Requirements

ASCI recommends that you run the VMSINSTAL procedure from the SYSTEM account.

The account you use for the installation must have the following privileges:

- CMKRNL
- TMPMBX

The account you use for the installation must have the following quotas:

- ASTLM = 24
- BIOLM = 18
- BYTLM = 18000
- DIOLM = 18
- ENQLM = 30
- FILLM = 20

EPSS requires a net utilization of at least 500 free disk blocks on your system disk.

The installation takes about 5 minutes from magnetic tape, operating on a VAX 8250.

2.2.7 The VMSINSTAL Procedure

VMSINSTAL is a Digital-supplied, command procedure, used for installing software products from their distribution media to your system. To invoke the procedure, issue the following command:

```
$ @SYS$UPDATE:VMSINSTAL  product  device  OPTIONS options_flag
```

You can invoke the command procedure, VMSINSTAL, without using any of the shown parameters. If so, the procedure prompts you for the necessary information.

Table 2–2 VMSINSTAL Parameters

Parameter	Description
product	The product name and version—in this case, EPSS033 This parameter is optional.
device	The device where you mount the distribution volume The format is ddcu, when dd is the device code, c is the controller, and u is the unit number. You may use a logical name that translates to a valid device specification. This parameter is optional.
OPTIONS	The presence of this keyword parameter that indicates, whether you have used the options_flag parameter to specify any options This parameter is optional.
options_flag	A letter which specifies an option The only valid option for this installation is N, which allows you to print and/or view the release notes before the installation continues. This parameter is optional and must follow the keyword OPTIONS.

The EPSS installation process uses the standard VMS installation procedure, VMSINSTAL. EPSS is normally distributed on 9-track, 1600-bpi magtape in BACKUP format.

The next section documents the installation procedure, step by step. Future versions of VMSINSTAL might contain new prompts that are not documented here; however, the installation should proceed normally.

2.2.8 Performing the Installation

Step 1: Log In

Log into the System Manager's account (SYSTEM) and use the operator's console terminal. Make sure that you do have all of the items, listed in the checklist in Section 2.2.3.

```
Username:  SYSTEM     Return
Password:             Return
```

Step 2: Run VMSINSTAL

Invoke the installation command procedure as follows:

```
$ @SYS$UPDATE:VMSINSTAL EPSS034 $2$MUA0: OPTIONS N     Return
```

EPSS033 is the product name and version.

2MUA0: represents a valid device name on which you mount the distribution media. If the media is a disk, use the directory and device names.

OPTIONS N indicates that you want VMSINSTAL to prompt you for the Release Notes options.

If you do not specify a product or device, the system prompts you for it.

If you do not want to type or print the release notes before the installation, omit OPTIONS **N**. After the installation you can find the release notes in the SYS$HELP directory.

To abort the installation, type CTRL Y. The installation procedure then deletes any files you have created up to that point and exits. To restart the installation, proceed with Step 2:.

Step 3: Check Warnings

VMSINSTAL might display any or all of the following warning messages and ask if you want to continue.

```
%VMSINSTAL-W-NOTSYSTEM, You are not logged in to the SYSTEM account.
%VMSINSTAL-W-DECNET, Your DECnet network is up and running.
%VMSINSTAL-W-ACTIVE, The following processes are still active:
   .
   .
   .
* Do you want to continue anyway? [NO]   YES  Return
```

If you choose not to continue, you return to the DCL level. Correct the warning condition, and restart with Step 2:, above.

Step 4: Create a Backup

If you have not backed up your system disk, do not continue with the installation. If you need to back up the system disk, type NO. VMSINSTAL exits and returns you to the DCL prompt. Backup your system disk and start VMSINSTAL at Step 2:, above.

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

Step 5: Load the Distribution Media

VMSINSTAL asks you to load the distribution media onto the device.

```
Please mount the first volume of the set on $2$MUA0:.
* Are you ready? [YES]   YES  Return
%MOUNT-I-MOUNTED, EPSS034 mounted on _$2$MUA0: (HSC000)
```

Step 6: Select the Release Notes Option

This step applies only if you specified the OPTIONS **N** in Step 2:.

```
Release Notes Options:

 1.  Display release notes
 2.  Print release notes
 3.  Both 1 and 2
 4.  Copy release notes to SYS$HELP
 5.  Do not display, print or copy release notes

* Select option [2]:   2    Return
* Queue name [SYS$PRINT]:   SYS$PRINT    Return
Job EPSS034 (queue SYS$PRINT, entry 1117) started on SYS$PRINT

* Do you want to continue the installation? [YES]
   YES    Return
```

Option 1 immediately displays the release notes at your terminal. Option 2 prompts you for a print queue and spools the release notes for printing. Option 3 does both. Option 4 copies the release notes to the SYS$HELP directory, and Option 5 prevents VMSINSTAL from displaying, printing, or copying them.

```
* Queue name [SYS$PRINT]:
```

Then VMSINSTAL asks, if you want to continue the installation:

```
* Do you want to continue the installation? [YES]
```

Step 7: Purge Existing Files

The PSS system creates a new generation of these system files during installation:

- SYS$LIBRARY:DCLTABLES.EXE
- SYS$STARTUP:EPSS_STARTUP.COM

VMSINSTAL displays the following prompt, asking whether you want to purge old versions of these files:

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

Step 8: Allow the Installation to Complete

The system provides no more prompts until the installation completes successfully. Then it displays the following messages:

```
        Installation of EPSS V3.3 completed at 12:20

 VMSINSTAL procedure done at 12:57
```

Step 9: Log out

Note that VMSINSTAL deletes or changes entries in the process symbol tables during the installation. Therefore, if you continue to use the System Manager's account and want to restore those symbols, log out and log in again.

```
$ LOGOUT   Return
SYSTEM     logged out at 27-July-1988  12:58:00.00
```

2.2.9 Error Conditions

If the installation procedure fails for any reason, VMSINSTAL displays the following message:

```
%VMSINSTAL-F-UNEXPECTED, Installation terminated due to unexpected event.
```

This unexpected event can result from any of the following conditions:

- Insufficient disk space to complete the installation
- Insufficient AST quota
- Insufficient buffered I/O-byte count
- Insufficient subprocess quota
- Insufficient open-file quota
- Insufficient process paging-file quota
- Insufficient process working-set quota
- Insufficient system maximum-working set
- Incorrect version of VMS

For descriptions of the error messages, generated by these conditions, see the *VAX/VMS System Messages and Recovery Procedures Reference Manual* and the *Guide to VAX/VMS Software Installation*.

If you are notified that any of these conditions exist, take the appropriate action, as described in the message. You might need to change a system parameter (with SYSGEN) or increase an authorized quota value (with AUTHORIZE). If the installation fails, restart the installation procedure from Step 2:.

2.2.10 A Sample Installation

A sample installation, indicating the sequence of answering the various VMSINSTAL prompts and queries, is shown in Example 2–4.

Example 2–4 Sample Installation of EPSS

```
$ SET UIC [1,4]
$ SET DEFAULT SYS$UPDATE
$ @VMSINSTAL

        VAX/VMS Software Product Installation Procedure V5.0-2

It is 15-JUN-1989 at 15:43.
Enter a question mark (?) at any time for help.

* Are you satisfied with the backup of your system disk [YES]? YES
* Where will the distribution volumes be mounted: $2$MUA0:

Enter the products to be processed from the first distribution volume
set.
* Products: EPSS033
* Options:
Please mount the first volume of the set on  $2$MUA0:.
* Are you ready? YES

The following products will be processed:

  EPSS V3.3

        Beginning installation of EPSS V3.3 at 15:44

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

******************************************************************

Attention -  System Manager / Installer:

1.      Please remember to add SYS$STARTUP:EPSS_STARTUP
        to your VMS Startup procedures.

2.      EPSS Release Notes can be found in SYS$HELP.

3.      PSS DCL commands will be added to your DCL Tables.

4.      PSS Help will be added to your VMS Help library.
```

Example 2–4 Cont'd on next page

Example 2–4 (Cont.) Sample Installation of EPSS

```
*****************************************************************

* Do you want to purge files replaced by this installation [YES]? YES
%VMSINSTAL-I-MOVEFILES, Files will now be moved to their target
directories

        Installation of EPSS V3.3 completed at 15:46

Enter the products to be processed from the next distribution volume
set.
* Products: EXIT

        VMSINSTAL procedure done at 15:47
```

After you install EPSS, ASCI recommends that you include the following command procedure as part of your system start up:

```
$ @SYS$STARTUP:EPSS_STARTUP
```

3 CONFIGURATION

3.1 Configuring PSS

Configuring the PSS environment consists of generating pseudo-terminal units for PSS and, more importantly, setting VMS system resources for PSS use.

3.1.1 Pseudo Terminal Units

A "pseudo" terminal is a PSS device that acts like a real terminal but doesn't really exist. PSS uses pseudo terminals for simulation purposes.

By default, the VMS device name for the PSS pseudo terminal is PSU. For example, PSUA0 is the first pseudo terminal unit. PSS_STARTUP defines the pseudo-terminal-device name, as well as the number of pseudo terminals you want the VMS system to have.

```
$ @SYS$STARTUP:PSS_STARTUP number-units device-name
```

The above procedure starts the PSS environment on a VMS system. This procedure allows you to enter two optional parameters. The first parameter, **number-units** is an integer from 1 to 65,537 that indicates the number of pseudo-terminal devices to configure for this system. By default one (1) pseudo-terminal device is configured.

The second parameter, **device-name** is a unique name (including the controller letter, **A**) that defines the pseudo-terminal units. By default, PSUA is the device name for pseudo-terminal units.

Prior to V3.1, pseudo-terminal units were known as PTA. However, Digital Equipment Corporation has begun using this device mnemonic. If you want to have PSS continue using PTA, simply invoke the start-up procedure with this command:

```
$ @SYS$STARTUP::PSS_STARTUP 1 PTA
```

This would create one pseudo terminal unit named PTA0. The controller letter is "A" for all pseudo terminal units. If you want to create 64 pseudo terminals named PSUA0 through PSUA63, enter:

```
$ @SYS$STARTUP:PSS_STARTUP 64
```

This manual assumes that you use the device name PSUA to designate pseudo terminals.

Note: The number of PSS terminal sessions that can run concurrently varies, depending on the type of VAX CPU and the load, currently running on the system. You must configure at least one pseudo device (for example, PSUAO). PSS requires this device for operation, but it is available for general PSS use.

3.1.2 Starting PSS

To start up the PSS system, issue the following command:

```
$ @SYS$STARTUP:PSS_STARTUP
```

The first time you run this command procedure, it loads the driver (PTDRIVER), connects the pseudo terminals, and starts the Server (PSSRV). Subsequent execution of this procedure on an already running VMS system causes only the Server to run (if it is not already running). To add more pseudo terminals **after** invoking PSS_STARTUP, you must manually execute the VMS SYSGEN utility and use the CONNECT command.

3.1.2.1 Privileges Required

To run successfully, PSS_STARTUP requires the following VMS privileges:

- ACNT
- ALTPRI
- CMKRNL
- DETACH
- OPER
- PRMMBX
- PSWAPM
- SYSNAM
- SYSPR
- WORLD

3.1.2.2 Quotas Required

Examine PSS_STARTUP to determine whether the PSSRV component has enough quota to execute your workload. Table 3–1 describes the quota and value necessary for proper PSS operation.

Table 3–1 PSSRV process quotas

Quota	Initial	Add	Use
ASTLM	100	4	for QIO and TIMER requests
BIOLM	100	1	for terminal QIO requests
BYTLM	65000	n/a	
DIOLM	100	3	for disk QIO requests
ENQLM	500	3	for open files
FILLM	300	3	for open files
PGFLQUOTA	50000	100	for an increase in virtual memory
TQELM	100	1	for TIMER requests
WSEXTENT	3000	n/a	
WSQUOTA	1000	n/a	

To support approximately 100 terminal sessions, the quota's, specified in Table 3–1, should be sufficient for PSSRV. The ADD column represents the incremental amount you must add to the INITIAL value for each additional terminal you want to support. If the workload you want to run requires additional quotas, edit PSS_STARTUP.COM and modify the RUN command appropriately.

3.1.3 PSS and VMS System Resources

PSS requires system-dynamic, nonpaged memory to correctly and efficiently operate. As with most computer applications, some of this memory is static and some dynamic. To help you determine the correct amount of dynamic memory PSS needs, let us examine the requirements PSS makes of nonpaged pool.

The PTDRIVER component requires approximately 3,400 bytes of nonpaged memory in which to operate. Each active session requires a dynamic allocation of approximately 312 to 568 bytes (depending upon record/playback and the specification of a user-delimiter table). On average PSS attempts to maintain approximately 1,600 bytes of nonpaged pool per session for input/output capture and/or input playback.

Recording, particularly output data, requires dynamic memory. PSS maintains a safety "valve" of approximately 8,000 bytes in which to buffer requests. Rather than continue to monopolize pool, PSS induces terminal sessions to wait by sending an XOFF command to the host to prevent additional output. When peak-pool requirements subside, PSS sends an XON command to the host to continue its output.

Table 3–2 Nonpaged Pool

Component	Amount	Use
PTDRIVER	3,400 bytes	Once for the LOAD
Terminal	312-568 bytes	Per session
Dynamic	1,600 bytes	Or 10 IRP's-per terminal
PSU	256 bytes	Per pseudo terminal

During Record and Playback operations, PSS requires varying amounts of nonpaged pool. PSS uses this pool for several interim data structures, and primarily data buffering. PSS maintains a minimum requirement of thirty (30) IRP's (each IRP is 196 bytes) and adjusts this number upwards to accommodate arriving data.

While the actual amount of pool varies, based on the PSS operations requested, ASCI suggests increasing the IRPCOUNT parameter, using the VMS SYSGEN utility. This recommendation avoids VMS's dynamic growth of nonpaged pool during a PSS operation. PSS allocates pool from the IRP lookaside list, as well as the nonpaged pool area.

NPAGEDYN, a VMS SYSGEN parameter, controls the initial allocation of nonpaged pool.

To compute the total amount to increase NPAGEDYN, use the following formula:

```
ADD_NPAGEDYN = 3400 + (number_pseudo_units*256) +
                      (number_active_sessions*568)
```

To compute the total amount to increase IRPCOUNT, use the following formula:

```
ADD_IRPCOUNT = 200 + (number_active_sessions*30)
```

To avoid running out of nonpaged pool, ASCI recommends that you set NPAGEVIR and IRPCOUNTV higher than NPAGEDYN and IRPCOUNT, respectively. NPAGEVER, a VMS sysgen PARAMETER, controls the maximum, dynamic allocation of nonpaged pool. IRPCOUNTV, a VMS SYSGEN parameter, controls the maximum, dynamic allocation of IRP's.

3.1.4 Shutting PSS Down

ASCI recommends that you add the following command to the VMS site-specific shutdown procedure (SYS$MANAGER: SYSSHUTDOWN.COM):

```
$PSS/SHUT
```

This command stops all PSS sessions and closes all PSS files properly before VMS shuts down.

3.2 Configuring EPSS

3.2.1 Starting EPSS

To start up the EPSS system, issue the following command:

```
$ @SYS$STARTUP:EPSS_STARTUP
```

3.2.2 Privileges

PSSEXT requires the following privileges:

- DETACH
- LOG_IO
- OPER
- PHY_IO
- PRMMBX
- SYSNAM
- SYSPRV
- TMPMBX
- WORLD

3.2.3 Quotas

Examine EPSS_STARTUP to determine, whether the PSSEXT component has enough quota to execute your workload. Table 3–3 describes the quota and value necessary for proper EPSS operation.

Table 3–3 PSSEXT process quotas

Quota	Initial	Add	Use
ASTLM	500	4	for QIO and TIMER requests
BIOLM	500	1	for terminal QIO requests
BYTLM	65000	n/a	
DIOLM	500	3	for disk QIO requests
ENQLM	1500	3	for open files
FILLM	500	3	for open files
PGFLQUOTA	50000	100	for an increase in virtual memory
TQELM	500	1	for TIMER requests
WSEXTENT	3000	n/a	
WSQUOTA	1000	n/a	

If PSSEXT exhausts of any of these quotas, you might experience performance degradation or the abnormal exit of PSSEXT, depending upon the particular quota, exhausted.

The maximum number of channels a VMS process can use is 127 and is governed by the SYSGEN parameter, CHANNELCNT. Depending upon your usage of EPSS this static SYSGEN parameter might need to be increased (the current parameter maximum is 2047).

4 PRODUCT USES

4.1 Additional Features

Although ASCI designed PSS primarily as an application testing facility, PSS provides additional benefits, not limited to testing.

However, before beginning the discussion of PSS functions, ASCI must issue a warning. PSS provides many options and features but, as with any system software, not all of these items "come free." Unwise use of the product can result in system degradation, due to resource depletion and/or system overhead. This chapter offers some general guidelines in the use of PSS but, naturally, in individual circumstances the rule of common sense must prevail.)

4.1.1 Performing Performance Simulations

You can use PSS to measure the load that the interactive use of multiple terminals places on your system. It plays back prerecorded-terminal sessions on real terminals or on dynamically-created, "pseudo" terminals. Thus, you can judge, for example, the response time of your system, when you add "n" number of terminals or when "x" number of interactive users run applications while you're running a large batch job.

The /OUTPUT option of the PSS-PLAYBACK command records all the output, directed to the playback terminal, on a disk file. ASCI intends that you use this feature during application testing, when you need to compare the results of tests with real output. Using this facility during performance simulation, when you must simulate large numbers of terminals, causes disk I/O overhead. This overhead—-when added to the normal disk I/O load that the programs (run by the simulation) cause—can skew the results of your performance evaluation.

The use of "pseudo" terminals also incurs a certain amount of system overhead, because you must solicit the output from pseudo terminals. When you use real terminals, you don't incur this overhead.

4.1.2 Application Testing

Probably the most useful application of PSS consists in allowing an applications programmer to create a test database of terminal interactions. You can store these interactions in one or more files and then replay them. Since PSS permits you to record the output of both the original recording, as well as the playback session, it makes available an automated form of application-regression testing. It stores a "bible" copy of the output log, and each time you make a change to the system you can run this automated-test system to detect any changes. Thus, you can investigate both the anticipated changes to the system that the DIF facility lists, as well as *unexpected* changes.

PSS captures its input data at the lowest possible level in VMS (prior to VMS-Terminal-Class-driver action). This allows a programmer who has written QIO logic actually to interface with the terminal to make changes in that QIO's performance. In addition you can experiment with different types of QIO's, as well as varying buffer sizes, and discover the resulting differences in response times and uses of system resources.

4.1.3 Capturing VAX Console Events

Use PSS in the production environment to obtain a dynamic record of events. For example, PSS can monitor an operator's console, keeping a disk-based log of all input to and output from that device. Using the PSS conversion utility, you can create your own version of the operator's log to view at your leisure. To find a specific item in the operator's log, using the VMS editor, you can look for it at *your* terminal, instead of searching through reams of console listings.

4.1.4 Turning an Interactive into a Batch Procedure

You can also use PSS as a "super" command procedure. For example, your data center might have a critical job that it must run every day. Running the job might be a complex process, requiring an operator to enter information at several points. With PSS, you must record the process only once. From that point on you can simply playback the input file and trust PSS to ensure that it enters the correct responses at the appropriate points. Through the use of the PSS conversion utility, you can even edit the recorded file and change those responses that differ from the previous run of the job.

4.1.5 System Security

Dial-up lines have been a traditional menace to system security procedures. If you think that unauthorized, dial-up users might violate the security of your system, PSS can help you. Using the RECORD facility, you can log the input and output data of any dial-up port to a disk file or to a control terminal. The users of the dial-up lines cannot detect that you can record all their actions.

The /PERMANENT qualifier of the PSS/RECORD command places the specified terminal in permanent-record mode. This means that the terminal remains in the recording mode until you bring down the PSS-Server process. A PSS/STOP command, issued to a terminal in permanent record mode, has *no* effect. Therefore, someone cannot issue a PSS/STOP in an attempt to disable the recording of your terminal session.

4.1.6 Capacity Planning

PSS and EPSS can help determine, when its time to reconfigure your hardware or software systems. Use PSS/EPSS to generate a load and, in conjunction with Digital's SPM or VPA products, create a report which provides a technical analysis of how your system is doing, as well as a report of what response time your system is capable of meeting.

4.1.7 Database Load

By first issuing a PSS/RECORD command at each terminal, involved in the initial load, and then issuing PLAYBACK sessions on real or pseudo terminals, PSS can reload a database. This feature proves valuable in many situations, as follows:

- When the possibility exists that you could lose data which you would need to reenter manually
- When a program, undergoing testing, has bugs that continuously corrupt the data and, therefore, require your reloading the database to restore it to a usable state

4.1.8 Interactive Demonstration of an Application

To demonstrate a system to one or more users, use PSS/RECORD to save the initial run of the system. Then play the session back on one or more terminals at a time, independent of physical locations. The %SUSPEND lexical can be very useful, when you want to present a screen of information and allow the user to press Return to continue the demonstration/presentation.

4.1.8.1 Benchmarking Software for Purchase

PSS excels at benchmarking software that you might plan to purchase. Simply execute the software to be purchased on your system and record all interactions with it. You can then edit the input scripts and provide whatever user mix you desire. Remember that PSS/EPSS can provide response-time information.

4.1.9 Parallel Testing

To run a full-production operation under a test environment, use PSS/RECORD to save the initial production run. Then use PSS/PLAYBACK to replay it at your convenience, for example, to locate bottlenecks in the system and to monitor the use of your system's resources.

4.1.10 Production Run in Testing Environment

You can use PSS to execute a full-production run in a testing environment. You need only specify PSS/PLAYBACK to save the initial-production run and replay it at a more convenient time, using PSS/PLAYBACK. You can, thus, locate bottlenecks in the system and determine your use of system resources.

4.1.11 Checking Response Times

To run a well-defined script against a real or pseudo terminal at various times of day, you could draw a time-series graph to show how the system responds. Use the PSS/PLAYBACK/RESPONSE command to obtain a complete response-event file which can then be reported through the PSS/RESPONSE utility.

4.1.12 Restart Recovery

To use PSS with applications that have no checkpoint-restart procedures, you must back up the database before the start of every business day. PSS then logs every terminal user of the database during the day. In the event an application crashes, PSS can play back recorded sessions to restore the database to its original state at the beginning of the day, saving you from manually keying in the lost transactions.

5 USING PSS

5.1 Overview of PSS

5.1.1 Recording Mode

To enable PSS to begin recording a terminal session, you must issue a PSS/RECORD command. A terminal session consists of two data streams:

- Input characters entered
- Output (including echo) characters displayed

PSS captures and collects input and/or output streams into two separate disk files. Once you begin recording a terminal session, PSS continues to passively record the selected terminal until you stop the session. At this time you have two files available—one for input and the other for output. Chapter 4 describes uses for the output file, but PSS uses the input file that it created for the playback process.

5.1.2 Playback Mode

Once you record a terminal input stream, you can elect to play it back on a real or pseudo terminal. To accomplish this, enter a PSS/PLAYBACK command. PSS reads your input file and plays it back to the selected terminal. Notice that PSS plays the input characters in approximately the same manner and time as they were entered during the recording phase. This adds to the realism of the playback. When you begin a playback, you can tell PSS to capture the output of the playback as well.

5.1.3 Reporting

To request a report on any PSS-selected terminal, enter the PSS/REPORT command.

- If the terminal is in record mode, PSS displays information, such as status and input/output character counts.

- If the terminal is in playback mode, it displays additional information, concerning response-time statistics.

5.1.4 Response-Event Reporting

PSS offers a facility which allows you to obtain the actual response-times seen for each input event. This information can be useful in the following ways:

- To determine response-times for specific events
- To plot the response-time data
- To programatically analyze the response-time data
- To obtain an input-event-to-a-response-time-seen report

Response-time event recording is a qualifier that you specify on the PSS/PLAYBACK command. Once the playback has terminated, you must issue the PSS/CONVERT command to convert the response-time-event file from PSS internal format to an RMS sequential fixed-length file. This file can then be analyzed by a user-written program or report-writing tool. In addition PSS provides the PSS/RESPONSE command that produces a report, matching input data events to the response-times, encountered.

5.1.5 Getting Help

PSS uses the DCL syntax for entry of its parameters and qualifiers. The installation process of PSS adds the PSS-help text to your system-help library. For on-line help on PSS, type this command:

```
$ HELP PSS
```

The command displays the following information:

```
PSS

   Allows a user to enter commands to the Performance Simulation System,
   produced by Advanced Systems Concepts, Inc. PSS allows you to record
   (intercept) and/or playback a selected terminal session. The playback
   can occur on a real or pseudo terminal.

   For more information concerning PSS, see the PSS User's Guide.

   Format:

   PSS/command_qualifier parameter1 [parameter2]

   The command qualifier may be one of the following: CONVERT, PLAYBACK,
   MONITOR, RECORD, REPORT, SHUT, STOP.


  Additional information available:

  CONVERT   IDENT    MONITOR    PLAYBACK   RECORD   REPORT   RESPONSE
  SHUT      STOP     Lexicals   Release_notes
```

5.1.6 Invoking PSS

All PSS commands share a common syntax, as follows:

```
$ PSS/qualifier Terminal/param-qualifier
```

The separate items in this command syntax represent the following:

PSS	Invokes the PSS command interpreter
/qualifier	Depicts one of these qualifiers: AUTHORIZE, CONVERT, MONITOR, PLAYBACK, RECORD, REPORT, RESPONSE, SHUT, or STOP
Terminal	Symbolizes a real or pseudo-terminal device name
/param-qualifier	Reflects one or more of these parameters: /INPUT, /OUTPUT, /USER_DELIM and /NOTIFY, etc.

To enable recording of the input and output streams of TTA4, for example, enter the following command:

```
$ PSS/RECORD TTA4
```

This command begins recording on TTA4. Input characters are placed in a file, named PTTA4.INP, and output is placed in a file, named PTTA4.OUT.

To stop recording the terminal session, enter the following command:

```
$ PSS/STOP TTA4
```

To enable playback of the just recorded data on TTA4, enter the following command:

```
$ PSS/PLAYBACK TTA4
```

This command plays back input characters on TTA4 from the file PTTA4.INP, but does not generate any output.

Section 5.2 describes all of the PSS commands that you can execute.

5.1.6.1 File-Specification-Default Rules

PSS applies the following file specification defaults for input, output, and playback files:

```
[Default Device/Directory] P [Terminal Device].{INP}
                                               .{OUT}
                                               .{PLB}
                                               .{RSP}
```

The items in this syntax represent the following:

- **Default Device/Directory** is your current default device and directory
- **P** stands for PSS.
- **Terminal Device** is the terminal device name and unit.
- The extensions listed represent the following:

  .INP is the input file.

  .OUT is the output file (record).

  .PLB is the output file (playback).

  .RSP is the response file (playback).

Table 5–1 Examples:

$ PSS/REC TXA6	yields	PTXA6.INP and PTXA6.OUT
$ PSS/REC TXA6/NOOUT	yields	PTXA6.INP
$ PSS/REC/INP=MYSCRIPT TXA6	yields	MYSCRIPT.INP and PTXA6.OUT

5.2 Command Summary

5.2.1 CONVERT Command

The PSS/CONVERT command converts files from PSS internal format to standard, RMS-text files or vice-versa. You can use PSS/CONVERT only on files that PSS has not already opened. Therefore, you normally use PSS/STOP to terminate PSS activity so that you can perform file conversion. You do use PSS/CONVERT to do the following:

- Convert an input file to an RMS text file
- Convert an RMS text file to an input file
- Convert an output or playback file to an RMS text file
- Convert an output or playback file directly to a terminal device
- Convert a response-event file to an RMS sequential file

Example 5–1 CONVERT Examples

```
1  $ PSS/CONVERT PTXA0/INPUT TEST/TEXT
```

This command converts the PSS input file, named PTXA0.INP, to an RMS text file named, TEST.TXT.

```
2  $ PSS/CONVERT TEST/TEXT PTXA0/INPUT
```

This command converts the RMS text file, named TEST.TXT, to a PSS input file, named PTXA0.INP.

```
3  $ PSS/CONVERT PTXA0/OUTPUT APPL/TEXT
```

This command converts the PSS output file, named PTXA0.OUT, to an RMS text file, named APPL.TXT. Please, note that you cannot convert a text file to a PSS output file.

```
4  $ PSS/CONVERT PTXA0/OUTPUT TT
```

This command converts the PSS output file, named PTXA0.OUT, into text format and displays it on your terminal in the same manner, as you once viewed it.

```
5  $ PSS/CONVERT PTXA0/RESPONSE TEST/TEXT
```

This command converts the PSS response-event file named, PTXA0.RSP, to an RMS sequential file, named TEST.TXT.

Chapter 7 discusses file conversion in greater detail.

CONVERT

This command allows you to convert PSS text or internal files into different formats.

FORMAT

PSS/CONVERT/qualifiers Input-file Output-file

Command Qualifiers	Defaults
/ABSOLUTE	*/NOABSOLUTE*
/RECORDSIZE=value	*/RECORDSIZE=2048*
/TRANSLATE	*/NOTRANSLATE*

Positional Qualifiers	Defaults
/INPUT	*See Text*
/PLAY	*See Text*
/OUTPUT	*See Text*
/RESPONSE	*See Text*
/TEXT	*See Text*

prompts

From_File:	Input file
To_File:	Output file

restrictions

You cannot convert /TEXT to /OUTPUT, /PLAY, or /RESPONSE.

PARAMETERS

input-file

Specify the text or PSS internal-format, file specification you want to feed into the conversion process.

output-file

Specify the text or PSS internal-format, file specification you want the conversion process to produce as output.

DESCRIPTION

The CONVERT command converts a text file to a PSS internal-format file or vice-versa.

COMMAND QUALIFIERS

/ABSOLUTE
/NOABSOLUTE (default)

To convert all delta-time expressions into absolute time, you can specify this qualifier, when you convert a PSS-input, internal-format file to a text file. You cannot, however, convert the resulting input-text file back to internal format, but this qualifier proves useful, when you want to know the actual time of an event.

/RECORDSIZE=value
/RECORDSIZE=2,048 (default)

You can use this qualifier to limit the size of records, written to a text file. The default is 2,048 bytes.

/TRANSLATE
/NOTRANSLATE (default)

This qualifier converts unprintable-control characters into printable sequences. You cannot convert the resulting input-text file back to internal format, but this qualifier can determine and print control sequences.

POSITIONAL QUALIFIERS

/INPUT

This qualifier denotes this file as a PSS internal-format, input-script file. The default-file extension is .INP.

/OUTPUT

This qualifier denotes this file as a PSS internal-format, output file. The default-file extension is .OUT.

/PLAY

This qualifier denotes this file as a PSS internal-format, playback-output file. The default-file extension is .PLB.

/RESPONSE

This qualifier denotes this file as a PSS internal, response-event, playback file. The default-file extension is .RSP.

/TEXT

This qualifier denotes this file as a text file. The default-file extension is .TXT.

EXAMPLES

1 $ **PSS/CONVERT PTXA3/INPUT TEST/TEXT**

This command converts the PSS internal-input file, PTXA3.INP, into an ASCII-text file, named TEST.TXT.

5.2.2 MONITOR Command

The PSS/MONITOR command monitors and controls PSS activity from a video-oriented device. With PSS/MONITOR, you can create windows of various information that PSS updates in real-time, as the terminal sessions continue. In addition, PSS/MONITOR allows you to enter SPAWN, HELP and any valid PSS command (except SHUT).

MONITOR

This command allows you to monitor and control PSS.

FORMAT

PSS/MONITOR/qualifiers window_name

Command Qualifiers	Defaults
/INTERVAL=delta-time	*/INTERVAL="00:00:03"*

PARAMETERS

window_name

Specifies the window name you want to create initially. Table 8–2 lists the valid window names. By default it displays BRIEF_ALL.

DESCRIPTION

The PSS/MONITOR command allows you to interactively monitor and control PSS sessions.

COMMAND QUALIFIERS

/INTERVAL=delta-time

/INTERVAL changes the default-refresh rate at which MONITOR updates the status window(s) with new or changed information. The time, entered, must not include **days**.

EXAMPLES

```
1    $ PSS/MONITOR
```

This command produces a BRIEF_ALL window that displays all terminal sessions, active within PSS.

5.2.3 PLAYBACK Command

Use the PSS/PLAYBACK command after you have created an input (.INP) file. You create an input file (sometimes called an *input script*) through one of two methods:

- Through use of the PSS/RECORD command
- Through use of a text editor, using the lexicals, described in Chapter 7

The PSS/PLAYBACK command allows you to replay this input-script file on a specific terminal. If this is the first time you are playing back this file, you might want to watch it on a real-terminal device to ensure its accuracy.

Example 5–2 PLAYBACK Examples

```
1  $ PSS/PLAYBACK TXA0:
```

This command produces an input file, named PTXA0.INP, in your current default device/directory that you can play to terminal, TXA0:. It does not capture the output data stream of the terminal.

```
2  $ PSS/PLAYBACK/OUTPUT TXA0:
```

This command, as the one in the example above, plays back the PTXA0.INP file to terminal, TXA0:, but also captures the output data stream into file, PTXA0.PLB.

```
3  $ PSS/PLAYBACK/NOTIFY/PARAM=(USER,USER)/INPUT=TEST TXA0:
```

This command plays back the file, TEST.INP, to terminal TXA0:. Two parameters are available to the input script for use with the %PARAM lexical, if you provide it. When the playback terminates, either normally or abnormally, the command sends a message to notify you of this.

Example 5–2 Cont'd on next page

Example 5–2 (Cont.) PLAYBACK Examples

```
4  $ PSS/PLAY/SPLIT=(1,3,6,10)/INPUT=TEST/OUTPUT=TEST TXA0:
```

This command plays back the file, TEST.INP, to terminal TXA0:. It captures output data into the file, TEST.PLB.

PSS maintains the following response-time intervals:

- <1 - less than 1 second
- 1-3 - between 1 and 3 seconds, inclusive
- 3-6 - between 3 and 6 seconds, inclusive
- 6-10 - between 6 and 10 seconds, inclusive
- >10 - greater than 10 seconds

These intervals are available for examination through the PSS/REPORT command.

PLAYBACK

This command plays back an input script to a specific terminal.

FORMAT

PSS/PLAYBACK/qualifiers Terminal-device

Command Qualifiers	Defaults
/INPUT[=file-specification]	*See text*
/KEYRATE=value	*/KEYRATE=10*
/LOOP=value	*/LOOP=1*
/NOTIFY	*/NONOTIFY*
/[NO]OUTPUT=file-spec	*/NOOUTPUT*
/PARAM=parameters	*See text*
/PASSWORD=string	*See text*
/RESPONSE=file_spec	*/NORESPONSE*
/SEED=value	*See text*
/SPLIT_TIMES=value_list	*/SPLIT_TIMES=(1,2,5,30)*
/[NO]TYPE	*/TYPE*
/VARY[=factor]	*/VARY=1*

restrictions

You must execute the PLAYBACK command to a valid PSS-supported-terminal device. PSS cannot currently use the terminal device for other operations, such as Recording.

prompts

Terminal Device: terminal-device

PARAMETERS

terminal-device

Specifies the terminal device upon which you want to playback

DESCRIPTION

The PSS PLAYBACK command begins an interactive exchange between PSS and a specified terminal. PSS reads all relevant input data from an input-script file and sends the data to the terminal device, using similiar wait times, encountered during the script generation. You can capture the terminal output and write it to a disk file for later analysis.

When a terminal enters into playback mode, PSS sets the READSYNC characteristic for that device to avoid losing any input data in the event a read is not outstanding. READSYNC indicates that VMS XOFFs the terminal, when a read QIO completes and XONs the terminal, when PSS posts a read. Then, when your playback session begins, the program you're playing back to sends the first character in the input-script file as unsolicited data (data it enters as input, but for which it does not post a read). PSS does this as a convienence since logging into VMS uses an unsolicited "Carriage Return" to stimulate LOGINOUT. To cause PSS to send input as unsolicited data within the playback session, you need to use the PSS-conversion utility, described in Chapter 7.

A useful feature of PSS is its ability to playback a terminal session on a pseudo terminal. You can do this to simulate terminal loads for performance analysis and/or to not "tie up" a real terminal for a long regression test.

Pseudo terminals take the format of PSUA (your System Manager might select a different name), followed by a decimal number. Your System Manager allocates the actual unit numbers, as well as the number of pseudo terminal devices. However, all PSS systems must have at least one pseudo terminal device. To avoid confusion, the examples in this section use PSUA0. A pseudo terminal is just like a real terminal. Your application can allocate it as a device or log it into VMS. This is your responsibility. If your application expects you to log it in, you must do so. This isn't a problem since you can record a login procedure as part of your input script.

The ability to play back on a real terminal allows you to watch the activities of a playback session.

COMMAND QUALIFIERS

/INPUT[=file-specification]

This qualifier represents the input-script file. By default, /INPUT is enabled, using the default-file-specification rules, as discussed in Section 5.1.6.1. Optionally, you can replace the default with a VMS file specification.

/KEYRATE=value
/KEYRATE=10 (default)

This qualifier allows you to specify the typing rate, per second, at which the terminal receives input.

The value can be any integer within the range of 0 to 100. If you don't specify this qualifier (or you specify a value of less than ten), you get the default rate of ten characters per second.

/LOOP=value
/LOOP=1 (default)

This qualifier directs PSS to execute "n" loops of the input-file script. This numeric value, if specified, must be between 1 and 32,767. By default, /LOOP=1.

/NOTIFY
/NONOTIFY (default)

This qualifier causes PSS to notify whoever executes the PLAYBACK command, via broadcast, when the input script reaches end-of-file, aborts via the PSS/STOP command, or terminates as a result of a run-time error. The default, /NONOTIFY, only notifies you, if your playback session terminates due to a runtime error.

/OUTPUT[=file-specification]
/NOOUTPUT (default)

This qualifier directs PSS to capture output data sent to the terminal device. /NOOUTPUT directs PSS not to intercept output data to the terminal. /NOOUTPUT is the default.

/PARAM=parameters

The /PARAM qualifier allows you to pass run-time parameters to the script as it plays back. PSS uses these parameters in conjunction with %PARAM lexicals, which the script contains. Each lexical has a number that refers to the position of a parameter within the /PARAM list. When PSS encounters the %PARAM lexical during playback processing, it substitutes the corresponding parameter from the list. You can enter a maximum of ten (10) parameters on the DCL command line. DCL limitations govern the maximum length of all parameters.

A useful feature of this qualifier allows you to specify the VMS user name and password as parameters to those input scripts that contain logon sequences.

/PASSWORD=string

Your PSS security administrator can, as an additional security precaution, require that you enter a password to positively identify yourself. If you must enter a password, PSS either prompts you for the password (with echo disabled) or you can enter the password as a value to this qualifier. ASCI recommends that you allow PSS to prompt for the password.

/RESPONSE[=file_specification]
/NORESPONSE (default)

The /RESPONSE qualifier creates a response-time-event file for your current playback session. For more information, see Section 7.4.

/SEED=value

This qualifier works with the %RANDOM and %RANDOM_TIME lexicals and allows you to specify the starting value, which a random-generator algorithm uses to generate random numbers.

When you specify this qualifier, the random-number generator always generates the same set of random numbers. Therefore, when you rerun multiple playback sessions, you can compare the results of the different test runs. By default a new seed value is used for each playback session.

/SPLIT_TIMES=[time-specification,...]
/SPLIT_TIMES=(1,2,5,30) (default)

This qualifier allows you to specify the response time intervals (split times) that you are interested in tabulating. These split-time intervals are entered as either whole numbers or as fractions (up to hundredths). The default split-times are, as follows:

- < 1 - less than 1 second
- 1-2 - 1 to 2 seconds, inclusive
- 2-5 - 2 to 5 seconds, inclusive
- 5-30 - 5 to 30 seconds, inclusive
- >30 - greater than 30 seconds

Enter the above defaults like this:

```
/SPLIT_TIMES=(1,2,5,30)
```

If you enter SPLIT_TIMES, PSS expects you to enter at least one, but no more than ten, split times.

/TYPE
/NOTYPE (default)

The /TYPE qualifier allows you to enter data during playback. This is the default. To disable terminal input during playback, enter /NOTYPE. PSS requires /NOTYPE for FMS, ALL-IN-1 and any other software that interrogates the terminal as to type (for example, issuing a "Who are you?" escape sequence).

/VARY[=factor]
/VARY=1 (default)

The /VARY=factor qualifier changes the recorded, waiting times between input items. The "factor," noted above, is a floating-point number that PSS multiplies against the wait time. To cut wait times in half, specify the qualifier as /VARY=.5. Use the command, /VARY=2 to double the wait times. The default vary factor is one (1). To disable wait times, enter the qualifier, /NOVARY.

EXAMPLES

1 $ **PSS/PLAYBACK TXA3:**

Start a playback to TXA3, using the input file, PTXA3.INP. It does not intercept any output.

2 $ **PSS/PLAYBACK/PARAM=(ABC,XYZ,54321) TXA3:**

This example shows a /PARAM qualifier with three parameters. Whenever PSS finds the %PARAM(1) lexical in the input script, it substitutes the string, **ABC**. It substitutes **XYZ**, when it finds the %PARAM(2) lexical. If it finds a %PARAM(4) lexical, when you have specified only three parameters, PSS terminates the playback session.

3 $ **PSS/PLAY/LOOP=4 TXA3:**

This command plays back the file, PTXA3.INP, on terminal TXA3 four times.

5.2.4 RECORD Command

The PSS/RECORD command records a terminal session. First, the only mandatory parameter is the terminal device. The device in a RECORD command must be a real terminal. You cannot specify a pseudo-terminal device. For RECORD, PSS captures the input- and output, terminal-data streams by default. The created file names are a manipulation of the terminal-device name.

In prior releases of the product, PSS provided response statistics for PLAYBACK sessions only. As of Version 3.3, PSS produces response statistics as a terminal session is being recorded. To enable this feature, the **/STATISTICS** qualifier has been added to the RECORD command's syntax. In addition, the **/SPLIT_TIMES** playback qualifier can now be specified for record sessions.

Example 5–3 RECORD Examples

```
1  $ PSS/RECORD TXA0:
```

This example captures both the input- and output-data streams for terminal, TXA0:, to a disk. The example names the disk files, PTXA0.INP, for input, and PTXA0.OUT, for output. The location of these files is your current VMS device/directory default.

```
2  $ PSS/RECORD/INPUT=TEST TXA0:
```

This example captures both the input- and output-data streams for terminal, TXA0:, to a disk. The example names the disk files TEST.INP, for input, and PTXA0.OUT, for output.

```
3  $ PSS/RECORD/INPUT=TEST/NOOUTPUT TXA0:
```

This example captures only the input-data stream for terminal, TXA0:, to a disk. The example names the input-disk file, TEST.INP.

```
4  $ PSS/RECORD/OWNER_PSS/PERMANENT TXA0:
```

This example captures both the input- and output-data streams for terminal, TXA0:, to disk. The owner UIC of the new disk files is [1,3], not your UIC/Identifier. In addition, the PSS/STOP command cannot terminate this session. The user on TXA0: must logoff before PSS automatically terminates recording activities on that terminal. Note

Example 5–3 Cont'd on next page

Example 5–3 (Cont.) RECORD Examples

that you can only specify /PERMANENT to a terminal that is **already** logged into VMS.

RECORD

This command records a terminal session's input- and/or output-data streams.

FORMAT

PSS/RECORD/qualifiers Terminal-device

Command Qualifiers	Defaults
/[NO]EIGHT	*/EIGHT*
/[NO]INPUT	*/INPUT*
/[NO]OUTPUT=file-spec	*/OUTPUT*
/OWNER_PSS	*See text*
/PASSWORD=string	*See text*
/PERMANENT	*Temporary*
/SPLIT_TIMES=value_list	*/SPLIT_TIMES=(1,2,5,30)*
/STATISTICS	*See text*
/USER_DELIMITER=(char,...)	*See text*

restrictions

You must execute the RECORD command for a valid, PSS-supported terminal device, which PSS does not currently use for other operations (such as a playback session). You cannot specify a pseudo-terminal device.

prompts

Terminal Device: terminal-device

PARAMETERS

terminal-device
Specifies the terminal device that you want to record

DESCRIPTION

The PSS/RECORD command records the input and/or output data of a specified, terminal device. PSS does not affect the program's interaction with that terminal. PSS captures the data in internally formatted files, which you can later use for playback or conversion.

COMMAND QUALIFIERS

/EIGHT (default)
/NOEIGHT

This qualifier allows you to independently record seven- or eight-bit characters, regardless of the terminal characteristics. /EIGHT is the default.

/INPUT[=file-specification] (default)
/NOINPUT

This qualifier directs PSS to record input data, entered into the system. The /NOINPUT qualifier directs PSS not to record or intercept input-terminal data.

By default PSS enables the /INPUT qualifier, using the default, file-specification rules, as discussed in Section 5.1.6.1.

/OUTPUT[=file-specification] (default)
/NOOUTPUT

This qualifier directs PSS to record output data, sent to the terminal device. The /NOOUTPUT qualifier directs PSS not to record or intercept output data to the terminal.

By default, PSS enables the /OUTPUT qualifier, using the default file specification rules, as discussed in Section 5.1.6.1.

/OWNER_PSS

This qualifier indicates that PSS create the input and/or output files under its start-up UIC. Normally, you use this feature with the /PERMANENT qualifier, which enhances overall system security. By default whoever executes the RECORD command owns the files.

/PASSWORD=string

Your PSS security administrator can as an additional security precaution require that you enter a password to positively identify yourself. If you must enter a password, PSS either prompts you for the password (with echo disabled), or you can enter the password as a value to this qualifier. ASCI recommends that you allow PSS to prompt for the password.

/PERMANENT

This qualifier indicates that the RECORD session must continue until the user of the RECORDed terminal signs off. The target terminal device *must* already be logged into VMS prior to using this qualifier. PSS/STOP commands are ignored, regardless of the source. This feature proves very useful, when you're recording a privileged user's terminal session. You can also use the /OWNER_PSS qualifier to further enhance system security.

/SPLIT_TIMES=[time-specification,...]

This qualifier allows you to specify the response time intervals (split times) that you are interested in tabulating. Enter these split-time intervals either as a whole number or as a fraction (up to a hundredths). The default split-times are, as follows:

- < 1 - less than 1 second
- 1-2 - 1 to 2 seconds, inclusive
- 2-5 - 2 to 5 seconds, inclusive
- 5-30 - 5 to 30 seconds, inclusive
- >30 - greater than 30 seconds

Enter the above defaults like this:

```
/SPLIT_TIMES=(1,2,5,30)
```

If you enter SPLIT_TIMES, PSS expects you to enter at least one, but no more than ten, split times. Note that this qualifier is ignored unless the /STATISTICS qualifier is also specified.

/STATISTICS

This qualifier indicates that the user wishes to generate response statistics for the RECORD session. By default, response statistics are not generated.

/USER_DELIMITER=values

PSS provides response-time statistics as one of its features during a playback session to make the session as realistic as possible. If you wait thirty (30) seconds between inputting fields, PSS must wait thirty seconds. To accomplish these two requirements, PSS uses an entity called a "delimiter table" with character values that terminate a query and create a program- or operating-system response. This definition fits in well with PSS requirements since PSS provides response-time data. PSS offers a default-delimiter table, which is the VMS standard-terminator table. (PSS considers all characters with a binary value of 0 to 31 terminators, including ESCAPE sequences). To specify your own delimiter table, use the **/USER_DELIMITER** qualifier, and specify a list of character mnemonics and/or decimal character values to create your own table. This table overrides the standard one. PSS does *not* append your table to it. This qualifier recognizes the following character mnemonics:

Table 5–2 User Delimiter Mnemonics

NUL	0	Null
TA	7	Bell
BS	8	Backspace
HT	9	Horizontal tab
LF	10	Line Feed
VT	11	Vertical Tab
FF	12	Form Feed
CR	13	Carriage Return
ESC	27	Escape

Note: PSS always considers ESCAPE a delimiter which it cannot disable.

EXAMPLES

1 $ **PSS/RECORD TXA3:**

Start recording TXA3 for both input (PTXA3.INP) and output (PTXA3.OUT).

2 $ **PSS/RECORD/NOINPUT TXA3:**

Start recording TXA3 for output (PTXA3.OUT). Do not record input.

3 $ **PSS/RECORD/INPUT=MY.INP TXA3:**

Start recording TXA3 for both input (MY.INP) and output (PTXA3.OUT).

4 $ **PSS/RECORD/USER_DELIM=(10,13) TXA3:**

This command enables the recording of terminal TXA3 for input and output. The qualifier, /USER_DELIM=(10,13), indicates that LINE FEED and CARRIAGE RETURN are the only PSS delimiters for determining response time.

5 $ **PSS/RECORD/STATISTICS TXA3:**

The statistics produced by the recorded session may be viewed by issuing a PSS REPORT command.

5.2.5 REPORT Command

This command obtains information, concerning the PSS terminal session.

Example 5–4 REPORT Example

```
1  $ PSS/REPORT/FULL PLAYBACK

Performance Simulation System (PSS) V031-000 for VMS on 1-APR-1989 14:52:57
Copyright (C) 1984, 1989 by Advanced Systems Concepts, Inc. All Rights
Reserved
Terminal LTA8  in Playback mode
Status= EOF, out, inp_wait, type, noeight, keyrate
Input file: DEV1:[BEN]TEST.INP;2
Output file: DEV1:[BEN]PLTA8.PLB;1
Response file: none
Input char count = 144         Output char count = 18740
Keyrate: 10/sec

Resp times     Bucket# 0   Bucket# 1   Bucket# 2   Bucket# 3   Bucket# 4
    <  1.00            0           0           0           0           0
1.00 - 2.00            8           0           0           0           0
2.00 - 5.00            4           0           0           0           0
5.00 - 30.00           0           0           0           0           0
    >  30.00            0           0           0           0           0
worst time    00:00:02.28  00:00:00.00  00:00:00.00  00:00:00.00  00:00:00.00

There are 1 terminals active: 0 in RECORD, and 1 in PLAYBACK (1 at eof).
```

REPORT

This command provides the reporting aspects of PSS.

FORMAT

PSS/REPORT/qualifiers Terminal-device

Command Qualifiers	Defaults
/BRIEF	*/BRIEF*
/FULL	*/BRIEF*
/OUTPUT=file-specification	*/OUTPUT=SYS$OUTPUT*

prompts

Terminal Device: terminal-device

PARAMETERS

terminal-device

Specifies the terminal device that you want to report

In addition to a valid VMS terminal device, you can use the following keywords:

ALL - report on all active PSS sessions (both record and playback)

PLAYBACK - report on all playback sessions

RECORD - report on all recording sessions

DESCRIPTION

The REPORT command obtains a brief or full listing of PSS terminal sessions. Based on the 'terminal-device' parameter, you can produce the following types of reports:

- On a specific terminal device
- On all RECORD sessions
- On all PLAYBACK sessions
- On all PSS sessions

Table 5–3 describes some of the status mnemonics, provided as part of the report.

Table 5–3 Mnemonics with Report Device

Status	Description
INP	Input characters are captured.
OUT	Output characters are captured.
EOF	Terminal is at input script's end of file.
INP_WAIT	Terminal is waiting for input.
USER_DELIM	User-Specified Delimiter Table is in use.
TIM	In RECORD mode, time stamp the next input character. In PLAYBACK mode, wait for the timer to expire before sending the next input character.
DRV_XOFF	PSS XOFF'd the terminal output, due to the high-server queue depth.
SYS_XOFF	VMS XOFF'd the terminal due to a no-read outstanding.
DISCONNECT	PSS is actively STOPping this unit.
PT_DEV	Unit is a Pseudo-Terminal device.
RESP	Unit is involved in an active, response-time check.
NO_READSYNC	Unit does not use READSYNC for data sync.
NOTIFY	PSS notifies you on completion of a playback session.
UNS_INP	Unit is inputting "unsolicited input" to VMS.
1STIM	Unit is in first-time-only processing.

If the terminal is in PLAYBACK mode, a response-time, bucket listing follows the first section of information. In seconds PSS analyzes and accumulates the response times as samples. For instance, the "less-than-one" field contains a number, which is the total number of occurrences in which PSS achieves this response time. The last line of the bucket indicates the worst-response time, seen as a time, in tenths of seconds. Bucket zero is the PSS delimiter, response-time block. If you specify any user-response-time buckets (via PSS File Conversion), PSS displays them, as well. The bucket number refers to the response-time ID you enter, when you insert your request.

COMMAND QUALIFIERS

/BRIEF (default)
/FULL

The /BRIEF qualifier provides a terse-report listing on PSS sessions. The /FULL qualifier provides a more verbose listing on PSS sessions. For terminal sessions in playback mode, /FULL provides response-time-bucket information, as well as input- and output-character counts—if you specify /OUTPUT during the playback session.

/OUTPUT[=file-specification]
/OUTPUT=SYS$OUTPUT (default)

The /OUTPUT qualifier allows you to place a report listing into any valid, VMS-file specification. This qualifier defaults to SYS$OUTPUT, which is the normal VMS-output specification.

If you enter /OUTPUT with a partial-file specification, such as /OUTPUT=[MYDIR], PSSRPT is the default-file name and RPT is the default-file type. When you enter a file specification, it must not include any wild-card characters.

EXAMPLES

1 $ **PSS/REPORT ALL**

```
$ PSS/REPORT ALL

Performance Simulation System (PSS) V031-000 for VMS on  1-APR-1989 14:54:37
Copyright (C) 1984, 1989 by Advanced Systems Concepts, Inc. All Rights Reserved
Device  Mode        Status
LTA7    Record      inp, out, tim
LTA8    Playback    EOF, out, inp_wait, type, noeight, keyrate

There are 2 terminals active: 1 in RECORD, and 1 in PLAYBACK (1 at eof).
```

5.2.6 SHUT Command

Use this command to stop all active PSS sessions and to then shutdown the PSS environment. To execute the PSS/SHUT command you must have OPER privilege.

SHUT

This command permits you to shutdown PSS.

FORMAT PSS/SHUT

restrictions This command requires OPER privilege.

DESCRIPTION

The SHUT command causes an implicit PSS/STOP ALL, followed by a shut down of PSS. The component, PSSRV, exits. However, PSS does not unload PTDRIVER, which remains unaffected by this command.

EXAMPLES

1 $ **PSS/SHUT**

This command stops all PSS sessions and then shuts down PSS. To restart PSS, execute the following command procedure:

```
$ @SYS$STARTUP:PSS_STARTUP
```

2 $ **PSS/SHUT/EXTERNAL**

This command stops all EPSS sessions and then shuts down EPSS. To restart EPSS, execute the following command procedure:

```
$ @SYS$STARTUP:EPSS_STARTUP
```

5.2.7 STOP Command

Use the PSS/STOP command to terminate PSS operations on a specified terminal or for all PSS terminal sessions. Use this command to perform any of the following functions:

- To terminate PSS activity for a recording session
- To terminate PSS activity for a playback session whose input script is at "end-of-file"
- To terminate PSS activity for a playback session prematurely

When you issue a PSS/STOP command, PSS terminates its activity with respect to the terminal and closes any files it has opened on behalf of that terminal session. This command might require up to five (5) seconds to successfully complete.

It is important to note that PSS/STOP does not affect any application or VMS process, associated with the terminal device. PSS/STOP only affects PSS operations.

The STOP command does not affect permanent RECORD sessions.

STOP

This command allows you to stop or abort one or more PSS terminal sessions.

FORMAT PSS/STOP Terminal-device

prompts Terminal Device: terminal-device

PARAMETERS *terminal-device*

Specifies the terminal device that you want to stop. In addition you can use the keyword, ALL, to stop every active PSS session both in RECORD or PLAYBACK modes.

DESCRIPTION

The STOP command stops PSS processing for an active terminal session (either recording or playing back) and/or disassociates an inactive terminal session from PSS. If you stop an active playback session that you started with /NOTIFY, an abort message appears.

EXAMPLES

1 $ **PSS/STOP TXA3:**

This example shows how to stop the terminal session to TXA3 and to close the PSS file(s), associated with this session.

2 $ **PSS/STOP ALL**

This example shows how to stop all currently running PSS sessions, except for those in PERMANENT RECORD mode.

6 USING EPSS

6.1 Overview of EPSS

EPSS provides the Remote Terminal Emulation facility of PSS, but uses a separate VAX computer system in addition to the actual System Under Test.

EPSS provides this facility by using terminal ports on the RTE machine which are directly connected to terminal ports on the SUT. This means output sent by the RTE is input to the SUT. The SUT "believes" there are real human operators entering data. In addition to testing the application's performance, this approach also provides real communcation and bus interaction.

6.1.1 Setting RTE Port Characteristics

Before beginning a playback under EPSS you must set some additional terminal characteristics. First, for all RTE terminal ports that you need you must set the terminal characteristics, within the following command:

```
$ SET TERM/PERM/NOTYPE/PASTHRU/NOTTSYNC terminal_device
```

Because of the special terminal-port connections, made between RTE and SUT, issue the above command *before* you enable interactive logins on a VMS start up.

6.1.2 Setting SUT Terminal Characteristics

The method by which EPSS determines whether data is solicited, is through the terminal characteristic, SCRIPT. Thischaracteristic generates a CTRL/A, when the SUT issues a Read QIO. This mechanism allows EPSS to determine, when a program on the SUT is ready for input data. Therefore, you must set all SUT terminals, participating with the RTE,as follows:

```
$ SET TERM/PERM/SCRIPT terminal_device
```

You may set other terminal characteristics, as well.

Note: You must ensure that nothing in your playback script, either directly or indirectly, clears the SCRIPT characteristic. Certain forms of the SET TERMINAL command have an indirect effect on SCRIPT.

6.1.3 Method 1: Direct Connection

This method requires that you connect the direct-connect, terminal ports on the RTE to the terminal ports on the SUT through a series of "null" modem cables. This ensures that the SUT receives data, transmitted by the RTE. Since data that the SUT sends becomes input for the RTE, set the RTE's terminal ports, NOTYPEAHEAD. Disabling TYPEAHEAD prevents a broadcast message (or noise), which the SUT sends to the RTE, from creating a CPU-consuming loop. Such a loop can initiate a LOGIN sequence on the RTE.

6.1.4 Method 2: Connection to a Terminal Server

The mechanics of this method are similiar to Method 1, except that you make the connections to a Terminal Server (Figure 6–1). The Terminal Server *believes* these connections are terminals. However, you should still disable TYPEAHEAD to prevent an endless loop. In addition you must also include Terminal-Server commands in the input script to indicate which node you want to act as the SUT.

Figure 6–1 SUT and RTE with Terminal Servers

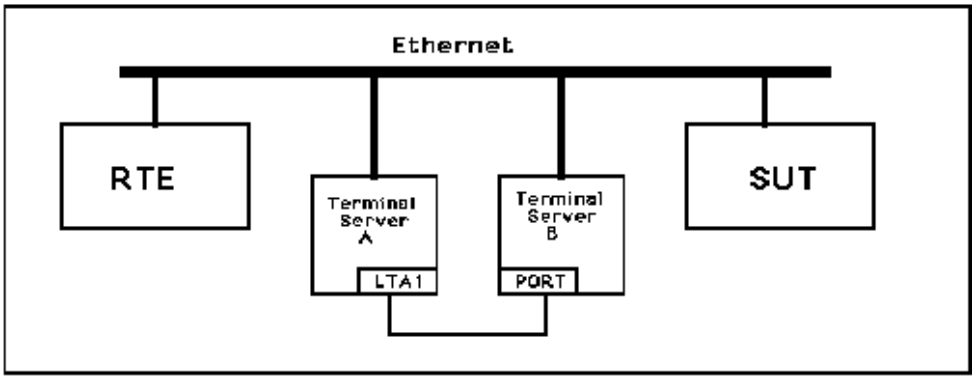


Figure 6–1 illustrates the use of an RTE and SUT, using a Terminal-Server arrangement. The RTE system creates a LAT application-port, named LTA1 (which references Terminal Server A, Port 1). A null-modem cable connects that port to a port on Terminal-Server B (Multiple port connections would require multiple null-modem cables.)

The input script on the RTE must contain input for the Terminal-Server prompts—for example, "Username" and "LOCAL>." The last prompt in particular would designate the node, which acts as the SUT. While the use of Terminal Servers requires dedicated-port pairs, this technique offers the most flexibility at the lowest cost. Any node can act as RTE and SUT. Example 6–1 illustrates the sample settings of a LAT port for use by the RTE system, and the settings of the LAT port for a SUT system. Note that the SUT Server port uses the "dedicated" feature of the LAT Terminal Server. This helps, when running existing scripts through EPSS, since the LOCAL dialogue, described above, will not occur. Depending on your LAT Terminal-Server model, you might need to change the SERVER QUEUE LIMIT attribute to NONE from its default (DECserver 200 - 24, DECserver 500 - 8). If not, when you reach the limit of host-initiated connections, EPSS will receive SYSTEM-F-HANGUP errors from the Terminal Server for each new connection.

6.1.5 Method 3: Connection via Outbound LAT Support

In Version 5.4-2 (and later releases), VMS provided support for forward, or outbound, LAT. Using outbound LAT, a VMS system can act like a terminal server to generate outgoing connections to other nodes within a local area network. The EPSS product can use the outbound LAT facility, eliminating the need for hard-wired connections.

Example 6–1 Sample RTE/SUT LAT Port Server Characteristics

```
--------------------RTE Server Characteristics--------------------
Port 1:  (Remote)

Character Size:            8          Input Speed:        9600
Flow Control:            XON          Output Speed:       9600
Parity:                 None          Modem Control:  Disabled

Access:                Remote          Local Switch:       None
Backwards Switch:          ^\          Name:            RTE_001
Break:                 Local          Session Limit:          4
Forwards Switch:           ^_          Type:               Hard

Preferred Service: None

Authorized Groups:   0
(Current)  Groups:   0

Enabled Characteristics:

Input Flow Control,  Output Flow Control,  Queuing

--------------------SUT Server Characteristics--------------------
Port 1:

Character Size:            8          Input Speed:        9600
Flow Control:            XON          Output Speed:       9600
Parity:                 None          Modem Control:  Disabled

Access:                Local          Local Switch:       None
Backwards Switch:          ^\          Name:             PORT_1
Break:                 Local          Session Limit:          4
Forwards Switch:        None          Type:               Ansi

Dedicated Service: SAMPLE

Authorized Groups:   0
(Current)  Groups:   0

Enabled Characteristics:

Autoprompt,  Broadcast,  Input Flow Control,  Loss Notification,
Message Codes,  Output Flow Control,  Verification
```

Note: To use the outbound LAT facility, you must be running VMS 5.4-2, or later. If you are running VMS 5.4-2, the optional LATUn054 (LATmaster) kit must be installed. The LATmaster software is integrated into VMS 5.5 and later.

EPSS sessions were established, in prior versions of the product, by issuing PLAYBACK commands to RTE terminal ports. These ports were, in turn, hard-wired to terminal ports on the SUT. For outbound LAT connections, however, the target of the PLAYBACK is a LAT **service**, rather than a terminal.

A LAT service is a resource that is available to members of a local area network. A VMS system can be set up as a LAT **service node** - in general, your DECnet node names will be synonymous with their LAT service names. However, you may also define a service to be a specific application on a specific node.

When a PLAYBACK command is issued, EPSS will invoke outbound LAT services, causing a LAT terminal device to be "cloned" on the RTE and connected to the SUT. EPSS will send all input to and receive all output

from this device. Depending on the qualifiers entered on the PLAYBACK command, a new LAT device may also be "cloned" on the SUT.

When your playback script has completed, you can issue the normal PSS STOP command to stop the EPSS session. STOP command processing will break the connection to the SUT and remove the cloned LAT device from the RTE.

For detailed information on LAT services, refer to the *VMS LAT Control Program (LATCP) Manual*.

To support outbound LAT connections, the syntax of the EPSS PLAYBACK COMMAND has been modified and extended. The PLAYBACK command has many qualifiers, but only one parameter. For hard-wired connections, this parameter identifies the terminal port. For outbound LAT connections, the parameter identifies the EPSS session and is used mainly for file naming purposes, as defined in the *PSS User's Guide*.

The **/SERVICE** qualifier has been added to the PLAYBACK command definition. The value entered into this qualifier must be the name of the LAT service to which you want to connect. This qualifier is required for outbound LAT. If it is omitted, EPSS will assume that the parameter specifies a terminal port and will attempt a hard-wired connection.

```
$ PSS/PLAYBACK/EXT/SERVICE=VARMINT/INPUT=VERIFY TEST1
```

This command will start up a playback to the LAT service VARMINT using input file VERIFY.INP.

The **/PORT** qualifier is used to identify a specific logical port that has been dedicated to an application service. The qualifier is optional but, if it is specified, the /SERVICE qualifier must also be present.

```
$ PSS/PLAYBACK/EXT/SERVICE=POLECAT/PORT=LTA2023 TEST2
```

This command will playback file PTEST2.INP to logical port LTA2023, which has been dedicated to application service POLECAT.

The same service may be offered by several nodes in a local area network. If only the /SERVICE qualifier is specified in this situation, LAT will connect the Playback session to the least-busy node. If, however, you want to force the connection to a specific node, you may specify the **/NODE** qualifier.

```
$ PSS/PLAYBACK/EXT/SERVICE=TESTSERV/NODE=CRITTER TEST3
```

This command will start up a playback to the TESTSERV service on node CRITTER.

6.2 Invoking EPSS

EPSS uses the same command set as PSS with the major exception of the RECORD command. To access the EPSS subsystem, specify the qualifier, /EXTERNAL, with this command:

```
$ PSS/qualifier/EXTERNAL Terminal/param-qualifier
```

For example, to start a playback on TXA3, enter:

```
$ PSS/PLAYBACK/EXTERNAL TXA3
```

This command starts a playback session on TXA3, using the input-script file, PTXA3.INP. It captures no output.

EPSS expects you to have already created an input script either by using the PSS/RECORD command or through an editor. Chapter 5 and Chapter 7 explain this process. Remember that your input script must include any login dialogue, necessary to achieve a successful playback session, since the SUT terminal ports are **not** logged in.

As with PSS, EPSS applies the same rules to determine, whether it can send "input" data to an SUT. EPSS and PSS always expect data to be solicited. That means the program on the SUT must issue a Read QIO to "solicit" the input data. For example, DCL always posts a Read QIO to read your VMS command. The only exception PSS makes to this solicited-input rule is on the initial playback line in your script. That line, beginning with the first character, is sent to the SUT, regardless of whether it has been solicited. This feature allows you to begin a LOGIN sequence (which requires that the first character sent be unsolicited) without having to edit your input script. The %BEGIN_UNSOLICITED lexical is available for other actions that require EPSS to send unsolicited data to the SUT.

PLAYBACK

This command plays back an input script to a specific terminal.

FORMAT

PSS/PLAYBACK/EXTERNAL/qualifiers object

Command Qualifiers	Defaults
/INPUT[=file-specification]	*See text*
/KEYRATE=value	*/KEYRATE=10*
/LOOP=value	*/LOOP=1*
/NODE=node_name	*See text*
/NOTIFY	*/NONOTIFY*
/[NO]OUTPUT=file-spec	*/NOOUTPUT*
/PARAM=parameters	*See text*
/PORT=port_name	*See text*
/RESPONSE=file_spec	*/NORESPONSE*
/SEED=value	*See text*
/SERVICE=service_name	*See text*
/SPLIT_TIMES=value_list	*SPLIT_TIMES=(1,2,5,30)*
/VARY[=factor]	*/VARY=1*

restrictions

You must execute the PLAYBACK command to a valid RTE terminal device or LAT Service.

prompts

Terminal or Session ID: object

PARAMETERS

object

Specifies the terminal device that you want to playback through on the RTE for hard-wired connections, or the name of the outbound LAT session.

DESCRIPTION

Use the EPSS PLAYBACK command to begin an interactive exchange between EPSS and a SUT. EPSS reads all relevant input data from an input script file and sends these to the SUT, using similiar wait times, encountered during the script generation. The terminal output can be captured and written to a disk file for later analysis.

An optional ASCI product, named **WATCH**, allows you to watch the playback session to ensure that the SUT responds correctly.

COMMAND QUALIFIERS

/INPUT[=file-specification]

This qualifier represents the input script file. By default, /INPUT is enabled, using the default-file-specification rules, as discussed in Section 5.1.6.1. Optionally, you can replace the default with a VMS file specification.

/KEYRATE=value
/KEYRATE=10 (default)

This qualifier allows you to specify the typing rate, per second, at which the terminal receives input.

The value can be any integer within the range of 0 to 100. If you don't specify this qualifier (or you specify a value of less than ten), you get the default rate of ten characters per second.

/LOOP=value
/LOOP=1 (default)

This qualifier directs EPSS to execute "n" loops of the input-file script. This numeric value, if specified, must be between 1 and 32,767. By default, /LOOP=1.

/NODE=node_name

This qualifier specifies the name of the node to which you want to direct your outbound LAT session. The qualifier is ignored if the /SERVICE qualifier is not specified.

/NOTIFY
/NONOTIFY (default)

This qualifier causes EPSS to notify whoever executes the PLAYBACK command, via broadcast, when the input script reaches end-of-file, is aborted via the PSS/STOP/EXTERNAL command, or is terminated as a result of a run-time error.

/OUTPUT[=file-specification]
/NOOUTPUT (default)

This qualifier directs EPSS to capture output data sent to the terminal device. /NOOUTPUT, which is the default, directs EPSS not to intercept output data to the terminal.

/PARAM=parameters

The /PARAM qualifier allows you to pass run-time parameters to the script as it plays back. EPSS uses these parameters in conjunction with %PARAM lexicals, which the script contains. Each lexical has a number that refers to the position of a parameter within the /PARAM list. When EPSS encounters the %PARAM lexical during playback processing, it substitutes the corresponding parameter from the list. You can enter a maximum of ten (10) parameters via the DCL command line.

A useful feature of this qualifier allows you to specify the VMS Username and Password as parameters to those input scripts that contain logon sequences.

/PORT=port_name

This qualifier specifies the name of the LAT port to which you want to direct your outbound LAT session. This qualifier is ignored if the /SERVICE qualifier is not specified.

/RESPONSE[=file_specification]

The /RESPONSE qualifier creates a response-time-event file for your current-playback session. For more information, see Section 7.4.

/SEED=value

This qualifier, working with the %RANDOM and the %RANDOM_ TIME lexicals, allows you to specify the starting value, which a random-generator algorithm uses to generate random numbers.

When you specify this qualifier, the random-number generator always generates the same set of random numbers. Therefore, when you rerun multiple playback sessions, you can compare the results of the different test runs. By default a new seed value is used for each playback session.

/SERVICE=service_name

This qualifier specifies the name of the LAT service to which you want to direct your outbound LAT session. It is required for outbound LAT connections.

/SPLIT_TIMES=[time-specification,...]

This qualifier allows you to specify the response time intervals (split times) that you are interested in tabulating. Enter these split-time intervals either as a whole number or as a fraction (up to a hundredths). The default split-times are, as follows:

- < 1 - less than 1 second
- 1-2 - 1 to 2 seconds, inclusive
- 2-5 - 2 to 5 seconds, inclusive
- 5-30 - 5 to 30 seconds, inclusive
- >30 - greater than 30 seconds

Enter the above defaults like this:

```
/SPLIT_TIMES=(1,2,5,30)
```

If you enter SPLIT_TIMES, PSS expects you to enter at least one, but no more than ten, split times.

/VARY[=factor]
/VARY=1 (default)

The /VARY=factor qualifier varies the recorded waiting times between input items. The "factor," noted above, is a floating-point number that EPSS multiplies against the wait time. To cut wait times in half, specify the qualifier as /VARY=.5. Use the command, /VARY=2 to double the wait times. The default vary factor is one (1). To disable wait times, enter the command, /NOVARY.

EXAMPLES

1 $ **PSS/PLAYBACK/EXTERNAL TXA3:**

Start a playback to RTE_port TXA3, using the input file, PTXA3.INP. It does not intercept any output.

2 $ **PSS/PLAYBACK/EXTERNAL/PARAM=(ABC,XYZ,54321) TXA3:**

This example shows a /PARAM qualifier with three parameters. Whenever EPSS finds the %PARAM(1) lexical in the input script, it substitutes the string, "ABC," and it substitutes "XYZ," when it finds the %PARAM(2) lexical.

3 $ **PSS/PLAY/EXTERNAL/LOOP=4 TXA3:**

This command plays back file, PTXA3.INP, on RTE_port TXA3 four times.

4 $ **PSS/PLAYBACK/EXTERNAL/SERVICE=VARMINT TESTA**

Start a playback to LAT service VARMINT, using the input file, PTESTA.INP.

5 $ **PSS/PLAYBACK/EXTERNAL/PORT=LTA1001/SERVICE=POLECAT TESTB**

Start a playback to port LTA1001 on LAT service POLECAT, using the input file, PTESTB.INP.

6 $ **PSS/PLAYBACK/EXT/PORT=LTA2023/SERVICE=TESTSERV/NODE=CRITTER TESTC**

Start a playback to port LTA2023 of LAT service TESTSERV on node CRITTER, using input file, PTESTC.INP.

6.3 Playing back DCL scripts

If your test scripts contain DCL commands and involve logging into the SUT, you need only specify the /SERVICE qualifier on your PLAYBACK commands. The service name you supply must be that of the LAT service node you want to test. For example, consider the following EPSS script:

```
%TIME(00:00:02.00)%PARAM(1)<CR>
%TIME(00:00:02.00)%PARAM(2)<CR>
%TIME(00:00:01.00)SHOW TIME<CR>
%TIME(00:00:05.00)SHOW SYSTEM<CR>
%TIME(00:00:02.00)SHOW MEMORY<CR>
%TIME(00:00:02.00)SHOW USERS<CR>
%TIME(00:00:01.00)SHOW TIME<CR>
%TIME(00:00:02.00)LOGOFF<CR>
```

This script will log into VMS, issue a few DCL commands, and then log off. The two %PARAM lexicals are used to supply the VMS Username and Password of the user logging in. For purposes of this example, assume the following:

- the name of the script file is DCLTEST.INP
- the name of the node to test is VARMINT
- on VARMINT, a VMS account exists for user TESTUSER, with a password of TESTPASS

The PLAYBACK command needed to run the test is:

```
$ PSS/PLAYBACK/EXTERNAL/INPUT=DCLTEST/OUTPUT -
     /SERVICE=VARMINT/PARAM=(TESTUSER,TESTPASS) TEST1
```

Note that this command requests EPSS to capture the output from the terminal session. Since no file specification was supplied with the /OUTPUT qualifier, the default file specification rules are used and the *session id* is used to form the name of the output file. The file specification formed is PTEST1.PLB.

When the script is run, it will appear to the SUT as if a terminal server is requesting a connection. The LAT software on the VARMINT node will create a new LAT terminal for TESTUSER to log into. The *SCRIPT* terminal characteristic has to be set for this terminal. However, since the terminal is "created" when the user logs in, it is not possible to preset the terminal before running the script. Therefore, the SET TERMINAL command must be issued in TESTUSER's LOGIN.COM file.

The following DCL command illustrates the method of setting the terminal:

```
$ SET TERMINAL/SCRIPT
```

Including this command in the LOGIN.COM files of your test accounts will ensure that your scripts run correctly.

6.3.1 Problems with SET TERMINAL/INQUIRE

The standard *SET TERMINAL/INQUIRE* command is used to set the device type based on a response elicited from the terminal. In many cases, this command is included in the LOGIN.COM file to automatically detect the terminal type at login.

Since the response to the *SET TERMINAL/INQUIRE* command is generated by the terminal hardware, you may experience problems when playing back scripts that include login sequences. After it is issued, the *SET TERMINAL/INQUIRE* command expects the next input item to be its response. With an EPSS play back, there is no phyical terminal to generate a response, so the *SET TERMINAL* command will "eat" the next item of input, possibly throwing off the logic of your script. To get around this problem, you can include the appropriate response in your input script. It should immediately follow the login sequence. The response is in the form of a Report ESCAPE sequence that identifies the terminal and its options.

For the script, illustrated above, including the response sequence following the "password" parameter will satisify the *SET TERMINAL/INQUIRE*. Otherwise, the *SHOW TIME* command would be taken as the response and would appear to be missing from the script:

```
%TIME(00:00:02.00)%PARAM(1)<CR>
%TIME(00:00:02.00)%PARAM(2)<CR>
%TIME(00:00:05.00)<ESC>[?63;1;2;3;4;6;7;8;9;13;15;16;18;19c<CR>
%TIME(00:00:01.00)SHOW TIME<CR>
%TIME(00:00:05.00)SHOW SYSTEM<CR>
%TIME(00:00:02.00)SHOW MEMORY<CR>
%TIME(00:00:02.00)SHOW USERS<CR>
%TIME(00:00:01.00)SHOW TIME<CR>
%TIME(00:00:02.00)LOGOFF<CR>
```

6.4 Playing back Non-DCL scripts

You may have test scripts that are not DCL-oriented, but are used to test specific applications. The terminals to which you play these scripts must exist prior to the test and must be dedicated to the application. If this is the case, then, on the SUT, you must use the VMS LAT Control Program (LATCP) to create a LAT **application service**, create one or more LAT ports, and dedicate the ports to the service.

Assume you want to test an application to see how it performs when handling 100 dedicated terminals. On the SUT machine, first create an application service called, for example, APPLSERV:

```
$ RUN SYS$SYSTEM:LATCP RETURN
LATCP> CREATE SERVICE APPLSERV/APPLICATION RETURN
```

Next, create the 100 terminal ports in the range LTA1000 to LTA1099:

```
LATCP> CREATE PORT LTA1000:/DEDICATED RETURN
LATCP> CREATE PORT LTA1001:/DEDICATED RETURN
                    .
                    .
                    .
LATCP> CREATE PORT LTA1099:/DEDICATED RETURN
```

Finally, dedicate each of the ports to the APPLSERV service:

```
LATCP> SET PORT LTA1000/DEDICATED/SERVICE=APPLSERV RETURN
LATCP> SET PORT LTA1001/DEDICATED/SERVICE=APPLSERV RETURN
                    .
                    .
                    .
LATCP> SET PORT LTA1099/DEDICATED/SERVICE=APPLSERV RETURN
LATCP>  CTRL/Z
```

At this point, the 100 terminals will exist on the SUT and will be associated with the APPLSERV service. Now, set the SCRIPT characteristic for each terminal with the SET TERMINAL command:

```
$ SET TERMINAL/PERMANENT/SCRIPT LTA1000 RETURN
$ SET TERMINAL/PERMANENT/SCRIPT LTA1001 RETURN
          .
          .
          .
$ SET TERMINAL/PERMANENT/SCRIPT LTA1099 RETURN
```

Note: You may receive a *SYSTEM-F-HANGUP, data set hang-up* error message when setting dedicated application ports. The SCRIPT characteristic will have been set correctly, even so.

You may now run your application program, assigning all the terminal ports to it.

On the RTE machine, you may now begin to run your test. If the test script is called APPLTEST.INP, the following PLAYBACK command will run a test:

```
$ PSS/PLAYBACK/EXTERNAL/SERVICE=APPLSERV/INPUT=APPLTEST ATEST1
```

The preceding form of the PLAYBACK command will run the test on the first available port dedicated to the APPLSERV service. To use all the ports, simply issue the command 100 times. If you want to run to a specific port, include the /PORT qualifier in the command:

```
$ PSS/PLAYBACK/EXTERNAL/SERVICE=APPLSERV -
     /PORT=LTA1023/INPUT=APPLTEST ATEST1
```

When you have completed your tests, you may remove the service and ports from your SUT by the following, LATCP commands:

```
$ RUN SYS$SYSTEM:LATCP RETURN
LATCP> DELETE PORT LTA1000: RETURN
LATCP> DELETE PORT LTA1001: RETURN
                      .
                      .
                      .
LATCP> DELETE PORT LTA1099: RETURN
LATCP> DELETE SERVICE APPLSERV RETURN
LATCP>  CTRL/Z
```

For complete details on LATCP commands and facilities, refer to the *VMS LAT Control Program (LATCP) Manual*.

6.5 Restrictions

There are several restrictions to using the Outbound LAT facility that you should be aware of:

- Outbound LAT can only be used on RTE's and SUT's that reside in the same local area network.
- Because the PSS STOP command disconnects the link between RTE and SUT and deletes the "cloned" terminals, your entire test must be contained within one EPSS session. For example, some users use one script to log into the SUT, another to run an application, and a third to logoff. Each of the scripts are run using separate PLAYBACK commands, with STOP's in between. This is fine for hard-wired connections, but, with outbound LAT, the connection is broken by the first STOP and a new one established by the next PLAYBACK. To perform this function using outbound LAT, all your commands should be contained in one script or the %EXEC lexical may be used to invoke scripts from a main script.
- A VMS LOGOFF command causes a LAT terminal to disconnect. If a script contains a LOGOFF command, its playback session will terminate as soon as the LOGOFF is executed. The /LOOP qualifier or %LOOP lexical will not work for these types of scripts.

7 FILE CONVERSION UTILITY

7.1 Overview

The PSS-conversion utility allows you to convert files from PSS-internal format to a form suitable for use by VMS facilities. It supports the following conversion operations:

- Converting the PSS-Recorded-Input format into Text
- Converting the Text to PSS-Recorded-Input format
- Converting the PSS-Captured-Output format into Text
- Converting the PSS-Response-Event format to RMS Sequential

Note: PSS does not support the conversion from text to PSS-Captured-Output format.

The text files, produced by the PSS conversion utility, are standard RMS sequential files with variable length records and implicit-carriage control. This allows you to use the converted-text files as input to such VMS utilities as TYPE, PRINT, or any VMS editor.

Conversion from text to recorded-input format permits you to edit PSS-recorded-input files and play back the modified scripts. PSS contains several useful playback features that you can access only after adding several of the lexicals, discussed in Section 7.3.1.

7.2 Invoking PSS Conversion Utility

The DCL command, PSS/CONVERT, invokes the PSS conversion utility. The format for the command is, as follows:

```
$ PSS/CONVERT/qual  from_file/type  to_file/type
```

Table 7–1 Command qualifiers

/TRANSLATE	default=no
/ABSOLUTE	default=no
/RECORDSIZE	default=2048 bytes

Table 7–2 Type qualifiers - One of the Following:

/INPUT
/OUTPUT
/PLAYBACK
/RESPONSE
/TEXT

Table 7–3 Default extensions:

Type	Extension
/INPUT	.INP
/OUTPUT	.OUT
/PLAYBACK	.PLB
/RESPONSE	.RSP
/TEXT	.TXT

Only /TEXT and /INPUT types are valid for the to_file parameter.

Example:

```
$ PSS/CONVERT   PTXA5/INP   TT:
```

This command converts a recorded-input file (PTXA5.INP) into ASCII text and displays it on the current terminal.

```
$ PSS/CONVERT   PTXA5/INP   SCRIPT/TEXT
```

This command converts a recorded-input file (PTXA5.INP) into ASCII text and stores it within SCRIPT.TXT.

Figure 7–1 PSS Conversion Utility Listings

```
$ PSS/CONVERT PLTA11/INPUT PLTA11/TEXT
%PSSCVT-I-RECINP, 1  records read from $1$DUA13:[BEN]PLTA11.INP;1
%PSSCVT-I-RECOUT, 9  lines written to $1$DUA13:[BEN]PLTA11.TXT;1
$ PSS/CONVERT PLTA11/TEXT TEST_SCRIPT/INPUT
%PSSCVT-W-LEXERR, lexical syntax error on line 8
-PSSCVT-W-RSPNOTENA, specified response-time bucket not enabled
%END_RESPONSE(1)

(..............)
%PSSCVT-I-RECINP, 10  lines read from $1$DUA13:[BEN]PLTA11.TXT;2
%PSSCVT-I-RECOUT, 2  records written to $1$DUA13:[BEN]TEST_SCRIPT.INP;1
%PSSCVT-W-DIAGS, 1 warnings issued
```

7.3 PSS Input Conversion Text Commands

Several features of PSS require that you edit the recorded-input file. However, before you do, you must use the PSS/CONVERT utility to convert the PSS-internal format to that of an ASCII-text file, as follows:

```
$ PSS/CONVERT  From_file/INP   To_file/TEXT
```

You now have a file that you can edit with any VMS editor you choose.

PSS provides several lexical functions for input and response-time handling.

Table 7–4 Lexicals

Lexical Name	Description
%BEGIN_CHECKPOINT(n)	Starts User Response Checkpoint "n"
%BEGIN_UNSOLICITED( )	Starts Unsolicited Input
%COMMENT(text)	Provides text as comment within script
%CUR_LOOP(ident)	Captures current %DO_LOOP value
%DO_LOOP(start,end,step,ident)	Repeats specified portion of script
%ENDDO_LOOP(ident)	Marks the end of the repeated script
%END_UNSOLICITED( )	Ends Unsolicited Input
%END_CHECKPOINT(n)	Ends User-Response Checkpoint "n"
%EXEC(filespec,args...)	Executes PSS script, "filespec"
%EXPER(dd-mmm-yyyy hh:mm:ss.cc)	Provides the date and time this PSS script recorded
%GOTO(label)	Transfer script control to named label (EPSS Only)
%INDEX(n)	Substitutes unique, numeric string, "n" digits
%LABEL(name)	Defines a label which can be used to transfer control to (EPSS Only)
%LOCAL_PARAM(n)	Substitutes lexical with Local Parameter "n" from %EXEC lexical
%MAIL(mailbox-name,text)	Sends text data to user-control/monitor program
%MATCH("string",label)	Checks output from SUT for match and allows transfer of control (EPSS Only)
%OUTPUT(text)	Produces text output to terminal (Text can contain imbedded-control characters and/or escape sequences.)
%PARAM(n)	Substitutes lexical with Global Parameter "n" from /PARAM=(p1, p2...pn) list.
%RANDOM(lower,upper,format,fill)	Generates random number within specific format with optional zero fill
%RANDOM_TIME(lower,upper)	Generates random-waiting time between specified boundaries
%SUSPEND(text)	Prints message and supports playback exexcution for terminal input
%TIME(hh:mm:ss.cc)	Waits before sending input

Note: This chapter assumes the use of EDT as the editor of choice. However, you can use any editor, such as TPU or LSE, which allows you to enter control characters.

The lexicals, listed on the following pages,can appear anywhere in the text file. You can create a text file from scratch with an editor or via the PSS/CONVERT utility. PSS requires that you specify only the unique portion of the lexical name (for example, %BEGIN_C(1)), which you can enter either in upper or lower case. EDT represents control characters as a corresponding pair of angle brackets. Within the brackets, you need a two-character abbreviation, indicating the control-character value. Certain, familiar, control characters have abbreviations, such as <CR> - Carriage Return, <LF> - Line Feed, <ESC> - Escape. A control figure, CONTROL, followed by a letter, represents all other control characters. ^Z describes Control/Z.

Place each input record on its own line for convenience and readability. PSS does not use the implied carriage return after each line. To input control characters within each record, simply type:

```
GOLD   decimal.char_value   GOLD keypad 3
```

This inserts the actual-character value within the text.

The PSS conversion utility allows you to convert the output or playback-output logs from PSS-internal format to an ASCII-text file which VMS utilities can manipulate. During conversion to a text file from an output log, PSS attempts to limit the record size to 2,048 characters. Note that this 2,048-byte limit does not apply, if you are sending the input file directly to a terminal, rather than converting it into a file. PSS can make control characters, which are normally invisible, *visible* (such as escape sequences), through use of the /TRANSLATE qualifier. PSS prints control characters as <^letter> or in a more familiar form, if one exists, such as CR, CF, ESC, or DEL.

Figure 7–2 Output Text Conversion

```
$ PSS/CONVERT/TRANSLATE PLTA11/INP TT:
%EXPER(1-APR-1989 16:31:43.90)
%TIME(00:00:00.00) <CR>
%TIME(00:00:00.16) <CR>
%TIME(00:00:00.48)SHO MEM <CR>
%TIME(00:00:02.32)SHO SYS <CR>
%TIME(00:00:02.88)SHO MEM <CR>
%TIME(00:00:04.08)SHO MEM <CR>
%TIME(00:00:02.08)SHO SYS <CR>
%PSSCVT-I-RECINP, 1  records read from $1$DUA13:[BEN]PLTA11.INP;2
%PSSCVT-I-RECOUT, 8  lines written to TT:[]PSSCVT.TXT;
```

The /ABSOLUTE qualifier converts the %TIME lexicals, found in an input file, to the absolute-system times at which these input items are executed. In the text file that it produces, the absolute times appear within %ABSTIME lexicals. It computes these absolute times, based upon the system time in the %EXPER (experiment time) lexical. You may specify %EXPER as the first lexical in the input file. Figure 7–3 shows how to convert an input file, using the /ABSOLUTE qualifier and containing the following data:

Figure 7–3 Input File Converted with /ABSOLUTE qualifier

```
$ PSS/CONVERT/ABSOLUTE/TRANSLATE PLTA8.INP TT:
%EXPER(18-AUG-1988 16:31:43.90)
%ABSTIME(1-APR-1989 16:31:43.90) <CR>
%ABSTIME(1-APR-1989 16:31:44.38)SHO MEM <CR>
%ABSTIME(1-APR-1989 16:31:46.46)SHO SYS <CR>
%ABSTIME(1-APR-1989 16:31:46.46)MONITOR PROC <CR>
%BEGIN_UNSOLICIT()
%ABSTIME(18-AUG-1989 16:32:16.46)<^Y>
%END_UNSOLICIT()
```

Note: You cannot convert a text file, produced with either the /TRANSLATE or /ABSOLUTE qualifiers, to an input file for subsequent playback. These qualifiers produce only documentation.

7.3.1 PSS Lexical Summary

%BEGIN_CHECKPOINT—Start User-Response Checkpoint

This lexical provides you with response-time-bucket handling.

FORMAT

%BEGIN_CHECKPOINT *(bucket)*

PARAMETERS

bucket
A number, currently from 1 to 10, into which PSS counts various levels of user-response time

DESCRIPTION

PSS maintains up to ten response-time intervals for use in recording statistics kept on response time during a playback session. You can specify your own response-time intervals (see /SPLIT_TIMES) or use the default, as follows:

- Less than one (1) second
- One - Two seconds
- Two - Five seconds
- Five - Thirty seconds
- Thirty seconds and greater

You can specify your own response times via the /SPLIT_TIMES qualifier on the playback command, as follows:

$ PSS/PLAYBACK/SPLIT_TIMES=(1,2,3,4,5,6,7,8,30)/INPUT=TEST TT:

This generates response intervals with the following information:

- Less than one (1) second
- One - Two seconds
- Two - Three seconds
- Three - Four seconds
- Four - Five seconds
- Five - Six seconds
- Six - Seven seconds
- Seven - Eight seconds
- Eight - Thirty seconds
- Thirty seconds and greater

PSS keeps the response statistics as a number of samples that fall into one of the above categories. The last category, which it keeps in VMS-time format (hh:mm:ss), represents the "worst response time seen." PSS uses the first bucket solely for response time, based upon service-level measurement of the individual input-script records. While delimiter-response time may prove useful, most applications which use a forms editor (FMS, TDMS, DECintact) do not require a "field" response time, but instead need a "form" (or collection of forms) response time.

PSS allows you to collect up to ten (10) individual response buckets per playback session. To use this feature, simply edit the following lexicals into your text file:

- %BEGIN_CHECKPOINT(n) — Start Response Bucket "n"
- %END_CHECKPOINT(n) — End Response Bucket "n"

The "n" is a value from 1 to 10. You can place these lexicals, like most of the others, anywhere. However, you usually do not want to place this lexical before a %TIME lexical, since the response-time check begins when PSS encounters the %BEGIN_CHECKPOINT lexical.

EXAMPLES

1
```
%TIME(00:00:05)%BEGIN_CHECKPOINT(1)
SHOW SYSTEM<CR>%END_CHECKPOINT(1)
```

This example measures the time a "SHOW SYSTEM" DCL command takes to execute. By nesting checkpoints you can measure the response in greater or finer detail.

2
```
%TIME(00:00:05)%BEGIN_CHECK(1))
SHOW SYSTEM <CR>
%BEGIN_CHECK(2)
@DTREXAMPL
%END_CHECK(2)
%END_CHECK(1)
```

You can nest the BEGIN_CHECKPOINT lexical within another BEGIN-END CHECKPOINT. This stream has CHECK(1) measure the time both the DCL-command SHOW SYSTEM and a DCL-command procedure take, while CHECK(2) only measures the DCL-command procedure.

%BEGIN_UNSOLICITED—Begin Unsolicited Data

This lexical indicates the beginning of unsolicited data.

FORMAT %BEGIN_UNSOLICITED *()*

DESCRIPTION

Normally, PSS uses READSYNC to determine, when your program (or the system) wants the input data that PSS has for it. However, there are certain types of programs and situations, wherein an "unsolicited" character begins a program action. For example, typing carriage return begins the LOGIN process. The LOGIN command does not solicit the carriage return. Programs, operating under VMS, can take advantage of this "poke in the shoulder" to begin processing.

While PSS allows your first input-data character to be unsolicited, (for easier LOGIN use), you need to use the %BEGIN_UNSOLICITED and %END_UNSOLICITED lexicals for embedding unsolicited data within the input script. For example, PSS considers typing Ctrl Y in response to the DCL-TYPE command as unsolicited data.

Note: When playing back a file, containing multiple VMS login/logoff sequences, or when looping a file with a login/logoff sequence, you must enclose the carriage return character that initiates the login with %BEGIN/%END_UNSOLICITED lexicals. You must precede the carriage return by a %TIME lexical of at least three seconds to compensate for the time, taken by logoff processing, as follows:

```
%BEGIN_UNSOL()
%TIME(00:00:03) <CR>
%END_UNSOL()
```

EXAMPLES

1
```
%BEGIN_UNSOLICITED()
 ^Y
%END_UNSOLICITED()
```

To specify unsolicited data, simply bracket that data between these two lexicals. In this example, the Ctrl Y is unsolicited input.

2
```
%TIME(00:00:10)TYPE VERY_LARGE_FILE.LIS
%BEGIN_UNSOLICITED()
%TIME(00:00:30)  ^Y
%END_UNSOLICITED()
```

This example sequence initiates the following actions:

a. PSS waits ten seconds.

b. PSS then places *TYPE VERY_LARGE_FILE.LIS* <CR> into the input buffer.

c. PSS displays the listing.

d. Thirty seconds after executing the command, PSS forces an unsolicited Ctrl Y into the input buffer, ending the display.

%COMMENT—Enter a Comment

This lexical allows you to place comments within your PSS scripts.

FORMAT

%COMMENT *(text)*

PARAMETERS

text

A string which represents your comment

The string can contain any characters, including other PSS lexicals that PSS ignores.

DESCRIPTION

This lexical allows you to place comments within your PSS script.

EXAMPLES

1
```
%COMMENT(SHOW DATE & TIME)  <CR>
%TIME(00:00:05)SHOW DATE  <CR>
```

PSS ignores the comment, "SHOW DATE & TIME," and any other data within the parentheses during a PSS-playback execution.

%CUR_LOOP—Capture Current Value

This lexical allows you to capture the current value of the %DO-LOOP lexical.

FORMAT

%CUR_LOOP *(ident)*

PARAMETERS

ident
A numeric value, ranging between 1 and 32,767, that represents an active DO_LOOP

DESCRIPTION

This lexical causes the current value of the DO_LOOP, specified by the *ident* argument, to be substituted into the input-file script.

EXAMPLES

1

```
%CUR_LOOP(100)
%DO_LOOP(10000,10010,1,100)
%TIME(00:00:00:02)RUN UPDATE_PAYROLL <CR>
%CUR_LOOP(100) <CR>
%ENDDO_LOOP(100)
```

This is an example for using the current %DO_LOOP value, when updating employee records within the UPDATE_PAYROLL program. %CUR_LOOP(100) contains values ranging from 10,000 to 10,010.

%DO_LOOP—Loop Portion of Input Script

This lexical allows you to repeat specified lines within an input script.

FORMAT

%DO_LOOP *(start, end, step, ident)*

PARAMETERS

start

A numeric value, ranging between 0 and 32,767, that specifies the starting value of the %DO_LOOP lexical. This value must be consistent with the "end" and "step" values specified.

end

A numeric value, ranging between 0 and 32,767, that defines the ending value of the %DO_LOOP lexical. When the current value of the DO_LOOP reaches this value, the loop terminates. This value must be consistent with the "start" and "step" values specified.

step

A numeric value, ranging between -32,767 and 32,767, that specifies an increment that PSS adds to the current value of the DO_LOOP (beginning with the "start" value). This value must be consistent with the "start" and "end" values specified.

ident

A numeric value, ranging between 1 and 32,767, that specifies a unique identifier for the %DO_LOOP lexical. Other %DO_LOOP lexicals can use this value, but only if the current scope is not active.

DESCRIPTION

This lexical in conjunction with the %ENDDO_LOOP lexical allows you to repeat a portion of an input script 'n' number of times. Certain restrictions apply to the use of %DO_LOOP, as follows:

- Each %DO_LOOP must have a matching %ENDDO_LOOP.
- Within a %DO_LOOP each nest must have a unique-loop identifier.
- %DO_LOOP can be nested, as long as the nesting is consistent from inner to outer loops.

EXAMPLES

1

```
%DO_LOOP(1,10,1,100)
%TIME(00:00:02)RUN UPDATE_DATEBASE
%TIME(00:00:03)RUN GENERATE_REPORT
%ENDDO_LOOP(100)
```

This example shows how to use the %DO_LOOP lexicals to repeat a portion of a script ten times.

2

```
%DO_LOOP(1,10,1,100)
%TIME(00:00:01)RUN TEST1
%DO_LOOP(5,10,5,299)
%TIME(00:00:02)RUN TEST2
%ENDDO_LOOP(299)
%ENDDO_LOOP(100)
```

This example shows how to use the %DO_LOOP lexicals to create nested DO_LOOP's. Program TEST1 is executed ten (10) times. Program TEST2 is executed twenty (20) times.

%END_CHECKPOINT—End-User-Response Checkpoint

This lexical indicates the end of a user-response checkpoint.

FORMAT

%END_CHECKPOINT *(bucket)*

PARAMETERS

bucket
A number, currently from 1 to 4, into which PSS counts various levels of user-response time

DESCRIPTION

This lexical ends a user-response checkpoint.

A %BEGIN_CHECKPOINT() lexical *must* precede this lexical.

EXAMPLES

1
```
%BEGIN_CHECKPOINT(1)
%TIME (00:00:10) SHOW SYSTEM <CR>
%END_CHECKPOINT(1)
```

This example shows how to end a user-response checkpoint 1.

%ENDDO_LOOP—End of DO_LOOP

This lexical indicates the end of a specified and active DO_LOOP.

FORMAT

%ENDDO_LOOP *(ident)*

PARAMETERS

ident
A numeric value, ranging between 1 and 32,767, that identifies an active DO_LOOP.

DESCRIPTION

This lexical in conjunction with the %DO_LOOP lexical permits you to repeat a portion of a script. The "ident" value must correspond to that of a matching active DO_LOOP.

EXAMPLES

1
```
%DO_LOOP(1,10,1,100)
%TIME(00:00:02)RUN UPDATE_DATABASE
%TIME(00:00:03)RUN GENERATE_REPORT
%ENDDO_LOOP(100)
```

This example shows how to repeat a portion of script ten times, using the %DO_LOOP lexical.

%END_UNSOLICITED—End Unsolicited Data

This lexical indicates the end of unsolicited data.

FORMAT

%END_UNSOLICITED *()*

DESCRIPTION

This lexical ends an unsolicited-input sequence. A %BEGIN_UNSOLICITED() lexical must precede this lexical.

EXAMPLES

1
```
%BEGIN_UNSOLICITED()
%TIME (00:00:05)  ^Y
%END_UNSOLICITED()
```

This example shows how to end an unsolicited input sequence.

%EXEC—Execute another Input Script

This lexical allows you to invoke an input script from another input script and optionally pass as many as 64 parameters to it.

FORMAT

%EXEC *(file-specification, [param-1,param-2...param-64])*

PARAMETERS

file-specification
An input-file specification that you want PSS to invoke

Note: You must include the complete VMS file specification—the device, directory, filename, and extension. There are no defaults.

DESCRIPTION

The %EXEC lexical allows an input script to invoke one or more subsidiary input scripts and optionally pass as many as 64 parameters to it. You cannot use PSS lexicals as parameter arguments to a %EXEC statement. When PSS encounters a %EXEC lexical in an input file, PSS opens the new input file, using the file-specification, specified as the first argument, and begins to play it back. When PSS reaches the end of the %EXEC input file, execution resumes at the point in the original input file that the %EXEC lexical was first encountered. The subsidiary file can also contain %EXEC lexicals, nested to any practical depth.

Note: While PSS does not place an absolute restriction on the number of %EXEC lexicals that you can specify, each one results in open- and close-file processing. Additional PSSRV quotas may need to be increased to accomodate %EXEC file processing.

EXAMPLES

1
```
%EXEC(SYS$SYSDEVICE:[MYDIR]SCRIPT.INP)
```

This example executes the PSS script SYS$SYSDEVICE:[MYDIR]SCRIPT.INP.

2
```
[  File MAIN_TEST.INP]
%EXEC(SYS$SYSDEVICE:[MYDIR]LOGON.INP)
%EXEC(SYS$SYSDEVICE:[MYDIR]TEST.INP)
%EXEC(SYS$SYSDEVICE:[MYDIR]LOGOFF.INP)
```

This example illustrates the most common form of the %EXEC lexical. The "LOGON" script executes the actual VMS login in the input script, "MAIN_TEST." (That script must use %PARAM lexicals for the username and password). The script "TEST" performs the actual test. When it finishes, the "LOGOFF" script logs you off the system. Properly used, the %EXEC lexical provides a highly modular and flexible test system.

3
```
%EXEC(SYS$SYSDEVICE:[MYDIR]SCRIPT1.INP,ABC,DEF,GHI)
```

This example demonstrates the method of passing parameters to an input-script file. In the example three parameters are passed—the strings "ABC," "DEF," and "GHI."

%EXPER—Experiment Time

This lexical appears in the input file. It contains the date and time that PSS recorded this input script.

FORMAT

%EXPER *(date-time-specification)*

PARAMETERS

date-time-specification
An indication of the date and time that PSS created the input-script file in dd-mmm-yyyy hh:mm:ss.cc format

DESCRIPTION

PSS automatically inserts this lexical into each input file that a RECORD command produces. It does not appear in input files that were recorded prior to Version 2.0 of PSS or in those, not produced by a PSS recording session.

EXAMPLES

```
1    %EXPER(28-FEB-1986 15:22:27.76)
```

This lexical indicates the start of a recorded experiment. Use this lexical primarily for documentation purposes and only if you want to convert the input file, using the /ABSOLUTE qualifier. If so, it must be the *first* in the file.

%GOTO—Unconditionally alter script control flow

This lexical allows you to unconditionally alter the control flow of an input script.

FORMAT %GOTO (label)

PARAMETERS *label*

Contains the name of the label to which control is to be transferred unconditionally.

%INDEX—Substitute Unique Numeric String

This lexical substitutes a unique numeric value within the input script for a playback session at run-time.

FORMAT

%INDEX *(length)*

PARAMETERS

length
A numeric value, representing the length of the numeric string that the %INDEX lexical places within the input script

DESCRIPTION

This lexical allows you to use a single input script for a multiple terminal playback session. The %INDEX(n) lexical induces PSS to insert "n" unique numeric digits into the input stream in place of the lexical. It begins the numeric value at 1 for the first playback session and increments it for each subsequent playback (if you require predictable substitution, see the %PARAM lexical). The %INDEX lexical specifies the length as zero-filled and substitutes the numeric value for the existing lexical within the input script.

EXAMPLES

1
```
%TIME(00:00:05]ADD <CR>
%TIME[00:00:01]TRANSACT%INDEX(4) <CR>
```

This example shows how to use the %INDEX lexical to add a four-digit-numeric string to the end of the data, "TRANSACT," yielding TRANSACT0001. The actual input data passed is, as follows:

```
ADD <CR>
TRNASACT0001 <CR>
```

%LABEL—Define a label

This lexical allows you to define a label in a script that can be used as the target of a %MATCH or %GOTO label.

FORMAT %LABEL (name)

PARAMETERS

name
Contains the name of the label.

%LOCAL_PARAM—Perform Local Parameter Substitution

This lexical allows you to retrieve parameters passed to an input script that was executed by the %EXEC lexical.

FORMAT

%LOCAL_PARAM *(position)*

PARAMETERS

position
A numeric value that denotes the position of the parameter, passed within the executed %EXEC lexical

DESCRIPTION

This lexical allows you to retrieve the parameter, specified within an executed %EXEC lexical. The parameter itself is then inserted into the input data stream of the playback session. Parameters passed, using the %EXEC lexical, are local to the input file, specified within the %EXEC "file-specification" argument. Note that %LOCAL_PARAM and %PARAM are not interchangeable.

EXAMPLES

1

```
[file MAIN_TEST.INP]
%EXEC(SYS$SYSDEVICE:[MYDIR]TEST,DEPT,HOURLY)

[file TEST.INP]
%TIME(00:00:03)RUN PAYROLL_REPORT
%LOCAL_PARAM(1) <CR>
%LOCAL_PARAM(2) <CR>
```

This example illustrates how to code the %LOCAL_PARAM lexicals within the "TEST" script. The arguments are passed via the %EXEC lexical which is in the script, "MAIN_TEST".

%MAIL—Send Data to User-Written Program

This lexical allows you to send data to a monitor program which controls or synchronizes PSS playback sessions.

FORMAT

%MAIL *(mailbox-name,"text")*

PARAMETERS

mailbox-name

A logical name for a system-wide mailbox into which PSS sends the textual data, represented by the "text" parameter

text

This argument represents data which is to be sent to the monitor application's mailbox. You must enclose this argument with double quotes, and it must *not* contain an embedded double quote.

DESCRIPTION

The %MAIL lexical provides a means for controlling and synchronizing a playback session, using a user-written program, which you as a PSS user write. The user-written program creates a system-wide mailbox, assigning the same logical name you specified in the "mailbox-name" parameter of the %MAIL lexical. When the input script reaches the %MAIL lexical, PSS sends the data, associated with the lexical, to the mailbox. The user-written program then reads this data through traditional VMS System Services. The user-written program, spawning PSS commands, can then initiate other playback sessions to test aspects of your application system that require rigid synchronization.

The format of the mailbox message, sent by PSS to the named mailbox, consists of a 16-byte-counted string, representing the name of the actual terminal under playback. The first byte indicates the actual number of significant characters.

The actual data message is specified within the %MAIL lexical.

EXAMPLES

1

```
%MAIL(MONITOR_MBX,"Test-Start")
%TIME(00:00:01)SHOW SYSTEM  <CR>
%TIME(00:00:01)SHOW MEMORY  <CR>
%MAIL(MONITOR_MBX,"Test-End")
```

This example illustrates use of the %MAIL lexical. Initially, you write a mailbox message, containing the data, "Test-Start," to the mailbox logical MONITOR_MBX. After executing the two VMS commands, the second (and final) %MAIL lexical writes the data, "Test-End," to the MONITOR_MBX logical.

%MATCH—Match Output String

This lexical allows you to match a string against the output produced by a playback input item.

FORMAT

%MATCH ("string",label)

restrictions

The %MATCH lexical must be paired with the input item that immediately procedes it in the script.

PARAMETERS

string

Contains the quoted, ASCII string that you want to match against the playback session output.

label

Contains the name of the label to which control is to be transferred on a successful match.

DESCRIPTION

Use this lexical to conditionally alter the control flow of an input script, based on the contents of the playback session's output.

%OUTPUT—Output Text String to Terminal

This lexical allows you to output text during a playback session.

FORMAT

%OUTPUT *(text)*

PARAMETERS

text

A string representing the data which is to be sent as output during the playback session

The data can contain any character values.

DESCRIPTION

This lexical allows you to output data to a terminal during playback. The data may contain any binary or ASCII-data values. This lexical proves particularly useful with CAD/CAM and/or graphic-terminal systems that often require local-terminal interactions to set various states. Rather than modify your application program to issue various escape sequences to set the terminal's state, you can use a %OUTPUT lexical to issue these sequences under PSS control.

EXAMPLES

1

```
%OUTPUT(<ESC>#6Hello)
```

This example shows how to display the word "Hello" in double-width mode on an ANSI terminal.

%PARAM—Perform Global Parameter Substitution

This lexical allows you to substitute a parameter, entered via the PSS PLAYBACK command, into the input script at run time.

FORMAT

%PARAM *(position)*

PARAMETERS

position

A numeric integer value, which represents the position of the parameter, specified with the /PARAMETER qualifier. The "position" argument must be a number between one (1) and ten (10).

DESCRIPTION

This lexical allows you to substitute a parameter, entered at run time as a part of issuing the PSS/PLAYBACK command. Use the PSS/PLAYBACK command qualifier, /PARAMETER, to enter the parameters. You can only substitute data, not other PSS lexical commands. Practical uses for this feature include usernames, passwords, and prompt responses. Note that unlike %LOCAL_PARAM, %PARAM can be coded anywhere within the input script, regardless of whether it was %EXEC'd or run as a primary input-script file.

EXAMPLES

1

```
$ PSS/PLAY/INPUT=TEST/PARAMETER=(USER,PASS) TTA1:
[ file TEST.INP]
%TIME(00:00:00) <CR>
%TIME(00:00:03)%PARAM(1) <CR>
%TIME(00:00:02)%PARAM(2) <CR>
```

This example shows how to enter a username and password without "hard coding" them into the input script. The script is now available for general use and does not compromise system security. You can specify parameters anywhere actual data would be entered.

%RANDOM—Generate a Random Number

This lexical allows you to generate a random number for insertion into the input-data stream

FORMAT

%RANDOM *(lower, upper, format, fill)*

PARAMETERS

lower

A numeric floating value that represents the lower boundary of a range within which PSS selects a random number

upper

A numeric floating value that represents the upper boundary of a range within which PSS selects a random number

format

The total size of the random number which is to be inserted into the input-data stream

This argument is specified in the format of "a.b" with "a" representing the total length of number, "." representing the decimal point, and "b" representing the number of digits to the right of the decimal point.

fill

This argument indicates whether the resultant-random number should be padded with leading zeroes. A zero (0) indicates that no leading zeroes be inserted (the default), a one (1) indicates that leading zeroes should be inserted if the random number does not completely fill the desired format.

DESCRIPTION

The %RANDOM lexical allows you to request that a random number be generated by PSS and inserted into the input stream. You can specify the lower and upper limits of the random number, selected, as well as the "display" format of the number. %RANDOM works in conjunction with the /SEED playback qualifier. Given a specific /SEED value, %RANDOM will select reproducible random numbers.

EXAMPLES

1

```
%TIME(00:00:04:15)RUN GET_INPUT <CR>
%RANDOM(2.1,3.7,2.10 <CR>
```

This example illustrates how the RANDOM lexical might be used. A random number in the range of 2.4 to 3.7 will be selected by PSS. The format of 2.1 means that the random number selected will be restricted to two (2) digits, in total (including the digits to the right of the decimal point), and one (1) digit to the right of the decimal point.

%RANDOM_TIME—Wait for Random Timer

This lexical, similiar to %TIME, generates a random waiting time.

FORMAT

%RANDOM_TIME *(lower, upper)*

PARAMETERS

lower
The lower boundary of the random time, as specified in VMS-delta-time format

upper
The upper boundary of the random time, as specified in VMS-delta-time format.

DESCRIPTION

This lexical allows you to generate a random waiting time. Note that all other processing is similar to the %TIME lexical. The purpose of this lexical is to provide a more realistic playback session by choosing random-wait times for various user's actions within an input script. This lexical is particularly useful, when executing the same input script for a large number of potential users.

EXAMPLES

1
```
%RANDOM_TIME(00:01:00.00, 00:05:00.00)
```

This example produces a random waiting time between one and five minutes.

%SUSPEND—Print Message and Suspend Playback

This lexical allows you to temporarily suspend a playback session.

FORMAT

%SUSPEND *(text)*

PARAMETERS

PARAMETERS
A string, representing a message to be displayed, prior to playback suspension

DESCRIPTION

The %SUSPEND lexical permits you to suspend a playback session. The session continues, when the user, at the playback terminal, presses the Return key. This lexical is useful when you use PSS for interactive "slide-show" type applications or when you must read a page of data and then RETURN to continue to the next slide.

EXAMPLES

1
```
%SUSPEND("Proceed, when ready")
```

This example causes the playback session to pause with the message, "Proceed, when ready," followed by "Enter Return to Continue," displayed on the playback terminal. When the terminal user presses the Return key, PSS continues the playback session.

%TIME—Wait for Timer

This lexical specifies the waiting time before passing data characters as input during a playback.

FORMAT

%TIME *(time-specification)*

PARAMETERS

time-specification

The time to wait (think time) before PSS resumes passing input characters to the playback terminal. The "time-specification" is entered as the time portion of a VMS-delta-time.

DESCRIPTION

The time found in the %TIME lexical consists of a relative (or delta) time, expressed in hours, minutes, seconds, and milliseconds. (Milliseconds are optional.) When recording a terminal session, PSS automatically inserts a %TIME lexical in front of each input-data record. The format of the input-data record is, therefore, %TIME, data, delimiter (if any).

Should the application *not* solicit input data after the time period expires, (and a %BEGIN_UNSOLICITED lexical is not in effect) PSS will still wait for the data to eventually be solicited by the program.

EXAMPLES

1
```
%TIME(00:00:05)SHOW SYSTEM  <CR>
```

This example shows how to make PSS wait five (5) seconds before entering "SHOW SYSTEM <CR>."

2
```
%TIME(00:01:10)MON PROC  <CR>
```

This example shows how to make PSS wait one (1) minute and ten (10) seconds before entering "MON PROC<CR>."

3
```
%TIME(00:00:00)EDT  <CR>
```

This example shows how to eliminate a waiting time, but provide delimiter-response checking.

7.4 PSS Response-Event Conversion

PSS offers you the opportunity to capture the actual response times encountered for each input event, in a disk file. This file, named the PSS Response Event file, contains the following items for each response event:

- The actual date/time the response was encountered
- The actual response time seen
- A mask, containing the active "buckets," enabled during the response collection

This file can be converted from PSS internal format to an RMS sequential file with the PSS/CONVERT command. The records in this file are all fixed length (20 characters) and have the following format:

Response Checkpoint	4 Bytes	See below
Experiment Date/Time	8 Bytes	VMS Quadword-Absolute
Response Time	8 Bytes	VMS Quadword-Delta

This file can be analyzed to provide further insight into your playback experiment's response time. For example, you could use Digital's VAX Datatrieve product to plot the results. This creates an interesting and useful graph. Just plot experiment time by response time.

The **Response Checkpoint** field is a bit-oriented field in which each bit corresponds to a number, specified as part of a %BEGIN_CHECKPOINT lexical within the input script. Bit 0, if set, indicates that Checkpoint zero (0) was active, bit 1 indicates the status of Checkpoint 1, etc. This feature allows you to search for response events that particularly interest you.

7.5 PSS Response Event Reporting Facility

PSS reports on the contents of the Response Event file. PSS/RESPONSE is the command that requests this feature. PSS/RESPONSE can report on selected response buckets within a selected-experiment time and on selected-response times encountered. An interesting feature of PSS/RESPONSE is the ability to match the input script, used with the actual response events. This allows you to determine exactly what input data caused a specific response event.

7.5.1 RESPONSE Command

RESPONSE

This command provides the response-event reporting aspects of PSS.

FORMAT

PSS/RESPONSE/qualifiers Response-file

Command Qualifiers	Defaults
/AFTER=VMS-Absolute-Time	*Beginning of Experiment*
/BEFORE=VMS-Absolute-Time	*End of Experiment*
/BUCKETS=bucket-list	*All Active Buckets*
/GT=VMS-delta-time	*See Text*
/GTE=VMS-delta-time	*See Text*
/INPUT=file-specification	*See Text*
/LOOP=number	*/LOOP=1*
/LINE_EDIT	*/LINE_EDIT*
/LT=VMS-delta-time	*See Text*
/LTE=VMS-delta-time	*See Text*
/OUTPUT[=file-specification]	*See Text*

prompts Response File: Response-file

PARAMETERS

Response-file

Specifies the converted Response Event file captured as a result of a PSS playback session.

DESCRIPTION

This command will produce a paginated 132-column report that analyzes the information contained within the Response Event file. You can print response data on all items within the file, or through the use of command qualifiers select only those items of interest. For each item selected the following information is reported:

- The number of the item as it exists in the response file
- The Experiment Time in VMS-date and -time format at which this item was played back
- The response-checkpoint buckets that were in effect, when this item was played back
- The response time, in VMS-delta-time format, taken by this item

In addition if a PSS input script file is provided, the input data that caused the response item to be recorded will be displayed. At the end of the report a response summary is generated, showing the experiment time ranges, covered by this report, the number of items on the response file,

the number of items selected for the report, the average response time of the selected items, and the median (or mid-point) response time.

COMMAND QUALIFIERS

/AFTER=absolute-time

This qualifier indicates the time after which items should be selected for this report. The time is expressed as a VMS-absolute time.

/BEFORE=absolute-time

This qualifier indicates the time before which items should be selected for this report. The time is expressed as a VMS-absolute time.

/BUCKETS=bucket-list

This qualifier allows you to choose which active response-checkpoint buckets should be selected for this report. Buckets are specified as a single digit in the range of 0 to 4, corresponding to the PSS response-checkpoint buckets. If more than one bucket is to be specified, they must be contained within parentheses and be separated by commas. For example, /BUCKETS=(1,4), indicates that buckets 1 and 4 are to be selected. By default all buckets are eligible for selection.

/GT=delta-time

This qualifier allows you to indicate that only items with actual response times, greater than that specified with this qualifier, are to be selected. For example, /GT=::4:00 indicates that a response time, greater than 4 seconds, is to be selected. The time is specified as a valid VMS-delta time. /GT and /GTE are mutually exclusive.

/GTE=delta time

This qualifier allows you to indicate that only items with actual response times, greater than or equal to that specified with this qualifier, are to be selected. For example, /GTE=::4:00 indicates that a response time, greater than or equal to 4 seconds, is to be selected. The time is specified as a valid VMS-delta time. /GT and /GTE are mutually exclusive.

/INPUT=file-specification

This qualifier specifies the PSS input script file for reporting purposes. If the input file contains %EXEC lexicals, response reporting follows the same rules as PSS/PLAYBACK. The file specification (or logical name) in the %EXEC lexical must be a fully expanded file specification of the new input file or a SYSTEM-wide logical name that has been equated to the fully expanded, file specification. No default-directory processing is performed. If this qualifier is omitted, the response information is produced, but no input data appears on the report.

/LOOP=number
/LOOP=1 (default)

This qualifier is required, if your playback was executed using the PSS/PLAYBACK/LOOP qualifier. The value specified should be the same as that specified during the playback.

/LINE_EDIT
/NOLINE_EDIT (default)

This qualifier allows the reporting program to substitute the occurance of an UP ARROW and/or DOWN ARROW key with the actual data recalled. Use this qualifier, if the input script contains UP/DOWN ARROW sequences that were meant to be interpreted as a command recall. If your application will interpret those keys differently, then you should specify /NOLINE_EDIT, to obtain the escape sequences entered.

/LT=VMS-delta-time

This qualifier allows you to indicate that only actual response times, less than or equal to that specified with this qualifier, are to be selected. For example, /LT=::5:00 indicates that a response time, less than 5 seconds, is to be selected. The time is specified as a valid VMS-delta-time. /LT and/LTE are mutually exclusive.

/LTE=VMS-delta-time

This qualifier allows you to indicate that only actual response times, less than or equal to that specified with this qualifier, are to be selected. For example, /LT=::5:00 indicates that a response time, less than or equal to 5 seconds, is to be selected. The time is specified as a valid VMS-delta-time. /LT and /LTE are mutually exclusive.

/OUTPUT[=file-specification]

This qualifier controls where the output of the PSS/RESPONSE command is sent. If you do not enter the /OUTPUT qualifier and/or a file-specification is not specified, the output is sent to the logical SYS$OUTPUT (usually your terminal). If you enter a file-specification that is not fully qualified, PSS applies the following default, PSSRSP.RPT. Wildcards cannot be specified.

EXAMPLES

1 $ (PSS/RESPONSE/INPUT=PLTA5 PLTA5)

```
$ PSS/RESPONSE/INPUT=PLTA5 PLTA5

                              PERFORMANCE SIMULATION SYSTEM
                    Copyright (C) 1983, 1989, Advanced Systems Concepts, Inc.
                                        Response Report

------------------------------------------------+----------------------------
Item     Exper Time      Buckets  Response Time| Input Data
------------------------------------------------+----------------------------
                                                |
 1  17-SEP-1988 18:06:35    0     0 00:00:01.70| %TIME(00:00:00.00) <CR>
 2  17-SEP-1988 18:06:51    0     0 00:00:00.40| %TIME(00:00:01.86)TEST_USER <CR>
 3  17-SEP-1988 18:06:59    0     0 00:00:00.10| %TIME(00:00:01.60)SAMPLE <CR>
 4  17-SEP-1988 18:07:05    0     0 00:00:00.11| %TIME(00:00:16.34)SHO SYS <CR>
 5  17-SEP-1988 18:07:35    0     0 00:00:07.97| %TIME(00:00:06.94)SHO MEM <CR>
 6  17-SEP-1988 18:07:36    0     0 00:00:00.12| %TIME(00:00:00.00)LOGOUT <CR>

                                                        +------------------+
                                                        ! Response Summary !
                                                        +------------------+

Experiment Time Range: 17-SEP-1988 18:06:35.98 - 17-SEP-1988 18:07:35.45
Items on File:                          6
Items Selected:                         6
Average Response Time:    0 00:00:02.06
Median Response Time:     0 00:00:00.40

Command Line: PSS/RESP/INP=PLTA5/OUT=PLTA5.RPT PLTA5.DAT
```

This example shows a complete response-event report. Notice that all the input data is matched with the actual response events. At the end of the report, the "Average" and "Median" response times are displayed.

Note: The example report above was compressed to physically fit on the page. As a result, some of the captions have been shortened and the Experiment Time, normally printed to the hundredth of a second, has been truncated.

8 INTERACTIVE REAL-TIME MONITORING

8.1 Overview

PSS MONITOR is an interactive real-time, window-oriented utility that performs three major tasks:

- Displays a full range of active sessions on any DEC-supported terminal as follows:

  A brief or full report for a particular terminal

  A brief or full report on all terminals in PLAYBACK mode

  A brief or full report on all terminals in RECORD mode

  A brief or full report on all terminals, regardless of PSS mode

  PSS/MONITOR allows you to control the size, location, and the order of the windows. In addition you can perform ADD, MOVE, ADVANCE, BACKUP, REMOVE, PASTE, PRINT, TOP, SET, REFRESH and UNPASTE operations so you can examine all or part of the current PSS activity.

- Allows you to issue most PSS commands: RECORD, PLAYBACK, STOP, CONVERT, and AUTHORIZE from the MONITOR prompt

- Permits you to issue SPAWN and HELP commands from the MONITOR prompt

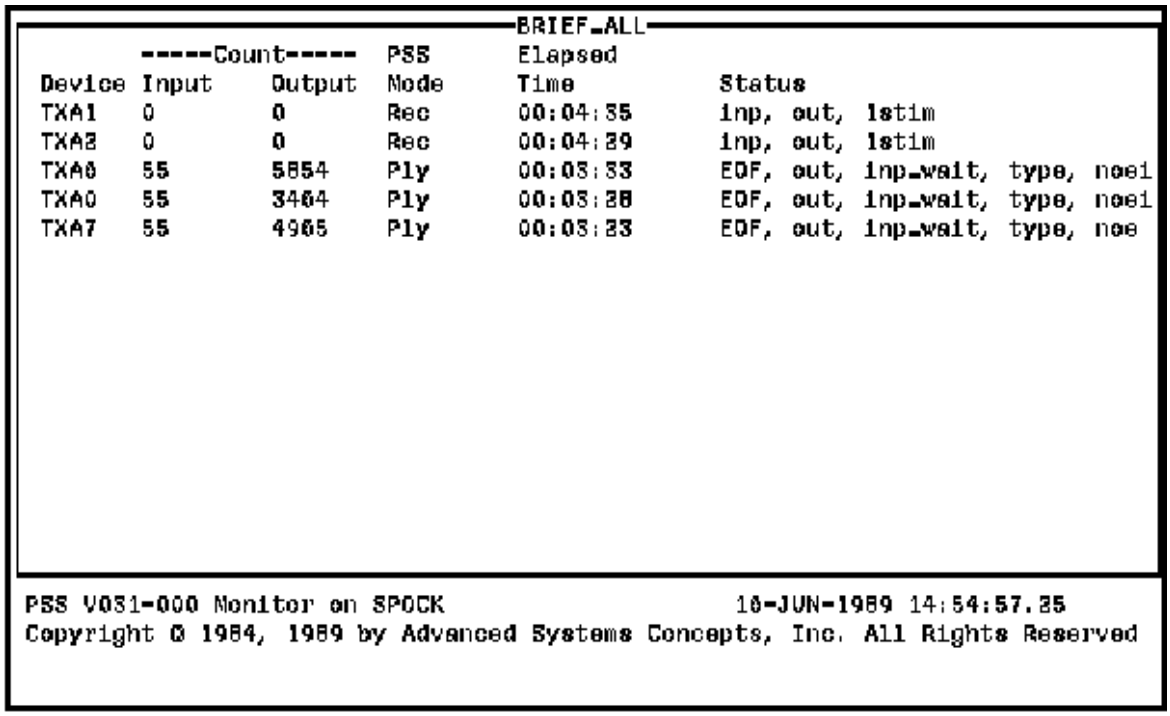
8.2 Starting the PSS MONITOR Utility

To start the PSS MONITOR Utility, enter the following DCL command:

```
$ PSS/MONITOR window_name
```

If "window_name" is not specified, PSS/MONITOR begins by displaying the BRIEF_ALL window. This window presents you with a brief display of all active PSS sessions (if any). "Window_name" must be one of the valid choices as presented in Table 8–2. For a complete description of the PSS/MONITOR command, please read Section 5.2 under "MONITOR".

Figure 8–1 MONITOR default window



8.3 Command Syntax

While PSS MONITOR is running, your selected window(s) are updated at the default-time interval of three (3) seconds. To enter MONITOR commands, you simply start typing. PSS MONITOR will move the cursor to the bottom of the screen and begin echoing your commands. Section 5.2.2 lists the commands which can be entered at the MONITOR> prompt. Use the following format to specify these commands:

```
MONITOR> command [window_name [{ AT or TO } position]]
```

Valid *commands* are specified in Table 8–1. Valid *window names* are specified in Table 8–2. Valid *position* commands are specified in Table 8–3. You can specify one or more position commands (i.e., R10 C5).

Figure 8–2 MONITOR command prompt

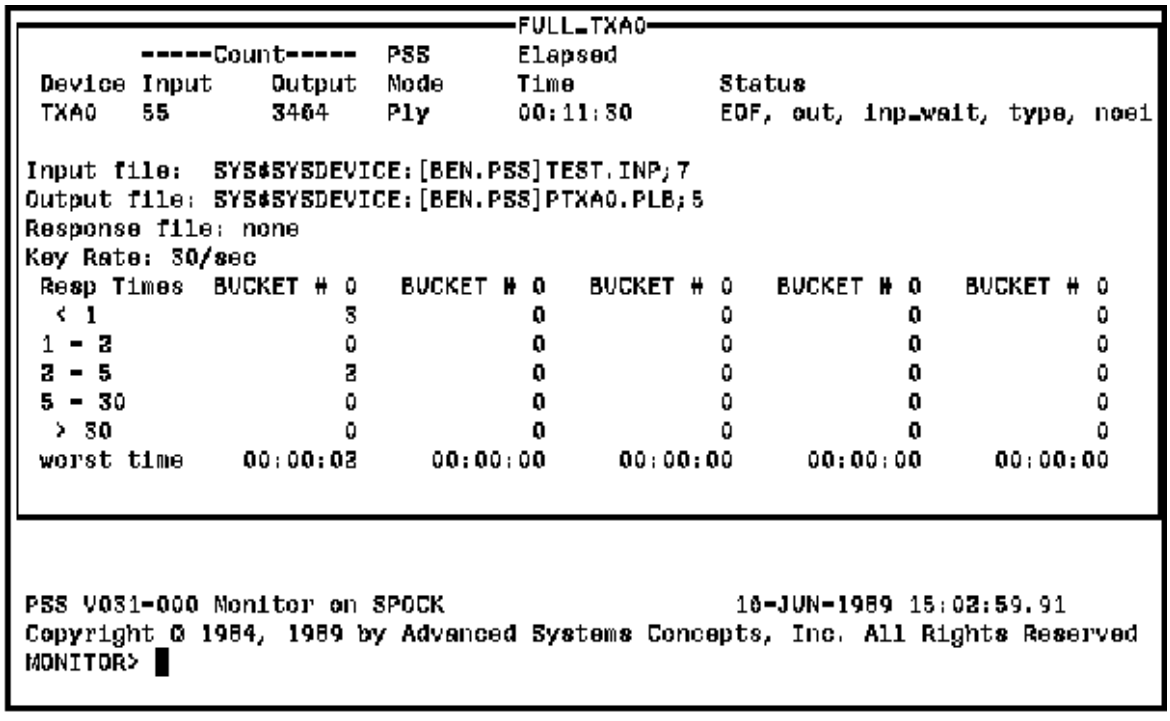


Table 8–1 MONITOR commands

Command	Key	Description
ADD	None	Adds a window
ADVANCE	Next Screen	Scrolls the information within a window forward
BACKUP	Prev Screen	Scrolls the information within a window backward
EXIT	Ctrl-Z	Exits the MONITOR Utility
MOVE	Insert Here	Moves a window to a new position on the screen
PASTE	None	Pastes a window to the front
PRINT	None	Prints Screen via VMS Print Services
REFRESH	Ctrl-W	Refreshes the screen
REMOVE	Remove	Removes a window
SET	None	Sets a MONITOR parameter
TOP	Find	Resets window display to logical TOP
UNPASTE	None	Makes a window invisible

Table 8–2 Window Names

Name	Description
BRIEF_ALL	Brief list of all PSS sessions
BRIEF_PLAYBACK	Brief list of all PSS playback sessions
BRIEF_terminal	Brief report of a selected PSS session
BRIEF_RECORD	Brief list of all PSS record sessions
FULL_ALL	Full report of all PSS sessions
FULL_PLAYBACK	Full report of all PSS playback sessions
FULL_terminal	Full report of a selected PSS session
FULL_RECORD	FULL report of all PSS record sessions

Table 8–3 Position Parameter

Position	Description
Hn	Halves (H1, top half; H2, bottom half)
Tn	Thirds (T1, T2, T3)
Qn	Quarters (Q1, Q2, Q3, Q4)
Rn	Row number
Cn	Column number
Sn	Window size (in rows)

8.4 Stopping the PSS MONITOR Utility

To stop the PSS MONITOR Utility, type EXIT or CTRL Z. For example:

```
MONITOR> EXIT
```

8.5 MONITOR Commands

8.5.1 ADD

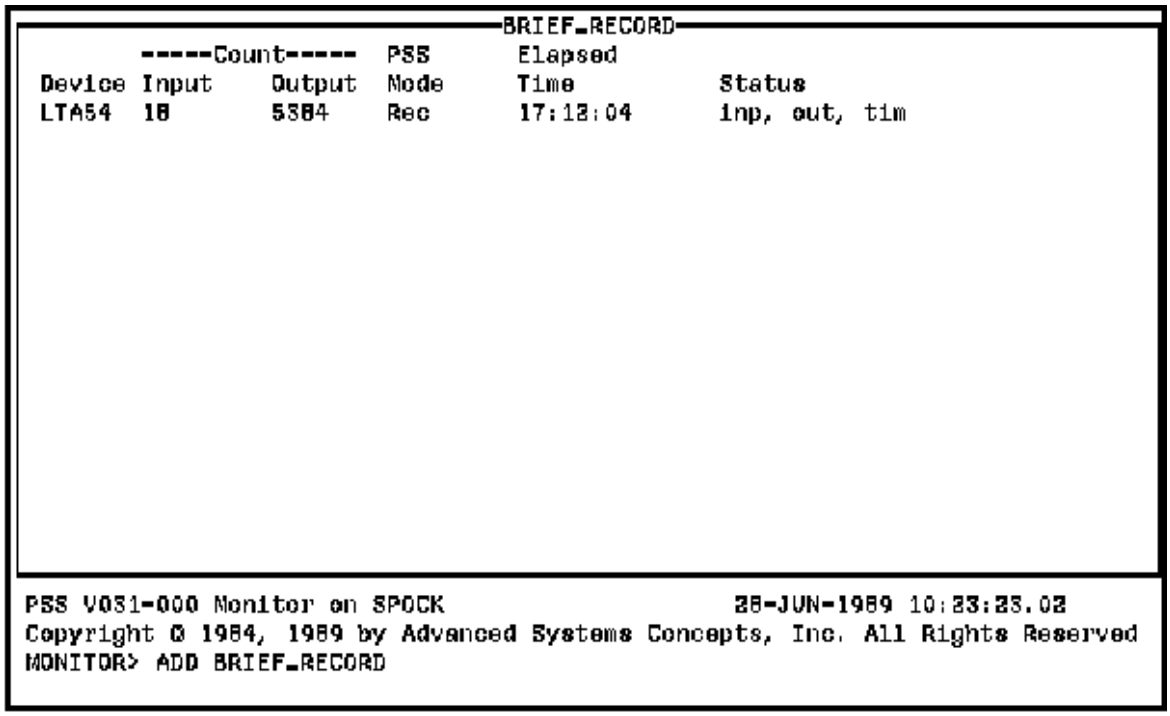
This command allows you to add a specified window to the display.

For example, if you want to view all RECORD sessions in brief mode, enter the following MONITOR command:

```
MONITOR> ADD BRIEF_RECORD
```

Figure 8–3 illustrates this command.

Figure 8–3 ADD BRIEF_RECORD Window

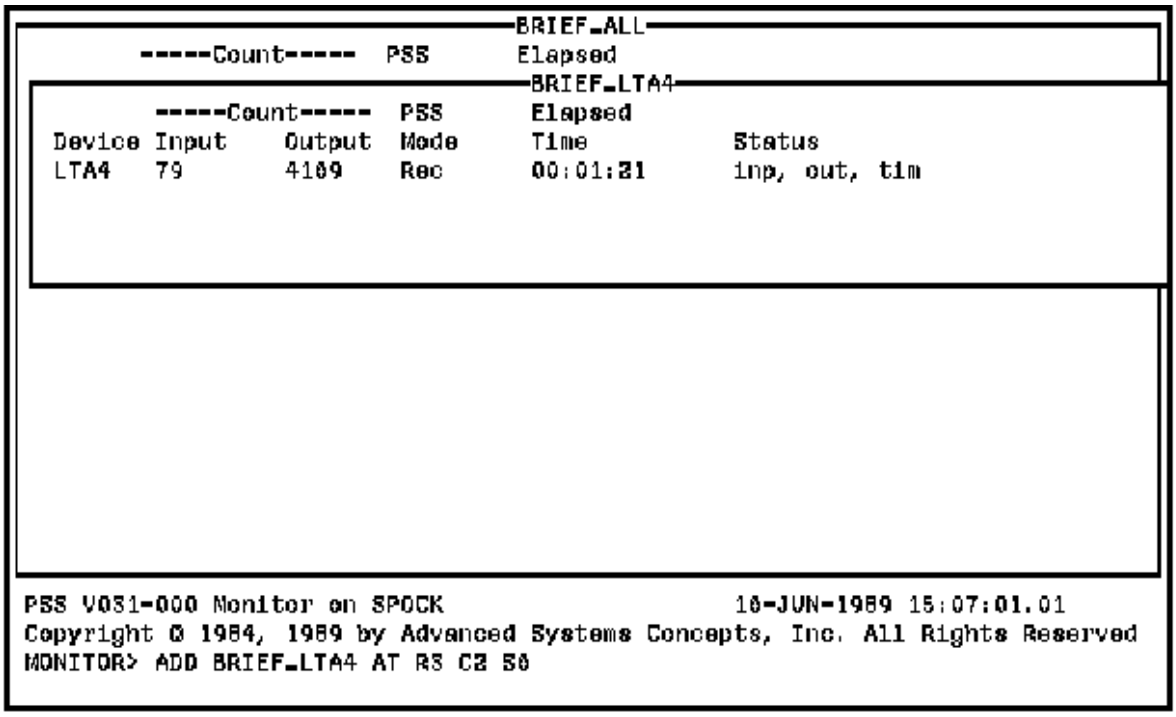


You can also control the location of the window on the screen by specifying the starting row number, starting column number, and the size of the window by entering the following command:

```
MONITOR> ADD BRIEF_LTA4 AT R3 C2 S6
```

This is depicted in Figure 8–4.

Figure 8–4 ADD BRIEF_LTA4 with Position Commands



8.5.2 ADVANCE

This command scrolls forward within the current window and displays the additional active PSS session or sessions. The ADVANCE command is valid for any nonterminal-specific window. Specifically, BRIEF_terminal or FULL_terminal windows cannot be advanced, since those windows only deal with one (1) terminal.

For example, if the current window displays a FULL_ALL session, and you want to view the remaining sessions, not currently displayed, issue the following command:

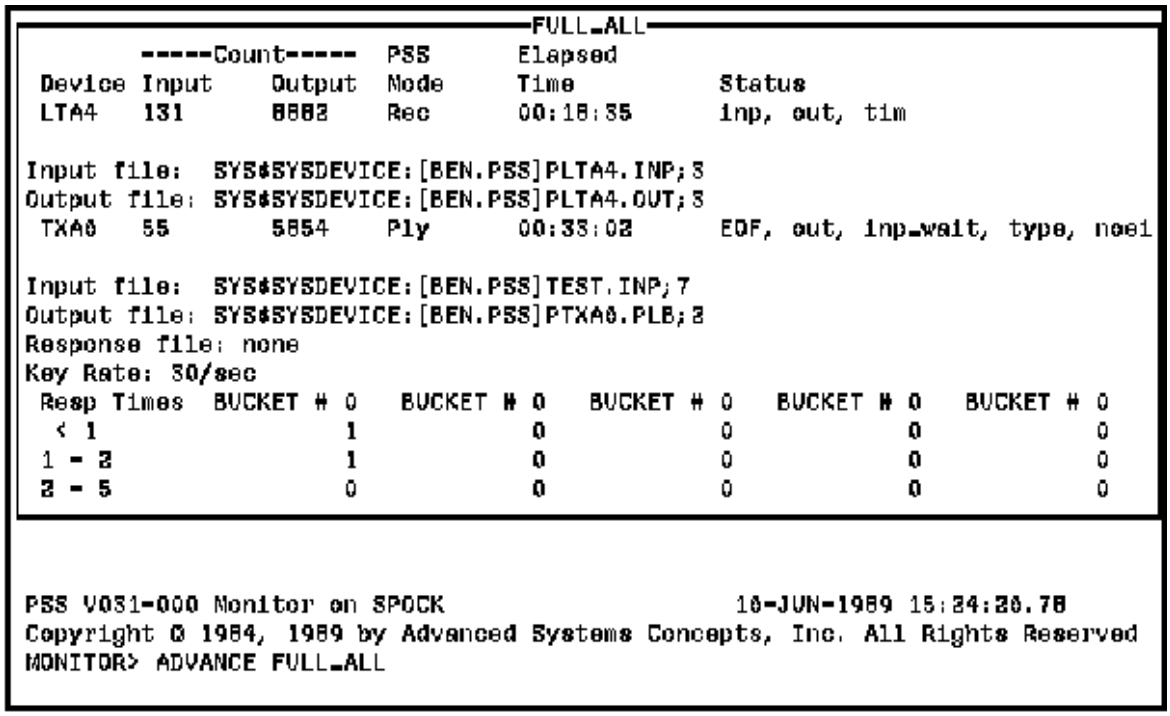
```
MONITOR> ADVANCE FULL_ALL
```

Figure 8–5 depicts the screen while the user is entering the command, and Figure 8–6 shows what the screen looks like after the command has been executed.

Figure 8–5 ADVANCE FULL_ALL Before Execution

```
                                    FULL_ALL
        -----Count-----  PSS      Elapsed
 Device Input    Output  Mode     Time          Status
 TXA1   0        0       Rec      00:24:09      inp, out, lstim

Input file:  SYS$SYSDEVICE:[BEN.PSS]PTXA1.INP;4
Output file: SYS$SYSDEVICE:[BEN.PSS]PTXA1.OUT;4
 TXA2   0        0       Rec      00:24:04      inp, out, lstim

Input file:  SYS$SYSDEVICE:[BEN.PSS]PTXA2.INP;1
Output file: SYS$SYSDEVICE:[BEN.PSS]PTXA2.OUT;1
 LTA4   109      6204    Rec      00:08:40      inp, out, tim

Input file:  SYS$SYSDEVICE:[BEN.PSS]PLTA4.INP;3
Output file: SYS$SYSDEVICE:[BEN.PSS]PLTA4.OUT;3
 TXA0   55       5854    Ply      00:23:07      EOF, out, inp_wait, type, neel


PSS V031-000 Monitor on SPOCK                    10-JUN-1989 15:14:31.74
Copyright © 1984, 1989 by Advanced Systems Concepts, Inc. All Rights Reserved
```

Figure 8–6 ADVANCE FULL_ALL After Execution



8.5.3 BACKUP

This command scrolls backward within the current window and redisplays the previous PSS session or sessions. The BACKUP command is valid for any nonterminal-specific window. Specifically, BRIEF_terminal or FULL_ terminal windows cannot be advanced, since those windows only deal with one (1) terminal.

To backward scroll, issue the following command:

```
MONITOR> BACKUP BRIEF_ALL
```

8.5.4 MOVE

This command allows you to move a window to a new location on the terminal screen. For example, if you want to move the BRIEF_All window from its default location of row one, column one, to row ten, column two, issue the following command:

```
MONITOR> MOVE BRIEF_ALL TO R10 C2
```

Figure 8–7 depicts the screen while the user is entering the command, and Figure 8–8 shows what the screen looks like after the command has been executed.

Figure 8–7 MOVE BRIEF_ALL Before Execution

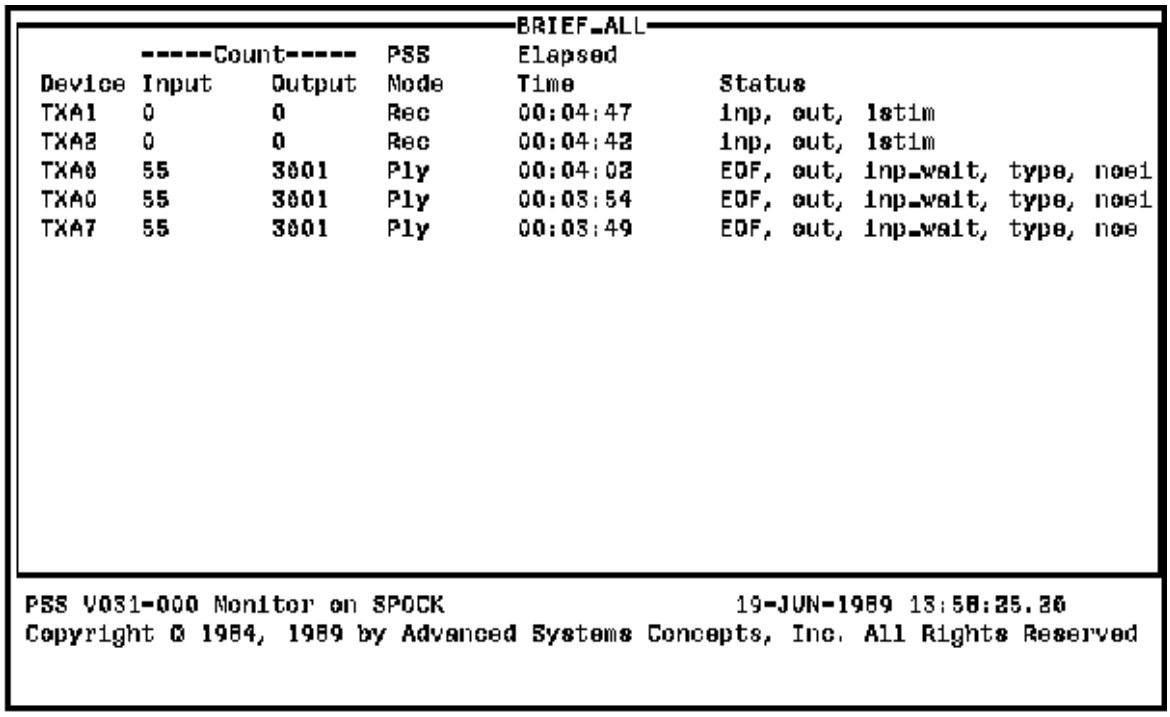
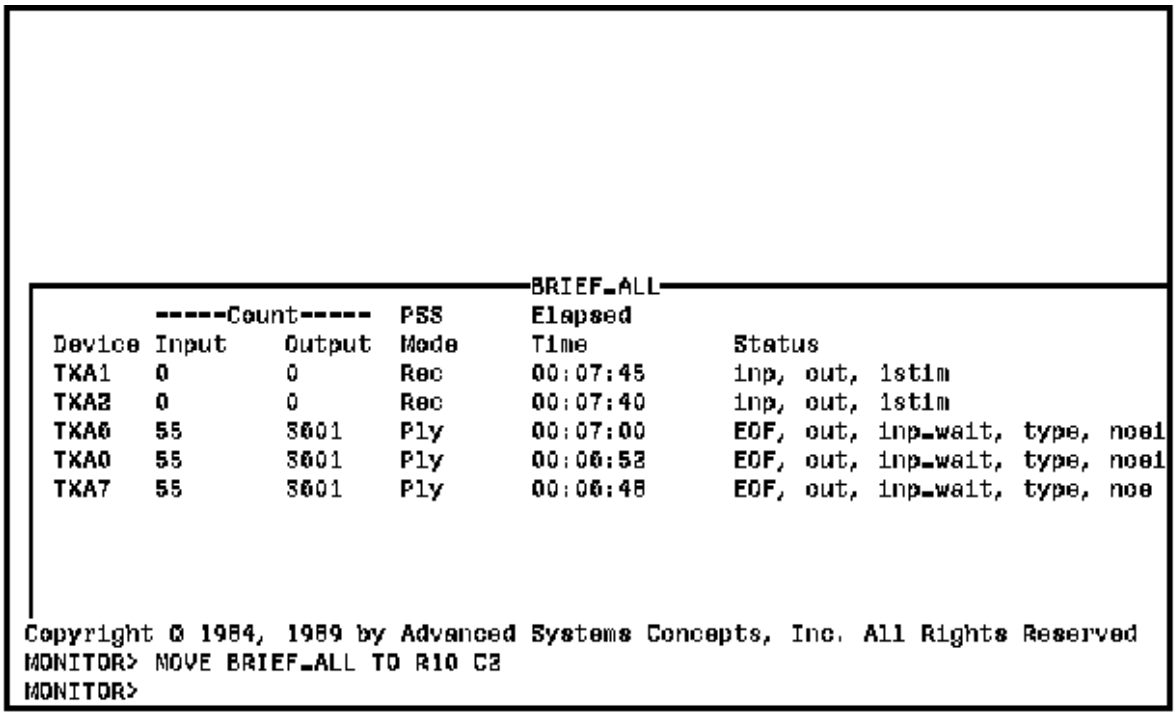


Figure 8–8 MOVE BRIEF_ALL After Execution



8.5.5 PASTE

This command allows you to move the selected window to the front of the display, ahead of all other windows. To view the BRIEF_ALL window, for example, when the first window displays a BRIEF_ALL session and the second and third windows display BRIEF_RECORD and FULL_RECORD_TX34, issue the following command:

```
MONITOR> PASTE BRIEF_ALL
```

8.5.6 PRINT

This command allows you to print the current screen with all visible windows. The screen is spooled to a selected VMS queue. The command format for PRINT is:

```
MONITOR> PRINT [queue copies form]
```

In this example **queue** is a VMS queue name (SYS$PRINT is the default), **copies** is the number of printed copies you would like (one, 1, is the default), and **form** is a site-specific form name (DEFAULT is the default).

8.5.7 REFRESH

To refresh the screen, issue the following command:

```
MONITOR> REFRESH
```

You can also use Ctrl W to refresh the screen.

8.5.8 REMOVE

This command allows you to permanently remove the specified window. For example, if you want to remove the BRIEF_ALL window, issue the following command:

```
MONITOR> REMOVE BRIEF_ALL
```

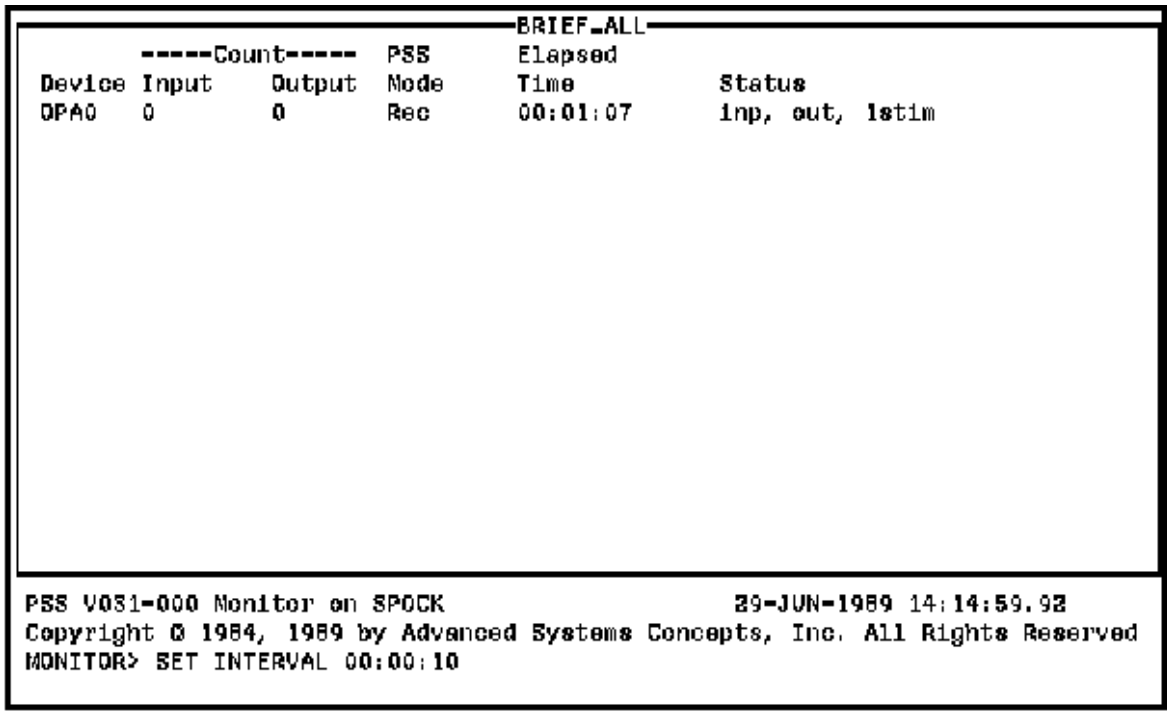
8.5.9 SET

This command allows you to override the internal monitor parameters at which the MONITOR updates the windows on the screen. Issue the SET command, as follows:

```
MONITOR> SET INTERVAL delta_time
```

The **delta_time** parameter is specified using VMS time syntax, no "days" can be included. Figure 8–9 illustrates this command.

Figure 8–9 SET INTERVAL Command



8.5.10 TOP

This command allows you to reset the screen to the top of the current window.

For example, to redisplay information from the first PSS session after entering a number of ADVANCE FULL_ALL commands, issue the following command:

```
MONITOR> TOP FULL_ALL
```

8.5.11 UNPASTE

This command temporarily hides the specified window from view.

```
MONITOR> UNPASTE BRIEF_ALL
```

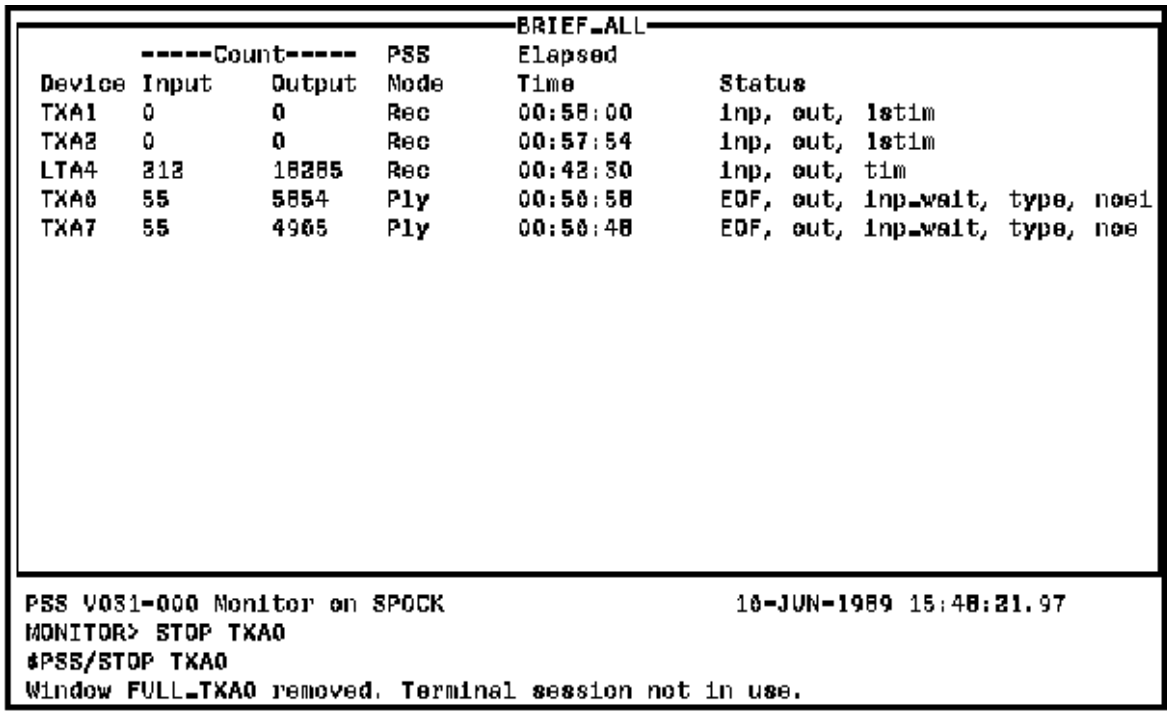
To return the window to view, issue the following command.

```
MONITOR> PASTE BRIEF_ALL
```

8.6 Entering PSS Commands

To enter PSS commands at the PSS MONITOR prompt, simply omit the "PSS/" and enter the rest of the command as discussed in Section 5.2. Figure 8–10 shows a STOP command, entered through the MONITOR. All PSS commands (except MONITOR, REPORT, and SHUT) can be executed through the MONITOR utility.

Figure 8–10 Entering PSS Commands through MONITOR

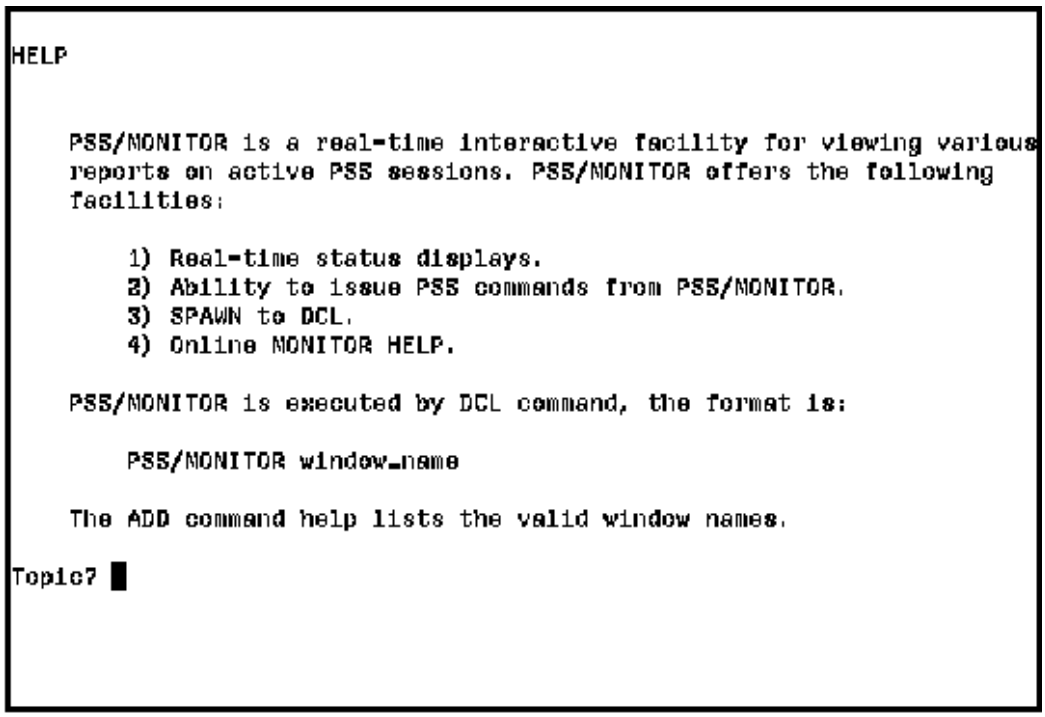


When TXA0 has been stopped, the window FULL_TXA0 is automatically removed. For further information on each of the commands, read Section 5.2.

8.7 Entering the HELP and SPAWN Commands

PSS MONITOR provides on-line help on all of its facilities, as well as the PSS product itself. Simply type HELP at the MONITOR prompt.

Figure 8–11 MONITOR On-Line Help



PSS MONITOR also supports the SPAWN DCL command which allows you to spawn a subprocess during which time you can execute other DCL commands. Type SPAWN at the MONITOR prompt to create a subprocess.

9 PRODUCT SECURITY

9.1 Terms and Concepts

PSS provides many features which permit you to determine application performance, reliability and response, and increase system security. To ensure that only authorized users invoke and use PSS properly, PSS provides a security subsystem. This subsystem allows selected individual(s) the ability to control access to PSS at several different levels.

PSS security categorizes potential PSS users into two classes:

- Administrators
- PSS Users

An *Administrator* is one or more individuals who control access to the PSS product. An Administrator can authorize new administrators and/or new PSS users. An Administrator is identified by his VMS Username. To prevent unauthorized access by VMS-privileged users, PSS administrators must also enter their *administrator-access password* prior to performing administrator-type functions.

A *PSS User* is one or more individuals, allowed access to the PSS product. A PSS User is identified by his VMS Username. PSS Users are given controlled access to *objects*. Table 9–1 contains a list of the current objects that PSS controls. The PSS security system uses a concept called Positive Authorization. This means that PSS Users must be specifically authorized to access an object. In the absence of a security record, no access is permitted. PSS also provides a wildcard capability, as well as a flexible set of objects, so that an Administrator does not have to enter multiple-security records for a PSS User.

Table 9–1 PSS Security Objects

Object	Description
Username	VMS Username. Wildcard (*) permitted.
Group	VMS Group UIC. Numeric-decimal value.
Terminal	VMS Terminal Devicename. Wildcard (*) permitted.
Server	LAT Server/Port-name or X29 Network-name.DTE

Each of these objects can be individually controlled, by entering one or more security records on behalf of the PSS User. Table 9–2 indicates the type of access that can be specified, for the PSS User, when accessing the object.

Table 9–2 PSS Access Operations

Access	Description
RI	Record Input. Allow user to record the terminal input data stream. Ignored for pseudo terminals.
RO	Record Output. Allow user to record the terminal output data stream. Ignored for pseudo terminals.
PI	Playback Input. Allow user to playback an input script.
PO	Playback Output. Allow user to capture the output of a playback session.
ALL	Full access. Same as RI,RO,PI,PO.
NONE	No access.

PSS also provides a feature which allows for positive identification of the authorized user by requiring that password(s) be entered, when requesting RECORD and/or PLAYBACK functions.

An administrator accesses the PSS security system by using the PSS/AUTHORIZE command.

9.2 Administrator Maintenance

9.2.1 Administrator Command Summary

AUTHORIZE/ADMINISTRATOR

This command provides several functions, relating to the overall maintenance of PSS administrator records.

FORMAT

PSS/AUTHORIZE/ADMINISTRATOR *Function Username*

Command Qualifiers	Defaults
/LOG	*/NOLOG*

restrictions

All AUTHORIZE commands are available only to authorized administrators

prompts

Function: function
Username: username

PARAMETERS

function

One of the following valid functions: ADD, UPDATE, DELETE, or LIST

username

Must be a valid VMS Username

Wildcard (*) is allowed only when using the LIST function.

DESCRIPTION

The PSS/AUTHORIZE/ADMINISTRATOR command allows an administrator to control and otherwise maintain administrator access to the system.

COMMAND QUALIFIERS

/LOG
/NOLOG
This qualifier causes positive confirmation of an operation's success by displaying an informational success message. /NOLOG is the default.

EXAMPLES

1 $ **PSS/AUTHORIZE/ADMINISTRATOR ADD ADAM**

This command adds the VMS User, ADAM, as a valid administrator.

2 $ **PSS/AUTHORIZE/ADMINISTRATOR DELETE ADAM**

This command deletes the VMS administrator, named ADAM. ADAM is no longer an administrator.

9.2.2 Creating an Administrator

When the PSS product is installed, a single administrator record is created for VMS Username SYSTEM. The password is ACCESS. ASCI recommends that you change this password or delete the SYSTEM administrator record immediately after creating your own administrator records.

To create an administrator record, type the following command:

```
$ PSS/AUTHORIZE/ADMINISTRATOR ADD username
```

This command will add an administrator record for the specified VMS Username. Section 9.2.1 provides a complete description of all qualifiers which are available. When adding an administrator (or invoking any AUTHORIZE function), you will be prompted for your administrator access password. This password will not echo. If entered correctly, you will be prompted twice for the new administrator's password. This password will also not echo and that is why you are prompted twice as a precaution against mistyping the password.

9.2.3 Updating an Administrator

To update an administrator record, type the following command:

```
$ PSS/AUTHORIZE/ADMINISTRATOR UPDATE username
```

The update process allows you to change an administrator's password. When updating an administrator, you will be prompted for your administrator access password. This password will not echo. If entered correctly, you will be prompted twice for the updated administrator's password. This password will also not echo and that is why you are prompted twice as a precaution against mistyping the password.

9.2.4 Deleting an Administrator

To delete an administrator record, type the following command:

```
$ PSS/AUTHORIZE/ADMINISTRATOR DELETE username
```

Note that if you accidentally delete all the administrator records, you will be effectively denied access to the PSS product from a control point-of-view. If this happens, follow this procedure:

- Login to VMS, using the SYSTEM account, and rename the file, PSS_PRODUCT:PSS_AUTHORIZE.DAT, to PSS_PRODUCT:PSS_ AUTHORIZE.DAT_OLD.

- Attempt to add a new administrator. PSS will create a new version of the security file, with the single administrator record, named SYSTEM. You can then proceed, as discussed in Section 9.2.2.

Note: The PSS_AUTHORIZE.DAT file *must* be created from the SYSTEM account. PSS will pass back an error if another account is used.

9.2.5 Listing Administrators

To obtain a listing of one administrator or all administrators, type the following command:

```
$ PSS/AUTHORIZE/ADMINISTRATOR LIST username
```

To select a specific administrator, enter that administrator's username as input to the "username" prompt. To select all administrators, type an asterisk (*) to the username parameter.

```
$ PSS/AUTHORIZE/ADMINISTRATOR LIST *
Your Administrator Password?

Performance Simulation System V031-000
List of Authorized PSS Administrators

SAMPLE
SYSTEM
```

9.3 User Maintenance

9.3.1 Creating a PSS User

To create a PSS User record, type the following command:

```
$ PSS/AUTH/SECU/TYPE=type/OBJECT=string ADD username
```

This command will add a PSS User record for the specified VMS Username. Section 9.3.5 provides a complete description of all qualifiers which are available. When adding a PSS User (or invoking any AUTHORIZE function), you will be prompted for your administrator-access password. This password will not echo. A PSS User may have one or more security entries, relating to many different objects.

9.3.2 Updating a PSS User

To update a PSS User record, type the following command:

```
$ PSS/AUTH/SECU/TYPE=type/OBJECT=string UPDATE username
```

The update process allows you to change a PSS User's access rights to an object. When updating a PSS User, you will be prompted for administrator access password. This password will not echo.

9.3.3 Deleting a PSS User

To delete a PSS User record, type the following command:

```
$ PSS/AUTH/SECU/TYPE=type/OBJECT=string DELETE username
```

This command will delete the security record for the specified user and object.

9.3.4 Listing PSS Users

To obtain a listing of one PSS User or all PSS Users, type the following command:

```
$ PSS/AUTH/SECU LIST username
```

To select a specific PSS User, enter that PSS User's username as input to the "username" prompt. To select all PSS Users, type an asterisk (*) to the username parameter.

```
$ PSS/AUTHORIZE/SECURITY LIST *
Your Administrator Password?

Performance Simulation System V031-000
List of Authorized PSS Users

Username         Operations   Type      Object
ADAM             RI,RO,PI,PO  Username  *
KEITH            RI,RO,PI,PO  Terminal  *
RANDI            RI,RO,PI,PO  Terminal  *
SYSTEM           RI,RO,PI,PO  Terminal  *
```

9.3.5 User Security Command Summary

AUTHORIZE/SECURITY

This command provides several functions, relating to the overall maintenance of PSS user records.

FORMAT

PSS/AUTHORIZE/SECURITY/qualifiers *Function Username*

Command Qualifiers	Defaults
/LOG	*/NOLOG*
/OBJECT=string	*See Text*
/OPERATION=access	*/OPERATION=(NONE)*
/TYPE=type_object	*See Text*

restrictions

All AUTHORIZE commands are available only to authorized administrators

prompts

Function: function
Username: username

PARAMETERS

function

One of the following valid functions: ADD, UPDATE, DELETE, or LIST

username

Must be a valid VMS Username

Wildcard (*) can be specified.

DESCRIPTION

The PSS/AUTHORIZE/SECURITY command allows an administrator to control and otherwise maintain PSS user access to the system.

COMMAND QUALIFIERS

/LOG
/NOLOG

This qualifier causes positive confirmation of an operation's success by displaying an informational success message. /NOLOG is the default.

/OBJECT=string

This qualifier, mandatory for ADD, UPDATE, and DELETE functions, indicates the object to which you want to grant or withdraw access. Table 9–1 indicates the acceptable values of this qualifier, based upon the /TYPE qualifier. Enclose SERVER objects within quotes (i.e., "SERVER1/PORT_1"). Note that they are case-sensitive.

/OPERATION=(access,...)
/NOOPERATION=(access,...)

This qualifier indicates the access to be granted or withdrawn for the specified object and username. Valid access codes are those specified in Table 9–2. By default, if /OPERATION is omitted, access is set to NONE.

/TYPE=type_object

This qualifier, mandatory for ADD, UPDATE, and DELETE functions, indicates the type of object specified, using the /OBJECT qualifier. Valid types are those specified in Table 9–1. Only the first character of the object type need be specified.

EXAMPLES

1 $ **PSS/AUTHORIZE/SECURITY/TYPE=T/OBJECT=TXA0: ADD KEITH**

This command adds the VMS User KEITH to the PSS Security database. The object type is TERMINAL and the object is TXA0:. Access is set to NONE.

2 $ **PSS/AUTHORIZE/SECURITY/TYPE=U/OBJECT=KEITH/OPER=ALL ADD KEITH**

This command adds the VMS User KEITH to the PSS Security database. The object type is USERNAME and the object is KEITH. Full access is available. Note that this command would allow the user KEITH to issue RECORD and/or PLAYBACK commands on any terminal he was already logged into.

3 $ **PSS/AUTHORIZE/SECURITY/TYPE=S/OBJECT="SERVER1/PORT_1" (OPER=(PI,PO)) ADD KEITH**

This command adds the VMS User KEITH to the PSS Security database. The object type is SERVER, and the object is SERVER1/PORT_1. Access is set to PI, PO. This object is extremely useful for LAT devices since the assignment of LTA devices is dynamic. Specification of the server allows specific geographic tracking. The SERVER object can also be used for X29 devices. The object specification is "network-name.DTE."

4 $ **PSS/AUTHORIZE/SECURITY/TYPE=G/OBJECT=20/OPER=RO ADD KEITH**

This command adds the VMS User KEITH to the PSS Security database. The object type is GROUP and the object is 20. Access is set to RO. This command would allow user KEITH to record the output data stream of any user, under UIC group 20, who is logged into the system.

5 $ **PSS/AUTHORIZE/SECURITY/TYPE=T/OBJECT=TXA0:/OPER=ALL UPDATE ADAM**

This command updates User ADAM in the PSS Security database. The object type is TERMINAL, and the object is TXA0:. Access is changed to full access.

6 $ **PSS/AUTHORIZE/SECURITY/TYPE=T/OBJECT=TXA0: DELETE KEITH**

This command deletes User KEITH's object TXA0: in the PSS Security database. Any other security entry for User KEITH has not been modified.

7 $ **PSS/AUTHORIZE/SECURITY/TYPE=T/OBJECT=* ADD ***

This command adds a wildcard user entry in the PSS Security database. All users will be able to access all terminals on the system with PSS. This command is listed so that you can disable the security system, although ASCI does not recommend that you do this on anything other than a stand-alone system.

9.4 User Password Maintenance

9.4.1 Creating a Password Record

To create a PSS User password record, type the following command:

```
$ PSS/AUTH/PASSWORD[/RECORD_PASSWORD=string/PLAYBACK_PASSWORD=string]
$__ ADD username
```

This command will add a PSS User password record for the specified VMS Username. Section 9.4.5 provides a complete description of all qualifiers which are available. When adding a PSS User password (or invoking any AUTHORIZE function), you will be prompted for your administrator access password. This password will not echo.

9.4.2 Updating a PSS User Password

To update a PSS User password record, type the following command:

```
$ PSS/AUTH/PASSWORD[/RECORD_PASSWORD=string/PLAYBACK_PASSWORD=string]
$__ UPDATE username
```

The update process allows you to change a PSS User's password record. When updating a PSS User, you will be prompted for your administrator access password. This password will not echo.

9.4.3 Deleting a PSS User Password

To delete a PSS User password record, type the following command:

```
$ PSS/AUTH/PASSWORD[/RECORD_PASSWORD=string/PLAYBACK_PASSWORD=string]
$__ DELETE username
```

This command will delete the password record for the specified user.

9.4.4 Listing PSS User Passwords

To obtain a listing of one or all PSS Users who require password entry, type the following command:

```
$ PSS/AUTH/PASSWORD LIST username
```

To select a specific PSS User, enter that PSS User's username as input to the "username" prompt. To select all PSS Users, type an asterisk (*) to the username parameter. Note that no passwords are displayed. PSS passwords are one-way encrypted.

```
$ PSS/AUTHORIZE/PASSWORD LIST *
Your Administrator Password?

Performance Simulation System V031-000
List of PSS Users Requiring Password Protection

Username         Operations
KEITH            Record
```

9.4.5 User Password Command Summary

AUTHORIZE/PASSWORD

This command provides several functions, relating to the overall maintenance of PSS user password records.

FORMAT

PSS/AUTHORIZE/PASSWORD/qualifiers *Function Username*

Command Qualifiers	Defaults
/LOG	*/NOLOG*
/PLAYBACK_PASSWORD=string	*See Text*
/RECORD_PASSWORD=string	*/See Text*

restrictions

All AUTHORIZE commands are available only to authorized administrators.

prompts

Function: function
Username: username

PARAMETERS

function

One of the following valid functions: ADD, UPDATE, DELETE, or LIST

username

Must be a valid VMS Username

Wildcard (*) is allowed only when using the LIST function.

DESCRIPTION

The PSS/AUTHORIZE/PASSWORD command allows an administrator to require a PSS User to enter a password, for positive indentification, prior to invoking a RECORD and/or PLAYBACK command.

COMMAND QUALIFIERS

/LOG
/NOLOG

This qualifier causes positive confirmation of an operation's success by displaying an informational success message. /NOLOG is the default.

/PLAYBACK_PASSWORD=string

This qualifier requires the named PSS user to enter a password specified above when issuing PLAYBACK commands. By default no password is required.

/RECORD_PASSWORD=string

This qualifier requires the named PSS user to enter a password specified above when issuing RECORD commands. By default no password is required.

EXAMPLES

1 $ **PSS/AUTHORIZE/PASSWORD/RECORD_PASSWORD=ABC ADD KEITH**

This command adds a password requirement for PSS User KEITH. This user must enter a password, whenever attempting RECORD operations.

2 $ **PSS/AUTHORIZE/PASSWORD/PLAYBACK_PASSWORD=DEFG UPDATE KEITH**

This command adds a password requirement for PSS User KEITH. This user must enter a password, whenever attempting PLAYBACK operations.

3 $ **PSS/AUTHORIZE/PASSWORD/NOPLAYBACK_PASSWORD UPDATE KEITH**

This command modifies the password requirement for PSS User KEITH. Using the previous example, this command would remove the requirement for entering a password during PLAYBACK commands.

4 $ **PSS/AUTHORIZE/PASSWORD DELETE KEITH**

This command deletes the password requirement for PSS User KEITH.

10 RESTRICTIONS AND LIMITATIONS

10.1 Restrictions and Limitations

Known restrictions and limitations, concerning PSS, are as follows:

1. PSS can support a recommended maximum of sixty-four (64) Pseudo Terminal Devices.

Note: Depending on your application and hardware, however, this number might or might not be feasible. Please note that this figure can be increased by adding the /MAXUNITS= qualifier to the first CONNECT statement after the running of the SYSGEN utility.

2. The maximum input-waiting time is slightly less than one (1) day. This is a permanent restriction.

3. Terminals, using PSS for playback sessions, must have READSYNC enabled. (PSS does this automatically and resets it to whatever state PSS finds it, when the playback stops.)

11 PRODUCT SUPPORT

PSS for OpenVMS contains system-privileged code and a device driver. While ASCI has thoroughly tested the product, ASCI recommends that you use it for the first time during the nonpeak hours of your computer system. This will allow you to experiment with the pseudo devices and SYSGEN parameters.

EPSS for OpenVMS contains no system-privileged code and runs entirely in user-mode.

PSS and EPSS are sold on a license basis per CPU. Each is sold separately. Send in your "License Registration Certificate" as soon as possible for your warranty support. The Software Product License Agreement, accompanying the software, describes the product warranty. If you encounter any problems or questions, concerning the product, contact your local distributor or Advanced Systems Concepts, Inc., for support.

A PSS MESSAGES

This section describes all of the known-error messages that PSS can display when in use.

NOTE: A secondary error, which further explains the problem(s) you might encounter, accompanies several PSS-error messages. While ASCI made every attempt to document the system errors which can occur, the secondary error code can refer to almost any VMS error. In most cases the error is self-explanatory.

A.1 Messages

ADD_ADMIN, administrator record added for user "user"

Facility: PSSCMD

Severity: Informational

Explanation: A PSS security-administrator record was successfully added to the database.

User Action: None.

ADD_PASSWORD, password record added for user "user"

Facility: PSSCMD

Severity: Informational

Explanation: A PSS password record was successfully added to the database.

User Action: None.

ADD_SECURITY, security record added for user "user"

Facility: PSSCMD

Severity: Informational

Explanation: A PSS security record was successfully added to the database.

User Action: None.

ALLOCERR, error allocating loop block

Facility: PSS

Severity: Warning

Explanation: PSS Server ran out of virtual memory.

User Action: check the PGFLQUOTA and increase if necessary. Otherwise, increase the SYSGEN parameter VIRTUALPGCNT appropriately. This message is accompanied by an additional message.

ASNCHAN, error assigning channel to PSS server mailbox

Facility: PSSCMD

Severity: Fatal

Explanation: This message usually indicates that you did not bring up PSS properly.

User Action:

1. Verify that you started PSS through the command procedure, SYS$MANAGER:PSS_STARTUP.
2. Verify that the driver and server are running.
3. Ensure that you have included sufficient quotas.

ASNERR, error in assigning command mailbox channel

Facility: PSS

Severity: Warning

Explanation: This message indicates that the Assign-Channel-system service returned an error, when the server attempted to communicate with the PSS-control program. The specific-system-service error follows it.

User Action: This message is accompanied by an additional message.

BADINPFIL, input file corrupted

Facility: PSS

Severity: Warning

Explanation: An input script file, either one directly specified or invoked through the %EXEC lexical, was incorrectly formatted.

User Action: Verify that the file is a proper PSS input file.

BADLPBQUEUE, error locating loop block on queue

Facility: PSS

Severity: Warning

Explanation: An internal consistency check occurred within PSS Server. Your playback has been halted.

User Action: If you are under Warranty or Maintenance service, please contact ASCI.

CLSERR, error closing "file"

Facility: PSS

Severity: Warning

Explanation: This message indicates that an error occurred, while the server was in the process of closing a file. A more detailed explanation of the problem follows.

User Action: None.

CONFQUAL, invalid, multiple-type qualifiers in FROM/TO specification

Facility: PSSCVT

Severity: Error

Explanation: You can specify only one file-type qualifier for each file in a conversion.

User Action: Verify your actions.

CRESEC, security file "file" created

Facility: PSSCMD

Severity: Informational

Explanation: The PSS security database has been successfully created.

User Action: None.

DELETE_ADMIN, administrator record deleted for user "user"

Facility: PSSCMD

Severity: Informational

Explanation: A PSS security-administrator record was successfully deleted from the database.

User Action: None.

DELETE_PASSWORD, password record deleted for user "user"

Facility: PSSCMD

Severity: Informational

Explanation: A PSS password record was successfully deleted from the database.

User Action: None.

DELETE_SECURITY, security record deleted for user "user"

Facility: PSSCMD

Severity: Informational

Explanation: A PSS security record was successfully deleted from the database.

User Action: None.

DEVERR, error obtaining device information on "device"

Facility: PSSCMD

Severity: Error

Explanation: A terminal device, specified as part of a PSS/AUTHORIZE function, cannot be accessed.

User Action: This message is accompanied by one or more additional messages which provide more information, concerning the error.

EOFRSPNCT, end of file with response buckets "n" active

Facility: PSSCVT

Severity: Warning

Explanation: PSS encounters an end-of-file read with no matching %END_CHECKPOINT lexical.

User Action: Verify the lexicals you used.

EOFUNSICT, end of file with unsolicited-input active

Facility: PSSCVT

Severity: Warning

Explanation: PSS encounters the end of the input-text file without detecting a %END_UNSOLICITED lexical.

User Action: Verify the lexicals you used.

ERRCRESEC, an error occurred creating new PSS security file

Facility: PSSCMD

Severity: Error

Explanation: Self-explanatory

User Action: Verify privileges. You must be logged into the SYSTEM account to create a security file.

GRPNOTNUM, group UIC specified not numeric

Facility: PSSCMD

Severity: Error

Explanation: A PSS/AUTHORIZE/SECURITY/TYPE=GROUP specified an object which was not numeric.

User Action: Correct and resubmit.

ILLOBJTYP, illegal object type. Type must be U, S, G, T

Facility: PSSCMD

Severity: Error

Explanation: Self-explanatory

User Action: Correct and resubmit.

ILLOPER, illegal operation specified

Facility: PSSCMD

Severity: Error

Explanation: An illegal, access-operation code was specified. Valid codes are RO, RI, PO, PI, ALL.

User Action: Correct and resubmit.

ILLNUMSPLIT, illegal number of SPLIT_TIMES specified. Min is 2, Max is 10

Facility: PSSCMD

Severity: Error

Explanation: Self-explanatory

User Action: Correct and resubmit.

ILLREC, security file has been tampered with. Access denied

Facility: PSSCMD

Severity: Error

Explanation: The PSS security database has been tampered with and modified without using the PSS/AUTHORIZE commands. The user record specified will not be used by PSS for any authorization purposes.

User Action: use the PSS/AUTHORIZE DELETE function to remove the tampered record.

INITERR, error during Server Initialization

Facility: PSS

Severity: Fatal

Explanation: This message indicates that a fatal error occurred, while the server initialized itself. A message follows that further explains the problem.

User Action: Verify that you've loaded PTDRIVER and connected at least one pseudo device (PT).

INSUFPARAM, insufficient PARAM(s) or LOCAL_PARAM(s) passed

Facility: PSS

Severity: Error

Explanation: A %PARAM or %LOCAL_PARAM lexical specified a parameter number higher than that actually passed through either DCL or the %EXEC lexical.

User Action: Verify, correct, and resubmit.

INVCON, invalid conversion operation

Facility: PSSCVT

Severity: Fatal

Explanation: The "from" and "to" files are not of different types or you made an attempt to convert from one PSS-internal format to another.

User Action: Verify your actions.

INVNUMB, invalid number, set to "n"

Facility: PSSCVT

Severity: Warning

Explanation: The lexical, you specified, contains an invalid, non-numeric character.

User Action:

- Verify the value, you specified, in the lexical.
- Correct and resubmit.

INVREQTYP, unsupported conversion operation

Facility: PSSCVT

Severity: Error

Explanation: PSS does not support conversion to /OUTPUT or /PLAYBACK.

User Action: Verify your actions.

INVTIME, invalid time value, set to zero

Facility: PSSCVT

Severity: Warning

Explanation: A %TIME lexical, contains an illegal- or invalid-time specification.

User Action: Verify that the time specification reads as valid VMS-delta time (with no day portion mentioned).

INVWIND, invalid window name

Facility: PSSCMD

Severity: Error

Explanation: An invalid window name was entered, when executing PSS/MONITOR from DCL.

User Action: Table 8–2 lists the valid window names that can be entered.

KMODESHUT, PSS Server Shutdown at "date/time" due to unexpected exception

Facility: PSS

Severity: Informational

Explanation: An unexpected exception occurred in Kernel mode (IPL=0) within the Server.

User Action: This message is accompanied with an additional message which denotes the exception. PSS Server will exit properly. However, PSS will be unusable until the PSS_STARTUP procedure is invoked. If you are under Warranty or Maintenance service, please contact ASCI.

LEXERR, lexical syntax error on line "n"

Facility: PSSCVT

Severity: Warning

Explanation: This message indicates that PSS detects an error, while processing an input lexical. The source-input-text file identifies the line number. A secondary message, which provides more detailed information, always accompanies this message.

User Action: Based on the secondary message, verify the lexical syntax and resubmit.

LOOPINTEXC, Loop interval exceeded—possible response skewing

Facility: PSS

Severity: Warning

Explanation: This message appears, if EPSS determines that PSSEXT does not have enough CPU resources to perform all of its required actions. As a result, you might experience higher response times.

User Action: Determine, whether you can bring down other programs running on the RTE to relieve the strain. If not, try raising PSSEXT's priority.

NEWFILERR, error initializing new-input file

Facility: PSS

Severity: Warning

Explanation: This message indicates that an error occurs, when the server attempts to initialize or open a new input file. A more detailed explanation of the error follows.

User Action: None.

NOADMIN, you are not a registered ADMINISTRATOR

Facility: PSSCMD

Severity: Error

Explanation: Only a registered PSS administrator can issue PSS/AUTHORIZE functions.

User Action: Self-explanatory.

NOEXPTIM, experiment time missing

Facility: PSSCVT

Severity: Warning

Explanation: A PSS/CONVERT/ABSOLUTE operation was specified and the input file's first lexical was not %EXPER.

User Action: Verify that the %EXPER lexical is the first line in the input script file.

NOLOGNAM, error accessing PSS_PSEUDO_DEVICE logical name

Facility: PSSCMD

Severity: Error

Explanation: The logical mentioned in the message was not found.

User Action: Verify that PSS_STARTUP has been successfully executed.

NOSECFILE, error accessing PSS Security File

Facility: PSSCMD

Severity: Error

Explanation: The PSS security file cannot be opened for read/write access.

User Action: This message is accompanied with one or more additional messages which further describe the problem.

NOSUCHDEV, no such device "device"

Facility: PSSCMD

Severity: Error

Explanation: The device you specified in the command does not exist.

User Action:

1. Verify actions.
2. Ensure that you spell and reference the device correctly.

NOTAUTHOPER, user not authorized for specified operations on terminal

Facility: PSSCMD

Severity: Error

Explanation: Your security record does not allow the operations you have requested for that terminal device.

User Action: Speak with your PSS security administrator, if there is a problem.

NOTAUTHUSER, user not authorized for access to terminal

Facility: PSSCMD

Severity: Error

Explanation: A security record does not exist which will allow you access to the terminal device.

User Action: Speak with your PSS security administrator if there is a problem.

NOTSYSTEM, user not logged into SYSTEM account

Facility: PSSCMD

Severity: Error

Explanation: An attempt was made to create a PSS security file, and the user was not logged into the SYSTEM account.

User Action: If you want to create a new PSS security file, log into the SYSTEM account.

NOTTERM, Device "device" is not a terminal

Facility: PSSCMD

Severity: Fatal

Explanation: This message appears if you issue a PSS command, concerning a device which is not a terminal.

User Action:

1 Verify your actions.

2 Ensure that the device is a terminal.

OPNERR, Error creating/opening file "file"

Facility: PSSCMD

Severity: Fatal

Explanation: The PSS Command attempted to verify that it could open and/or create the file and found an error. A secondary message, which provides more information, always follows this message.

User Action: The causes usually lie in privileges, incorrect/illegal directory/specification, and quotas. Ensure that PSS can access the named files and/or directories for write operations.

OUTRANGE, value out of range, set to "n"

Facility: PSSCVT

Severity: Warning

Explanation: You specified a value for the lexical that is out of the valid range. Thus, it defaulted, as shown.

User Action:

1 Verify the lexical you used and its appropriate range.

2 Correct and resubmit.

PSSNOTUP, PSS/EPSS server is not running

Facility: PSSCMD

Severity: Error

Explanation: The PSS/EPSS server is not running.

User Action: Execute PSS_STARTUP.COM to bring up the PSS system.

PSWDNOMATCH, password entered does not match with security file

Facility: PSSCMD

Severity: Error

Explanation: The password you entered (or omitted) does not match the password you are required to enter for either RECORD and/or PLAYBACK operations.

User Action: verify that you are entering the correct password for the type of operation specified.

RECALREXTS, security record for "user" already exists

Facility: PSSCMD

Severity: Error

Explanation: A PSS/AUTHORIZE ADD command was entered for a user/administrator which already exists in the security file.

User Action: Verify and resubmit.

RECINP, "n" records read from file "file"

Facility: PSSCVT

Severity: Informational

Explanation: This message indicates the number of records PSS reads from the input file you specified.

User Action: None.

RECINPL, "n" lines read from file "file"

Facility: PSSCVT

Severity: Informational

Explanation: This message indicates the number of lines PSS reads from the input file you specified.

User Action: None.

RECNOTFND, Record "user" not found

Facility: PSSCMD

Severity: Error

Explanation: A PSS/AUTHORIZE command was entered for a user/administrator which could not be found in the security file.

User Action: Verify and resubmit.

RECOUT, "n" lines written to file "file"

Facility: PSSCVT

Severity: Informational

Explanation: This message indicates the number of lines, written to the output file you specified.

User Action: None.

RECOUTR, "n" records written to file "file"

Facility: PSSCVT

Severity: Informational

Explanation: This message indicates the number of records, written to the output file you specified.

User Action: None.

RECTFB, record at line "n" continued on following line

Facility: PSSCVT

Severity: Warning

Explanation: This message indicates that the input source file, while being converted to PSS text, detected a line which exceeded the maximum-record size, specified for the operation (by default, 2,048 bytes). PSS writes the input source at its maximum-record size and writes the remaining data to the next data record.

User Action: Ensure that the /RECORDSIZE qualifier (and its default) is appropriate.

REQPRMABS, you must specify INPUT, /OUTPUT, /PLAYBACK, or /TEXT

Facility: PSSCVT

Severity: Error

Explanation: You did not specify the conversion operation.

User Action: Verify actions.

REQPRMABS, /OUTPUT or /INPUT must be specified

Facility: PSSCMD

Severity: Error

Explanation: PSS requires a valid file specification, when you use /INPUT or /OUTPUT qualifiers.

User Action:

1 Verify actions.

2 Correct, and resubmit.

RPTERR, error in Report Command processing

Facility: PSS

Severity: Warning

Explanation: An error occurred while the server was processing a PSS REPORT command. this message is followed by a more detailed explanation of the problem.

User Action: None.

RSPALRENA, specified response-time bucket, already enabled

Facility: PSSCVT

Severity: Warning

Explanation: PSS encounters two or more %BEGIN_CHECKPOINT lexicals for the same response-time bucket with no intervening %END_CHECKPOINT lexical.

User Action: Verify the lexicals you used.

RSPNOTENA, specified-response-time bucket, not enabled

Facility: PSSCVT

Severity: Warning

Explanation: PSS encounters a %END_CHECKPOINT lexical without a matching %BEGIN_CHECKPOINT.

User Action: Verify the lexicals you used.

SCHERR, schedule Wakeup Error

Facility: PSS

Severity: Fatal

Explanation: This message indicates the Schedule-Wakeup-System service returned an error. The actual-system-service error follows.

User Action: None.

SECERR, security error in recording terminal "device"

Facility: PSSCMD

Severity: Error

Explanation: PSS encounters a security error, while attempting to record a terminal.

User Action:

1. Verify that your privileges are appropriate for the type of terminal you want to record.
2. Verify that you have spelled your INPUT password correctly.

SECNOTFND, security file not found

Facility: PSSCMD

Severity: Informational

Explanation: The PSS security database cannot be accessed.

User Action: One or more messages follow which more accurately describe the severity of this message.

TERMNOTLOG, terminal not logged in

Facility: PSSCMD

Severity: Error

Explanation: A PSS/RECORD/PERMANANT was issued to a terminal that was not logged in.

User Action: Verify the terminal you want to record and ensure that it is logged in.

UNSALRACT, unsolicited input, already active

Facility: PSSCVT

Severity: Warning

Explanation: PSS encounters a %BEGIN_UNSOLICITED lexical with no intervening %END_UNSOLICITED.

User Action: Verify the lexicals you used.

UNSFROM, unsupported FROM file type

Facility: PSSCVT

Severity: Error

Explanation: PSS does not directly support the file type you specified.

User Action: You can specify any file type so long as you also specify the conversion qualifier. In this case, you specified an unsupported-file type without any conversion qualifier.

UNSNOTACT, unsolicited input, not active

Facility: PSSCVT

Severity: Warning

Explanation: PSS encounters a %END_UNSOLICITED lexical with no matching %BEGIN_UNSOLICITED.

User Action: Verify the lexicals you used.

UPDATE_ADMIN, administrator record updated for user

Facility: PSSCMD

Severity: Informational

Explanation: A PSS security-administrator record was successfully updated in the database.

User Action: None.

UPDATE_PASSWORD, password record updated for user

Facility: PSSCMD

Severity: Informational

Explanation: A PSS password record was successfully updated in the database.

User Action: None.

UPDATE_SECURITY, security record updated for user

Facility: PSSCMD

Severity: Informational

Explanation: A PSS security record was successfully updated in the database.

User Action: None.

USRMBXERR, error accessing user mailbox

Facility: PSS

Severity: Warning

Explanation: An input script, containing a %MAIL lexical was executed and PSS Server could not access the user's mailbox, using the logical name specified.

User Action: check the spelling of the logical name and ensure that the user-written program is running.

WRTERR, error writing to file

Facility: PSS

Severity: Warning

Explanation: This message indicates that an error occurred, when the server attempted to write a record to a file. A more detailed explanation of the problem follows this message.

User Action: None.

GLOSSARY

The expressions in this GLOSSARY are alphabetically arranged, strictly by the letters themselves, regardless of spaces, underscores, quotation marks, brackets, slashes, or special characters. When identically spelled, capitalized expressions precede those in lower-case letters, and those expressions, preceded by slashes, precede those without. In addition acronyms are defined only by the spelled-out names they represent. For a full definition look under the complete name of the respective file, service, or subsystem.

A

Administrator: A person who assigns the type of access that individuals, including other PSS administrators, need to use PSS

D

Device name: A physical device name that identifies a particular hardware device; the name the operating system uses to refer to a hardware device

G

Group: A user-defined division of an ISAM (Indexed Sequential Access Method) file, containing records or indexed entries (Once specified, *group* size is constant throughout all parts of an ISAM file.)

I

I/O-request packet: A data structure that the Queue I/O system service constructs to describe a user's I/O request to the device driver

IRP: The acronym for an I/O-request packet

IRPCOUNT parameter: A VMS SYSGEN parameter that affects the size of the fixed-area IRP pool

N

NPAGEDYN parameter: A VMS SYSGEN parameter which affects the initial size of the dynamic nonpaged pool (Nonpaged pool is used for a variety of system buffering uses and the loading of device drivers)

Nonpaged pool: The area of physical memory, used by VMS for a variety of system-buffering uses

P

PASTE: A MONITOR command that moves a selected window to the front of the display, ahead of all other windows

Pseudo terminal: A PSS device that acts like a real terminal, but doesn't actually exist

PSS MONITOR utility: A utility that allows you to display a full range of active sessions on any DEC-supported terminal and to issue most PSS commands, as well as the SPAWN and HELP commands

PSSCMD: A component that executes all DCL-command processing for the PSS product (except for the /CONVERT and /MODIFY commands)

PSSEXT: An EPSS component that performs RTE services, on one system, to test another system's applications

PSSRV: The PSS server component that processes in both the kernel and user modes and writes all collected data via the PT driver to one or more files per terminal

PSUA: The default for the device name of pseudo terminals

PTA: The old default for the device name of pseudo-terminal units, prior to V3.1

PTDRIVER: The VMS driver component that intercepts and collects the input and/or output data of a specific terminal in the recording mode and provides support for a new pseudo-terminal device

R

REMOVE: A MONITOR command that permanently removes a specified window

RTE: A Remote Terminal Emulator, a system that simulates terminal traffic in order to test an application system for performance or reliability

RTE port: A device on the RTE system which is used to communicate with the System Under Test

Response-level measurement: A type of calculation that computes the response time, beginning with the termination of a READ request and ending with the first output data, sent to you

S

Server: A PSS object, representing a LAT Server/Port name or X29 Network name (DTE)

Service-level-response measurement: A type of calculation, which PSS and EPSS perform, that computes the response time, beginning with the termination of a READ request and ending with a new READ request and that determines the overall time you must wait to perform the next computer-related activity

SPAWN: A command that spawns a subprocess to allow you to execute other DCL commands

SUT: System Under Test

System Under Test: A system upon which the application to be tested is actually running

T

Terminal: A PSS object, representing a VMS Terminal device name

U

Unit: An integer from 0 to 65537 that indicates the number of devices within a specific controller

UNPASTE: A MONITOR command that hides a specified window from view

User: A person, authorized to have (limited) access of PSS-controlled objects

User name: A PSS object, representing a VMS User name

V

VMSINSTAL: A command procedure, which Digital Equipment Corporation supplies, that installs software products from their distribution media to your system

VMS SYSGEN utility: A utility that can load device drivers and their controller/units, as well as modify various VMS parameters

X

XON/XOFF: The abbreviation for transmitter on/transmitter off; an item on the line-characteristics menu, used to synchronize data transmission between the Professional and a host

Index

A

B

C

D

E

G

H

I

L

M

N

O

P

R

S

T

U

V