digital

VAX/VMS
System Manager's Guide

Order No. AA-D027A-TE

VAX11

August 1978

This document covers the following tasks of VAX/VMS system management: 1) setting up users' accounts, 2) managing public files and volumes, 3) controlling overall system performance, 4) monitoring system activity, and 5) recognizing and dealing with errors and failures.

The aims of this guide are to provide a background for understanding these tasks and to provide rules and guidelines for performing them.

VAX/VMS
System Manager's Guide

Order No. AA-D027A-TE

17

SUPERSESSION/UPDATE INFORMATION:	This is a new document for this release.
OPERATING SYSTEM AND VERSION:	VAX/VMS V01
SOFTWARE VERSION:	VAX/VMS V01

To order additional copies of this document, contact the Software Distribution Center, Digital Equipment Corporation, Maynard, Massachusetts 01754

digital equipment corporation · maynard, massachusetts

First Printing, August 1978

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

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

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

The postage-prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DIGITAL	DECsystem-10	MASSBUS
DEC	DECtape	OMNIBUS
PDP	DIBOL	OS/8
DECUS	EDUSYSTEM	PHA
UNIBUS	FLIP CHIP	RSTS
COMPUTER LABS	FOCAL	RSX
COMTEX	INDAC	TYPESET-8
DDT	LAB-8	TYPESET-11
DECCOMM	DECSYSTEM-20	TMS-11
ASSIST-11	RTS-8	ITPS-10
VAX	VMS	SBI
DECnet	IAS	

CONTENTS

CONTENTS (Cont.)

FIGURES

Page

FIGURES (Cont.)

TABLES

PREFACE

The VAX/VMS System Manager's Guide is both a useful adjunct to a formal course in VAX/VMS system management and an indispensable reference book for every VAX/VMS system manager.

MANUAL OBJECTIVES

The objectives of this guide are twofold. The first objective is to give the reader an understanding of the reasons for performing certain tasks of VAX/VMS system management. The second objective is to show the reader how to perform the important tasks of VAX/VMS system management.

INTENDED AUDIENCE

This guide is intended for any person who has overall responsibility for controlling the operations of a VAX/VMS installation. The reader of this guide is most likely a data processing generalist, not necessarily a programmer and probably not a systems programmer.

STRUCTURE OF THIS DOCUMENT

This system manager's guide consists of six parts, as follows:

Part I: Introduction to System Management

Duties of the system manager and a general introduction to the major components of the VAX/VMS system

Part II: Setting Up and Using a System of Accounts

The user authorization file (UAF), groups, limits, priority, privileges, accounting, and use of the AUTHORIZE program

Part III: Managing Public Files and Volumes

Initializing and mounting public volumes, backing up files on public volumes, creating permanent global sections and installing known images, and defining system logical names

Part IV: Overall Control of the System

Maintaining the start-up command procedures, setting up spooling, and creating and controlling batch queues and print queues

Part V: Monitoring the Activity of the System

Using the DISPLAY program and the operator's log file

Part VI: Recognizing and Dealing with Errors and Failures

Error logging and using software performance reports (SPRs)

ASSOCIATED DOCUMENTS

The VAX-11 Information Directory lists and describes all the documents the system manager may need to refer to in the course of performing system management duties.

For general background information about the system, see the VAX/VMS Summary Description and the VAX/VMS Primer.

The following documents may also be useful:

VAX/VMS Command Language User's Guide
VAX/VMS Operator's Guide
VAX/VMS Release Notes
VAX-11 Software Installation Guide
VAX/VMS System Messages and Recovery Procedures Manual

PART I

INTRODUCTION TO SYSTEM MANAGEMENT

This introduction to VAX/VMS system management contains both a definition of system management and a profile of the VAX/VMS system.

Part I contains two chapters:

- Chapter 1: Duties of the System Manager
- Chapter 2: Components of the VAX/VMS Operating System

CHAPTER 1

DUTIES OF THE SYSTEM MANAGER

In a nutshell, a system manager of a VAX/VMS installation has two main tasks:

- To make decisions that have to do with optimizing the overall performance and efficiency of the system
- To perform procedures that have to do with the overall management and control of the system

To make good decisions, the system manager has to understand both the needs of users and the capabilities of the VAX/VMS operating system. To perform system management procedures well the system manager has to have a working knowledge of the functions of the VAX/VMS operating system.

The pattern followed throughout this guide is to discuss the issues and the facts that will help the system manager make decisions and then to give general rules and guidelines for performing the procedures of system management.

It is not possible to prescribe a precise set of formulas for setting up and running the VAX/VMS system. To be blunt, system management cannot be done by rote or from a cookbook. Rather, the system manager must -- by combining an understanding of users' needs and system capabilities with a working knowledge of the functions of VAX/VMS -- work out a coherent strategy for effective system management.

1.1 WHAT IS SYSTEM MANAGEMENT?

In its most abstract sense, system management means the overall control of the operations of a computer system for the benefit of the users of the system. System management is a function that may be exercised by a single system manager, who is assisted by system operators, or it is a function that may be shared by several persons, all of whom or some of whom may also serve as system operators as well as system managers.

A computer installation exists not for its own sake but to serve its users. Ideally, then, it should be operated to provide service to all users with efficiency and economy. This is the challenge of system management.

1.2 TASKS THAT THE SYSTEM MANAGER PERFORMS

For practical purposes, system management is best defined in terms of the tasks that the system manager typically performs. The tasks of system management fall into the following six categories:

1. Getting the system up and running
2. Setting up users' accounts
3. Managing public files and volumes
4. Controlling the overall performance of the system
5. Monitoring the activity of the system
6. Recognizing and dealing with errors and failures

The first topic (getting the system up and running) is the subject of the VAX-11 Software Installation Guide. The remaining five topics are the subjects of this system manager's guide.

CHAPTER 2

COMPONENTS OF THE VAX/VMS OPERATING SYSTEM

This chapter is an introduction to the components, or files, that make up the VAX/VMS operating system. These components are, in effect, the environment in which the system manager works.

Among the first and most important responsibilities of the system manager is getting the VAX/VMS system up and running. At a minimum, this means the bootstrap loading of the operating system that is distributed by DIGITAL.

Often, in addition to bootstrapping the VAX/VMS operating system, getting a system up and running means both customizing some of the parameters of the operating system and incorporating optional software into the system. This optional software, which runs under control of the VAX/VMS operating system, includes VAX-11 FORTRAN IV-PLUS, PDP-11 BASIC-PLUS-2/VAX, PDP-11 COBOL-74/VAX, and the VAX/RSX-11 Development Package.

For an explanation of the steps that the system manager may have to take in getting a VAX/VMS system up and running, see the VAX-11 Software Installation Guide.

The components of the VAX/VMS operating system are cataloged in nine directories on the system distribution medium. The names of these directories and brief descriptions of their contents follow.

1. [SYSLIB], or [1,1]
   This directory contains various macro and object libraries.

2. [SYSMSG], or [1,2]
   This directory contains system message files.

3. [SYSMGR]
   This directory contains files used in managing the operating system.

4. [SYSHLP]
   This directory contains text files for the HELP utility.

5. [SYSERR]
   This is the directory for the error log file (ERRLOG.SYS).

6. [SYSTEST]
   This directory contains files used in testing the functions of the operating system.

7. [SYSMAINT]
   This directory contains system diagnostic programs.

8. [SYSUPD]
   This directory contains files used in applying system updates.

9. [SYSEXE], or [1,4]
   This directory contains the executable images of most of the functions of the operating system.

For a complete list of the files contained on the distribution medium, see the VAX-11 Software Installation Guide.

PART II

SETTING UP AND USING A SYSTEM OF ACCOUNTS

Part II of the VAX/VMS System Manager's Guide deals with how a system manager carries out one of the most important duties of system management -- the management of users' accounts.

Some of the main reasons for setting up a meaningful system of users' accounts are summarized as follows:

1. To denote the users of the system.

2. To define important relationships among the users of the system. These relationships are the basis of a system of file protection, of interprocess communication, and of a system of accounting.

3. To grant some users the privileges to perform certain sensitive system functions and thus to restrict some users from performing certain sensitive system functions.

4. To set limits on the use of reusable system resources.

5. To give users priorities in using the system.

Chapters 3 through 7, which develop these principal themes of system management, are as follows:

- Chapter 3: The User Authorization File
- Chapter 4: Groups
- Chapter 5: Limits, Priority, and Privilege
- Chapter 6: Accounting for the Use of System Resources
- Chapter 7: Using the User Authorization Program

CHAPTER 3

THE USER AUTHORIZATION FILE

If setting up a system of users' accounts is one of the system manager's most important duties in controlling the performance of the VAX/VMS system, then the user authorization file (UAF) is one of the most important data structures with which the system manager must be concerned.

The UAF contains one record for each user of the system. Each user's record contains information about the user; in effect, it defines the user to the system.

Besides the users' records, the UAF also contains a default value record and a system management record. How the records of the UAF are created, deleted, modified, and examined is the subject of Chapter 7, which describes how to use the User Authorization Program (AUTHORIZE).

Why is the UAF so important in controlling the performance of the VAX/VMS system? Simply stated, the reason is that each process on the VAX/VMS system is associated with a user. Each user is allotted certain system resources and is given a certain priority and certain privileges. All the attributes of a user (from identity to privilege) are specified in the user's record in the UAF.

When a user logs in to the system, a process is created on behalf of that user. This process is the context in which images that the user requests are executed. For example, a user may want to edit, compile, and link a program. This is done within the context of the process created at the time of logging in. To edit, compile, and link a program, the process executes three images: the editor, a compiler, and the linker.

The process created for a user acquires certain attributes, or characteristics, of the user for whom it was created. These attributes are the same attributes that the system manager put into the user's record in the UAF. Thus, the user's limits on the use of system resources, the user's base priority, and the user's privileges to make use of restricted system functions are all passed on to the process created for the user.

When the process originally created for a user creates subprocesses that in turn create other subprocesses, some of the resources allotted to the original process are shared among all the related processes, and some of the resources of the original process are passed on undiminished to the subprocesses. Attributes of the original process, including privileges and priority, may simply be passed on unaltered or may be passed on with changes.

This tree-like structure of a process branching into related subprocesses, all accountable to one user, is called a job. The result of associating users and jobs in this way is that the

attributes of jobs on the VAX/VMS system are rigidly predetermined by the decisions that the system manager makes in creating users' account records in the UAF. Hence, the contents of the UAF govern the utilization of system resources and thus the performance of the system.

In brief, each user's record in the UAF contains the following types of information about the user:

1. User's identification
    a. User name
    b. Password
    c. User identification code (UIC)
    d. Default account name
2. User's default directory name and default device name
3. User's default command interpreter name
4. User's allotment of system resources
5. User's privileges
6. User's base priority

For details on how the UIC is used to define groups of users, see Chapter 4; for details on limits, priority, and privileges, see Chapter 5.

The tables and figures that follow provide more information about the users' records of the UAF and about the default value record and the system management record of the UAF. Table 3-1 contains a concise description of each field of the UAF record. Note that all UAF records have the same form.

Table 3-2 lists both the default values that are usually assigned to the fields of the users' account records of the UAF and the minimum values that can be assigned to these fields. Chapter 7 explains the mechanics of assigning values to users' records.

Table 3-3 lists the initial values of the fields of the default value record of the UAF. Chapter 7 explains how to create and use this record.

Table 3-4 lists the initial values of the fields of the system management account record of the UAF. Chapter 7 explains how to create this record.

Figures 3-1, 3-2, and 3-3 illustrate the way these records are displayed to the system manager by the AUTHORIZE utility program. Chapter 7 fully explains how the SHOW command of the AUTHORIZE program displays UAF records.

Table 3-1
Fields of the UAF Record

Field	Description
Account name (ACCOUNT)	Default account name for user (for example, a billing number)
Buffered I/O byte count limit (BYTLM)	Limit on the total number of bytes that can be specified for transfer in outstanding buffered I/O operations
Buffered I/O count limit (BIOLM)	Limit on the number of buffered I/O operations (for example, terminal I/O operations) that can be outstanding at one time
CPU time quota	Reserved for maximum amount of CPU time a user can accumulate
Default AST queue limit (ASTLM)	Limit on the total number of asynchronous system trap operations and scheduled wake-up requests that can be outstanding at one time
Default command interpreter (CLI)	Name of the default command interpreter; DCL is standard
Default device name (DEVICE)	Default device name string; the device must be a random-access mass-storage device
Default directory name (DIRECTORY)	Default directory name
Default Files-11 file protection	Reserved for default Files-11 file protection for files created by the user
Default paging file limit (PGFLQUOTA)	Limit on the number of pages that the user's process can use in the system paging file (expressed in units of 256 pages)
Default priority of user's processes (PRIO)	Default base priority
Default working set size (W$DEFAULT)	Default working set size for the user's process; that is, the initial working set size for the process
Direct I/O count limit (DIOLM)	Limit on the number of direct I/O operations that can be outstanding at one time

(continued on next page)

Table 3-1 (Cont.)
Fields of the UAF Record

Field	Description
Log-in command file (LGICMD)	Name of a user's log-in command file; if this field is left blank, the user's log-in command file is by default LOGIN.COM
Log-in flags (FLAGS)	Log-in flags DISCTLY and DEFCLI; the flag DISCTLY, if set, disables CTRL/Y interrupts for a user; the flag DEFCLI, if set, restricts a user to using the default command interpreter (such a user cannot use the /CLI qualifier when logging in to the system)
Open file limit (FILLM)	Maximum number of files that the user's process can have open at one time; this limit includes the number of network logical links that can be active at the same time
Owner's name (OWNER)	ASCII string that may, for example, be useful for billing purposes
Password (PASSWORD)	Hashed password; a user must enter both a password and a user's name to gain access to the system
Privilege vector (PRIVILEGES)	Privileges to be allowed the user's process
Subprocess creation limit (PRCLM)	Maximum number of subprocesses that the user's process can have in existence at one time
Timer queue entry limit (TQELM)	Maximum number of timer entries that the user's process can have in the timer queue
User identification code (UIC)	User identification code (UIC), which consists of a group number and a member number; the UIC is the basis of the system's data protection scheme
User's name (USERNAME)	Unique name that the user types (along with a password) in logging in to the system
Working set size limit (WSQUOTA)	The maximum number of pages that the user's process can have in its working set

Table 3-2
User's Account Record of UAF

Field	Default Value	Minimum Value
Account name (ACCOUNT)	Blank	--
Buffered I/O byte count limit (BYTLM)	4096	1024
Buffered I/O count limit (BIOLM)	6	2
CPU time quota	Reserved	Reserved
Default AST queue limit (ASTLM)	10	2
Default command interpreter (CLI)	DCL	--
Default device name (DEVICE)	Blank	--
Default directory name (DIRECTORY)	[USER]	--
Default Files-11 file protection	Reserved	--
Default paging file limit (PGFLQUOTA)	40	1
Default priority of user's processes (PRIO)	4	1
Default working set size (WSDEFAULT)	150	50
Direct I/O count limit (DIOLM)	6	2
Log-in command file (LGICMD)	Blank (LOGIN.COM)	--

(continued on next page)

Table 3-2 (Cont.)
User's Account Record of UAF

Field	Default Value	Minimum Value
Log-in flags (FLAGS)	None set	--
Open file limit (FILLM)	20	2
Owner's name (OWNER)	Blank	--
Password (PASSWORD)	USER	--
Privilege vector (PRIVILEGES)	For non-privileged users: GROUP PRMMBX TMPMBX	--
Subprocess creation limit (PRCLM)	8	0
Timer queue entry limit (TQELM)	10	0
User identification code (UIC)	[200,200]	--
User's name (USERNAME)	None; this is the name given to the account by the system manager	--
Working set size limit (WSQUOTA)	200	50

```
USERNAME: BENNETT          OWNER: S BENNETT
ACCOUNT: 32V               UIC: [122,020]
DIRECTORY: [BENNETT]       DEVICE: DB1:
LOGIN COMMAND FILE:        LOGIN FLAGS:
CLI: DCL          PRCLM:         8 PRIO:           4
ASTLM:       10   BIOLM:         6 BYTLM:       4096
DIOLM:        6   FILLM:        20 TQELM:         10
WSDEFAULT:  150   WSQUOTA:     200 PGFLQUOTA:     40
PRIVILEGES:
   GROUP PRMMBX TMPMBX
```

Figure 3-1 Typical User's Account Record of UAF as Displayed by AUTHORIZE program

Table 3-3
Initial Default Value Record of UAF

Field	Initial Value
Account name (ACCOUNT)	Blank
Buffered I/O byte count limit (BYTLM)	4096
Buffered I/O count limit (BIOLM)	6
CPU time quota	Reserved
Default AST queue limit (ASTLM)	10

(continued on next page)

Table 3-3 (Cont.)
Initial Default Value Record of UAF

Field	Initial Value
Default command interpreter (CLI)	DCL
Default device name (DEVICE)	Blank
Default directory name (DIRECTORY)	[USER]
Default Files-11 file protection	Reserved
Default paging file limit (PGFLQUOTA)	40
Default priority of user's processes (PRIO)	4
Default working set size (WSDEFAULT)	150
Direct I/O count limit (DIOLM)	6
Log-in command file (LGICMD)	Blank (LOGIN.COM)
Log-in flags (FLAGS)	None set
Open file limit (FILLM)	20
Owner's name (OWNER)	Blank
Password (PASSWORD)	USER
Privilege vector (PRIVILEGES)	GROUP PRMMBX TMPMBX
Subprocess creation limit (PRCLM)	8

(continued on next page)

Table 3-3 (Cont.)
Initial Default Value Record of UAF

Field	Initial Value
Timer queue entry limit (TQELM)	10
User identification code (UIC)	[200,200]
User's name (USERNAME)	DEFAULT
Working set size limit (WSQUOTA)	200

```
USERNAME: DEFAULT          OWNER:
ACCOUNT:                   UIC: [200,200]
DIRECTORY: [USER]          DEVICE:
LOGIN COMMAND FILE:        LOGIN FLAGS:
CLI: DCL          PRCLM:         8 PRIO:          4
ASTLM:       10 BIOLM:          6 BYTLM:       4096
DIOLM:        6 FILLM:         20 TQELM:         10
WSDEFAULT:  150 WSQUOTA:     200 PGFLQUOTA:     40
PRIVILEGES:
   GROUP PRMMBX TMPMBX
```

DEFCLI DISCTLY

Figure 3-2 Default Value Record of UAF
as Displayed by AUTHORIZE Program

Table 3-4
Initial System Management Account Record of UAF

Field	Initial Value
Account name (ACCOUNT)	Blank
Buffered I/O byte count limit (BYTLM)	20480
Buffered I/O count limit (BIOLM)	6
CPU time quota	Reserved
Default AST queue limit (ASTLM)	20
Default command interpreter (CLI)	DCL
Default device name (DEVICE)	Blank
Default directory name (DIRECTORY)	[SYSMGR]
Default Files-11 file protection	Reserved
Default paging file limit (PGFLQUOTA)	40
Default priority of user's processes (PRIO)	4
Default working set size (WSDEFAULT)	150
Direct I/O count limit (DIOLM)	12
Log-in command file (LGICMD)	Blank (LOGIN.COM)
Log-in flags (FLAGS)	None set

(continued on next page)

Table 3-4 (Cont.)
Initial System Management Account Record of UAF

Field	Initial Value
Open file limit (FILLM)	20
Owner's name (OWNER)	SYSTEM MANAGER
Password (PASSWORD)	MANAGER
Privilege vector (PRIVILEGES)	CMKRNL ALTPRI CMEXEC SETPRV SYSNAM TMPMBX GRPNAM WORLD ALLSPOOL OPER DETACH EXQUOTA DIAGNOSE NETMBX LOG_IO VOLPRO GROUP PHY_IO ACNT BUGCHK PRMCEB PRMGBL PRMMBX SYSGBL PSWAPM MOUNT
Subprocess creation limit (PRCLM)	10
Timer queue entry limit (TQELM)	20
User identification code (UIC)	[001,004]
User's name (USERNAME)	SYSTEM
Working set size limit (WSQUOTA)	1024

```
USERNAME: SYSTEM              OWNER: SYSTEM MANAGER
ACCOUNT:                      UIC: [001,004]
DIRECTORY: [SYSMGR]           DEVICE:
LOGIN COMMAND FILE:           LOGIN FLAGS:
CLI: DCL          PRCLM:          10 PRIO:           4
ASTLM:        20  BIOLM:           6 BYTLM:       20480
DIOLM:        12  FILLM:          20 TQELM:          20
WSDEFAULT:   150  WSQUOTA:     1024 PGFLQUOTA:      40
PRIVILEGES:
CMKRNL CMEXEC SYSNAM GRPNAM ALLSPOOL DETACH DIAGNOSE LOG_IO
GROUP ACNT PRMCEB PRMMBX PSWAPM ALTPRI SETPRV TMPMBX
WORLD OPER EXQUOTA NETMBX VOLPRO PHY_IO BUGCHK PRMGBL
SYSGBL MOUNT
```

Figure 3-3 System Management Account Record of UAF as Displayed by AUTHORIZE Program

CHAPTER 4

GROUPS

In setting up a system of users' accounts, the system manager makes many decisions. Of all these decisions, one in particular is very difficult to change once it has been made. This is the decision about setting up groups.

The issue of groups principally concerns the protection of data and devices and secondarily concerns interprocess communication and control. A group is a collection of users who (through the processes and jobs created on their behalf) normally have access to each other's files, to each other's file-structured volumes, to each other's mailboxes, to shared pages of memory, to common event flags, and to the group logical name table. In addition, the processes and jobs created on behalf of these users may be given special privileges (described in Chapter 5) to exercise control over each other.

In setting up a group, the system manager must aim toward two goals: 1) to facilitate sharing of data and cooperation between users and user processes and 2) to protect users from unauthorized access to their processes and data.

The importance of properly setting up groups of users should not be underestimated. As a system is used more and more and as users create more and more files and other protected data structures and devices, the relationships among the members of a group and their data structures, their devices, and their processes become more and more complex. Thus, with the passage of time, it becomes more and more difficult to go back and redefine the basic relationships among the users of the system and among the processes created on their behalf.

4.1 THE USER IDENTIFICATION CODE

Whether a user is a member of a particular group or not depends on the user identification code (UIC) that the system manager assigns to the user. As explained in Chapter 7, the system manager uses the User Authorization Program (AUTHORIZE) to store the UIC in the user's account record of the user authorization file (UAF).

For the system manager, the choice of UICs and their assignment to users involve the following two important questions:

1. Which users should be allowed to share data and file access rights and which should not?

2. Which processes should be allowed to cooperate and which should not?

The UIC is, therefore, not simply a means of identifying a user; rather, it is the basis of the data protection scheme of the VAX/VMS system and one of the factors (along with privilege) that govern the ways in which processes created for users can interact with one another.

A UIC is an expression that consists of two numbers, represented as [g,m], as follows:

1. A group number (represented by the letter g) that can have a value in the range of 0 through 377 octal

2. A member number (represented by the letter m) that can have a value in the range of 0 through 377 octal

For example, the UIC can have such values as [10,40], [143,10], and [200,200]. Users whose UICs have the same group numbers (for example, [320,101], [320,102], and [320,103]) belong to the same group. Users whose group numbers are between 0 and 10 octal, inclusive, are system users. One of these system users is the system manager, whose UIC is [1,4].

The user's account record for each authorized user of the system contains a UIC that is explicitly assigned by the system manager. Like other attributes of an authorized user, a user's UIC is passed on to subprocesses created on behalf of the user.

Similarly, every protected data structure that a user creates (or, more properly, that a user's process creates) has an owner's UIC -- normally, the same UIC as is in the user's account record. Devices owned by users also have the owner's UIC.

4.2 PROTECTION OF DATA STRUCTURES AND DEVICES

The relationship between a user and a protected data structure or device is defined by the relationship between the UIC of the user and the UIC of the owner of the data structure or device. Hence, the UIC that the system manager assigns to a user is, in effect, the user's key to data structures and devices that are protected by UICs.

These data structures and devices are files and file-structured volumes, mailboxes, shared pages in memory (global sections), common event flags, and the group logical name table. A user's access to them is governed by two factors, which are described in the following paragraphs: the type of user (as defined by UIC) and the type of access desired.

For purposes of data protection, four different categories of user are defined with respect to protected data structures or to owned devices. These users (defined by UIC) are as follows:

1. Owner -- a user whose UIC is identical with the UIC of the owner of the data structure or device. For example, the owner of a file is ordinarily the creator of the file.

2. Group -- users of the system who are members of the same group as the owner of the data structure or device. Thus, the group number of a member of a group is identical with the group number of the data structure or device.

3. System -- users of the system whose group numbers fall between 0 and 10 octal, inclusive. Sometimes, these system users may also fall into the owner category for files they create and into the group category for files created by members of their groups.

4. World -- all other users. This category includes users who are not owners of the data structure or device in question, are not members of the same group as the owner of the data or device, and are not system users. Thus, these are users whose group numbers are different from the group number of the data structure or device and are not system group numbers.

Figure 4-1 illustrates the relationships of these four types of user to a file that has an owner's UIC of [100,100]. Of special interest are the owner of the file and members of the same group as the owner of the file. The owner of the file is any user who has the same UIC as the file: [100,100]. Members of the same group as the owner are all other users who have a group number of 100.

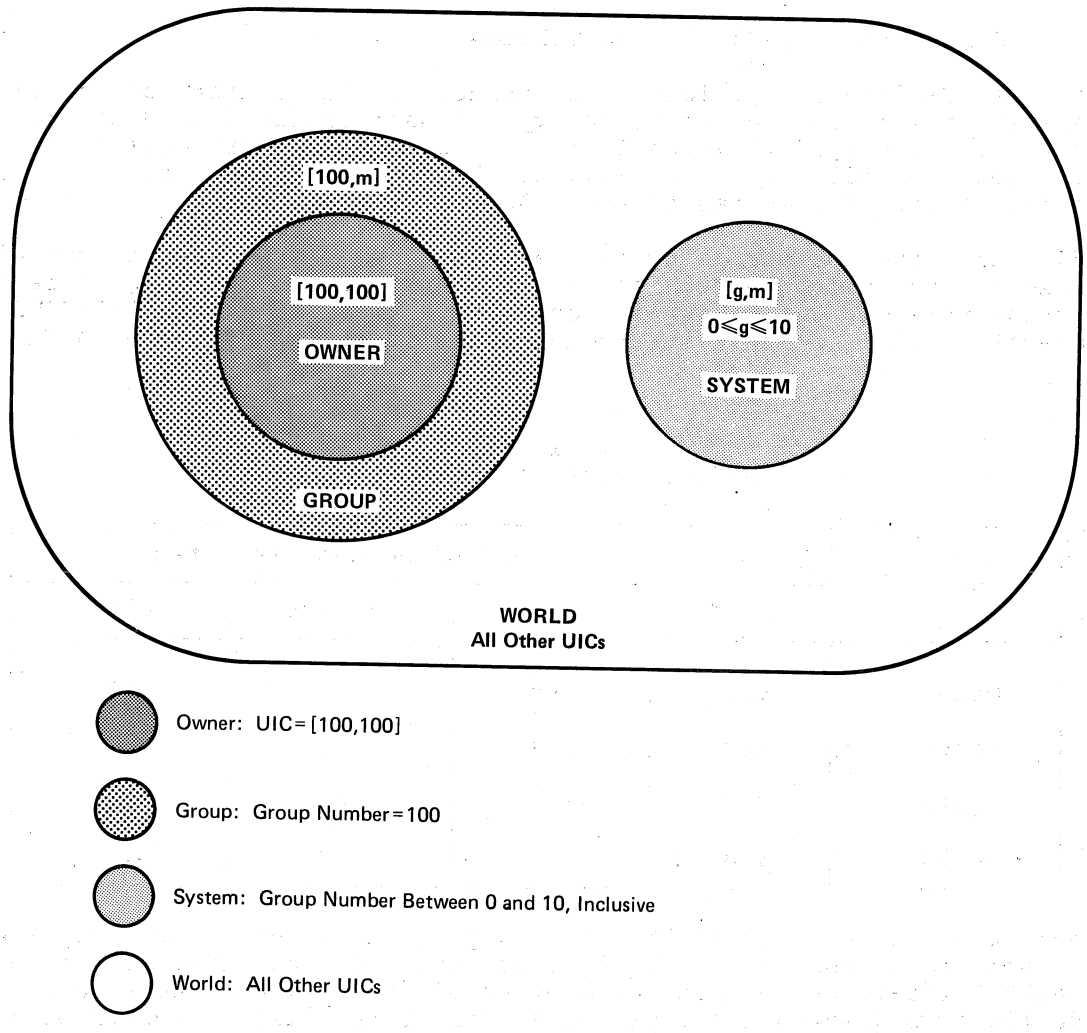


Figure 4-1 Classification of Users with Respect to a File with a UIC of [100,100]

Users are permitted four types of access to protected data structures and devices, as follows:

1. Read access (abbreviated R)
2. Write access (abbreviated W)
3. Execute access (abbreviated E)
4. Delete access (abbreviated D)

In general, any type of user can be permitted or denied any type of access to a data structure or device. There are, however, exceptions to this rule, for not all types of access apply to all data structures and devices. For example, execute access applies only to files that contain executable program images. Nonetheless, the UIC is still the basis for protecting the different data structures and devices. The following sections describe the protection provided for each of the data structures and devices.

4.2.1 Files and File-Structured Volumes

Either explicitly or implicitly (by default), the creator of a file specifies the kind of access each type of user (defined by UIC) has to the file. This specification is in the form of a protection code. This code controls which, if any, of the four kinds of access each of the four types of user has to the file.

Ordinarily, the operating system assigns to each user a default Files-11 protection code, or mask. Thus, unless a user chooses to override this default protection code (by use of the SET PROTECTION command), files created by the user are protected as shown in Table 4-1.

Table 4-1
Default Files-11 Protection

Type of User	Type of Access Permitted			
	Read	Write	Execute	Delete
Owner	Yes	Yes	Yes	Yes
Group	Yes	Yes	Yes	Yes
System	Yes	Yes	Yes	Yes
World	Yes	No	Yes	No

Figure 4-2 illustrates how the UIC and protection code work together to control access to a file that has a UIC of [100,100] and has default Files-11 protection. The owner of the file (whose UIC is [100,100]) is granted all types of access to the file; members of the owner's group (g=100) have all types of access to the file; system users (g=7 and g=1) have all types of access to the file; and world users (represented by a user with a UIC of [200,10]) have read access and execute access to the file.

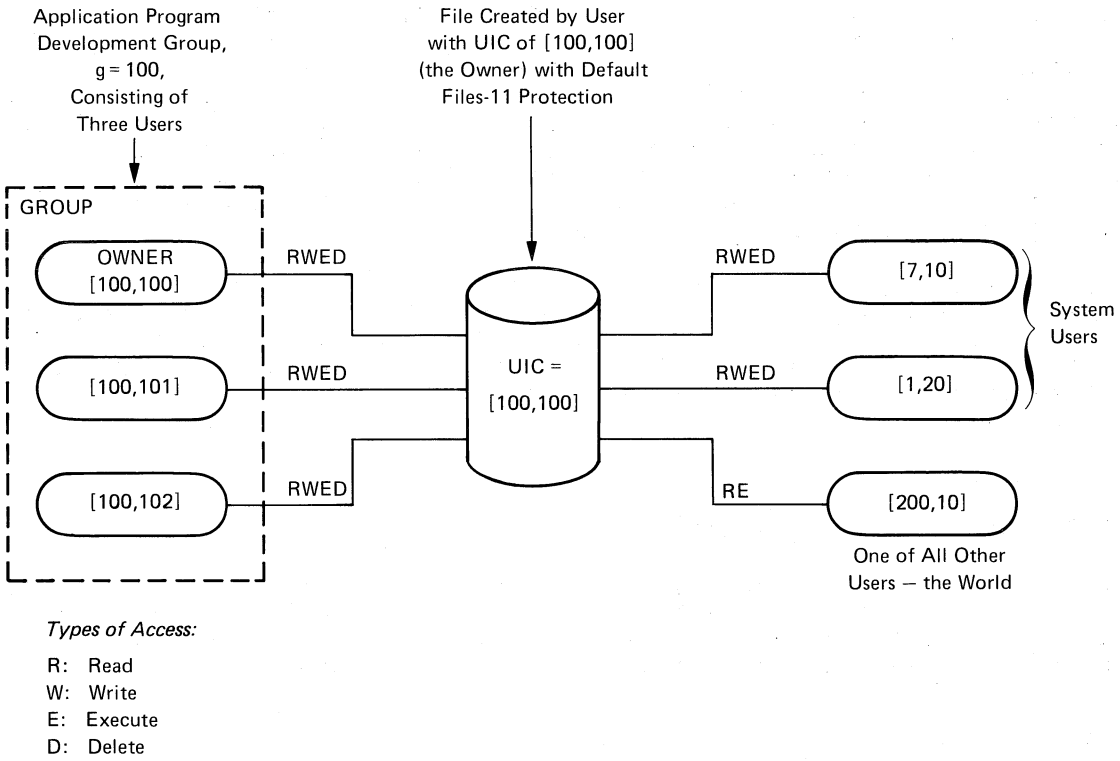


Figure 4-2 Controlling Access to a File

Sometimes, in creating files, users choose to override the default protection and thus to protect files on an individual basis. For example, a user can freely share many files with a group but still deny group access to one or more files that contain particularly sensitive data.

The scheme for protecting file-structured volumes is similar to the one for protecting files except that execute access to a volume gives a user the right to create files on the volume.

Users who have been assigned the VOLPRO privilege can override the protection of file-structured volumes. In addition, the system manager can partly overcome the system of file and volume protection based on UIC by granting special users either or both of two privileges: LOG_IO and PHY_IO. Users who have one or both of these privileges have the same access rights to files and to file-structured volumes as system users have.

4.2.2 **Mailboxes**

Mailboxes (both temporary and permanent) are protected by a code, or mask, that is similar to the code used in protecting files. As with files, four types of user (defined by UIC) can gain access to a mailbox: owner, group, system, and world. However, only two types of access -- read access and write access -- are meaningful to users of a mailbox. Thus, when creating a mailbox, the user can specify read access and write access to the mailbox separately for each type of user.

Because temporary mailboxes are customarily used for interprocess communication between cooperating processes that have the same group number, they are of special interest here. The logical names of these mailboxes are entered in the group logical name table. Temporary mailboxes thus have two layers of protection. First, easy access to the logical names of temporary mailboxes is granted to users who have the same group number as the creator of the mailbox; other users have no such easy access. Second, by use of the protection mask, the creator of the temporary mailbox grants additional security to the mailbox. As a rule, users who are not in the same group as the creator are totally excluded from using the mailbox.

Furthermore, the creator of a temporary mailbox can discriminate between owners and other group members in granting read access and write access to a temporary mailbox. For example, owners may be allowed only read access or write access to the mailbox, but other members of the group may be allowed both read access and write access to the mailbox.

4.2.3 Shared Pages in Memory (Global Sections)

All shared pages in memory (global sections) are protected by a code, or mask, that is the same as the one used to protect other files. Thus, the creator of a global section can specify read access, write access, execute access, and delete access separately for each of the four types of user.

Of special interest are group global sections; such a section can be shared only by processes that have the same group number as the global section. This group number is that of the creator of the global section. Thus, group global sections have two layers of protection. First, if a process is to map to a global section, the group numbers of the process and of the global section must be identical. Second, by use of the protection mask, the creator of a group global section grants additional security to that global section.

4.2.4 Common Event Flags

Common event flags, which are used in establishing communication and synchronization among cooperating processes in the same group, are protected by UIC.

When a common event flag cluster is created, it takes on the UIC of the process that requested its creation and, thus, the UIC of the user whose process created the cluster. To control access to a common event flag cluster, a protection indicator is used with the UIC. If the value of this indicator is 0 (the default), both read access and write access to the cluster are extended to all processes and users in the owner's group. If the value of this indicator is 1, both read access and write access to the cluster are restricted to owners of the cluster and to subprocesses that have been created on behalf of these owners.

4.2.5 Group Logical Name Table

Entries in the group logical name table are protected by group number.

When a logical name and its equivalence name are entered into the group logical name table, the group number of the process that created the logical name is associated with the entry. Access to an entry in the group logical name table is thus restricted to users and processes that have the same group number as the entry.

4.3 INTERACTION AMONG PROCESSES

The ability of one process to control or examine another process is regulated by UIC and user privilege, which work hand in hand. The system manager assigns both UIC and privilege.

For one process to affect another process, the first process may need to have a user privilege. The amount of user privilege needed depends on the relationship of the UIC of the controlling or examining process to the UIC of the controlled or examined process. There are three levels of privilege, as described below.

1. No privilege -- a process needs no special privilege to control or examine a subprocess that it created. Here, the UIC of the controlling or examining process and the UIC of the controlled or examined process are the same.

2. GROUP privilege -- a process needs this privilege to control or examine a detached process that is a member of its own group. Here, the group number of the controlling or examining process and the group number of the controlled or examined process are the same.

3. WORLD privilege -- a process needs this privilege to control or examine a process outside its own group. Here, the group number of the controlling or examining process and the group number of the controlled or examined process are different.

Users whose processes must exercise control over each other should, ideally, be assigned to the same group and should have the group privilege. With this privilege, a process created on behalf of a group member can control or examine processes created for other members of the same group. This is done by executing the following process control system services: the Suspend Process system service, the Resume Process system service, the Delete Process system service, the Set Priority system service, the Wake system service, the Schedule Wakeup system service, the Cancel Wakeup system service, the Force Exit system service, and the Get Job/Process Information system service.

Forming a group and assigning group privilege to members of the group thus serve the need of interprocess control and at the same time restrict interaction among processes to the group itself.

5.L.

CHAPTER 5

LIMITS, PRIORITY, AND PRIVILEGE

This chapter contains detailed descriptions of the attributes that the system manager assigns to a user when creating the user's account record in the user authorization file (UAF). These attributes are:

- The limits on the use of reusable system resources
- The base priority used in scheduling the process that the system creates for the user
- The privileges of using restricted and sensitive system functions

5.1 LIMITS

Limits are set on system resources that can be reused, for example, the amount of memory that a process can have in use for queued I/O requests. Each user of the system has limits on the use of system resources. The system manager sets up these limits when the user is defined to the system. Most limits restrict the use of system dynamic memory.

Limits can be either deductible or nondeductible. These terms refer to the way in which a process shares its allotment of a resource with the subprocesses that it creates. If the limit on the use of a resource is deductible, a subprocess may be given only a portion of the total allotment of the resource of the process that created it. The portion given to the subprocess is deducted from the allotment given to the creating process. If, on the other hand, the limit is nondeductible, the subprocess receives an allotment of the resource equal to the total allotment of the process that created it, and there is no deduction from the allotment of the creating process.

In creating a user's account record in the UAF, the system manager assigns values to the following ten limits.

1. Default AST queue limit (ASTLM)
2. Buffered I/O count limit (BIOLM)
3. Buffered I/O byte limit (BYTLM)
4. Direct I/O count limit (DIOLM)
5. Open file limit (FILLM)
6. Default paging file limit (PGFLQUOTA)

7. Subprocess creation limit (PRCLM)
8. Timer queue entry limit (TQELM)
9. Default working set size (WSDEFAULT)
10. Working set size limit (WSQUOTA)

These limits are described in the following sections. Usually, the system manager simply assigns the default values of these limits. These defaults can, however, be overridden, as described in Chapter 7.

5.1.1 Default AST Queue Limit (ASTLM)

The default AST queue limit (ASTLM) limits the sum of the following:

- The number of asynchronous system trap (AST) operations that a user's process can have outstanding at one time
- The number of scheduled wake-up requests that a user's process can have outstanding at one time

This limit, therefore, affects not only all system services that accept an AST address as an argument but also the Schedule Wakeup system service.

ASTLM is a nondeductible limit that has the following minimum and default values:

- Minimum value: 2
- Default value: 10

5.1.2 Buffered I/O Count Limit (BIOLM)

The buffered I/O count limit (BIOLM) limits the number of outstanding buffered I/O operations permitted to a user's process.

A buffered I/O operation is an I/O operation in which the data transfer takes place from an intermediate buffer in the system pool, not from a process-specified buffer. Thus, during a buffered I/O operation, the pages containing the process-specified buffer that is given as an argument to the Queue I/O Request system service need not be locked in memory.

BIOLM is a nondeductible limit that has the following minimum and default values:

- Minimum value: 2
- Default value: 6

5.1.3 Buffered I/O Byte Count Limit (BYTLM)

The buffered I/O byte count limit (BYTLM) limits the amount of buffer space that a user's process can use.

This buffer space is used for buffered I/O operations and for the creation of temporary mailboxes.

BYTLM is a deductible limit that has the following minimum and default values:

- Minimum value: 1024 bytes
- Default value: 4096 bytes

5.1.4 Direct I/O Count Limit (DIOLM)

The direct I/O count limit (DIOLM) limits the number of outstanding direct I/O operations permitted to a user's process.

A direct I/O operation is an I/O operation in which the data transfer takes place directly from a process-specified buffer. The pages containing this buffer are locked in memory by the operating system during the direct I/O operation.

DIOLM is a nondeductible limit that has the following minimum and default values:

- Minimum value: 2
- Default value: 6

5.1.5 Open File Limit (FILLM)

The open file limit (FILLM) limits the number of files that a user's process can have open at one time. This limit includes the number of network logical links that can be active at the same time.

FILLM is a deductible limit that has the following minimum and default values:

- Minimum value: 2
- Default value: 20

5.1.6 Default Paging File Limit (PGFLQUOTA)

The default paging file limit (PGFLQUOTA) limits the number of pages that the user's process can use in the system paging file.

The paging file or files of a process provide backing storage (on disk) for pages that the process has altered since they were last read into memory either from an image file or from a paging file.

PGFLQUOTA is a deductible limit that has the following minimum and default values:

- Minimum value: 1 (256 pages)
- Default value: 40 (10000 pages allotted in blocks of 256 pages)

5.1.7 Subprocess Creation Limit (PRCLM)

The subprocess creation limit (PRCLM) limits the number of subprocesses that can be created for the user.

The process that is created when a user logs in to the system can in turn create subprocesses. These subprocesses are all accountable to the user and share the resources allotted to the initial process.

PRCLM is a deductible limit that has the following minimum and default values:

- Minimum value: 0
- Default value: 8

5.1.8 Timer Queue Entry Limit (TQELM)

The timer queue entry limit (TQELM) limits the sum of the following:

- The number of entries that a user's process can have in the timer queue
- The number of temporary common event flag clusters that a user's process can have

This limit does not govern the creation of permanent event flag clusters.

Time queue entries are used in time-dependent scheduling; common event flags are used in synchronizing activities among groups of cooperating processes.

TQELM is a deductible limit that has the following minimum and default values:

- Minimum value: 0
- Default value: 10

5.1.9 Default Working Set Size (WSDEFAULT)

The default working set size (WSDEFAULT) sets the initial working set size for a user's process.

In other words, the default working set size is used in establishing the working set size of the process before an image is activated.

WSDEFAULT is a nondeductible limit that has the following minimum, default, and maximum values:

- Minimum value: 50 pages
- Default value: 150 pages
- Maximum value: The value of the working set size limit (WSQUOTA; see Section 5.1.10) for the created process -- by default, 200 pages

Changes in default working set size may have a noticeable effect on system performance.

5.1.10 Working Set Size Limit (WSQUOTA)

The working set size limit (WSQUOTA) limits the size to which the working set of a user's process can be expanded. This enlargement of the working set (the initial size of which is determined by the limit WSDEFAULT; see Section 5.1.9) is accomplished by use of the Adjust Working Set Limit system service or by the SET WORKING_SET command.

WSQUOTA is a nondeductible limit that has the following minimum and default values:

- Minimum value: 50 pages
- Default value: 200 pages

Changes in working set size limit have a noticeable effect on system performance.

5.2 PRIORITY

A user's priority is the base priority used in scheduling the process that the system creates for the user. There are 32 levels of software priority in the VAX/VMS system, 0 through 31. The highest priority is 31; the lowest is 0. The range of priorities for normal processes is 1 through 15; the range for time-critical processes is 16 through 31.

Processes with time-critical priorities are scheduled strictly according to base priority; in other words, the executable time-critical process with the highest base priority is executed first. Processes with normal priorities are scheduled according to a slightly different principle to promote overlapping of computation and I/O activities.

In the user's account record of the UAF, the default value of a user's priority is 4; for practical purposes, the minimum value is 1.

5.3 PRIVILEGES

Many system services are protected by privileges, which restrict the use of the system services to certain users. These restrictions protect the integrity of performance of the operating system and thus the integrity of service provided users. The system manager grants privileges to each user on the basis of two factors: 1) whether or not the user has the skill and experience to use the privilege without disrupting the system and 2) whether or not the user has a legitimate need for the privilege.

In terms of skill and experience, there are perhaps five types of user, as follows:

- The beginning user, who lacks skill and experience and is thus generally granted no privileges

- The average time-sharing and batch-program user, who is generally assigned the default privileges as defined in the default value record of the UAF
- The system operator, who is generally assigned both the default privileges and the privileges GRPNAM, PRMCEB, OPER, LOG_IO, PHY_IO, VOLPRO, DIAGNOSE, and SYSNAM
- The system manager, who is generally assigned all privileges in the initial system management account record of the UAF
- The system programmer, who is generally assigned all privileges needed

In terms of need, a user cannot ordinarily execute an image that requires privileges that the user does not have. For example, if a user has a legitimate need to execute a program that attempts to issue logical I/O requests, that user must be given (barring serious lack of prerequisite skill and experience) the LOG_IO privilege.

There is, however, a notable exception to this rule that a user cannot execute an image that requires privileges the user does not have. When the user executes a special kind of image, called a known image, the privileges of the user may if necessary be temporarily increased so that, in effect, the user's privileges match the privileges of the image. The user can thus execute an image requiring more privilege than the user is normally allowed. The system manager is responsible for installing known images; see Chapter 10.

A user's privileges are recorded in the user's account record of the UAF in a 64-bit privilege vector. When a user logs in to the system, the user's privilege vector is stored in the header of the user's process. Thus, the user's privileges are passed on to the process created for the user.

Table 5-1 lists the privileges that have been defined and gives brief, general definitions of them. The following sections describe each privilege in detail.

Table 5-1
Definition of the Privilege Vector

Privilege	Meaning
ACNT	To suppress accounting messages
ALLSPOOL	To allocate spooled devices
ALTPRI	To alter the priority of a process; a process with this privilege can raise its own priority or set the priority of another process higher than its own priority
BUGCHK	To use the VAX/VMS Bugcheck Facility
CMEXEC	To execute Change Mode to Executive system service
CMKRNL	To execute Change Mode to Kernel system service

(continued on next page)

Table 5-1 (Cont.)
Definition of the Privilege Vector

Privilege	Meaning
DETACH	To create detached processes
DIAGNOSE	To issue diagnostic functions
GROUP	To issue process control system services within a group
GRPNAM	To insert group logical names in the name table
LOG_IO	To issue logical I/O requests
MOUNT	To execute the mount volume QIO
NETMBX	To perform any function related to DECnet
OPER	To execute operator functions
PHY_IO	To issue physical I/O requests
PRMCEB	To create or to delete permanent common event flag clusters
PRMGBL	To create permanent global sections
PRMMBX	To create permanent mailboxes
PSWAPM	To change process swap mode
SETPRV	To set any privilege bits; a process with this privilege can create processes whose privileges are greater than its own
SYSGBL	To create system global sections
SYSNAM	To insert system logical names in the name table
TMPMBX	To create temporary mailboxes
VOLPRO	To override volume protection
WORLD	To issue process control system services to the world

5.3.1 The ACNT Privilege

This is the privilege of suppressing accounting messages for a created process.

Only a user who has the ACNT privilege can create subprocesses or detached processes in which accounting is disabled. Thus, only such a privileged user can issue the RUN command with the /NOACCOUNTING qualifier.

5.3.2 The ALLSPOOL Privilege

This is the privilege of allocating spooled devices.

If the system manager grants a user the ALLSPOOL privilege, the user's process is allowed to allocate a spooled device by executing the Allocate Device system service, or the user is allowed to allocate a spooled device by using the ALLOCATE command.

The Allocate Device system service lets a process allocate, or reserve, a device for its exclusive use. A shareable mounted device cannot be allocated.

Ordinarily, the privilege of allocating a spooled device is granted only to symbionts.

The system manager should grant this privilege only to users who need to perform logical or physical I/O operations to a spooled device.

5.3.3 The ALTPRI Privilege

This is the privilege of setting or changing the base priority of a process.

If the system manager grants a user the ALTPRI privilege, the user's process is allowed to 1) increase its own base priority and 2) set the base priority of another process to a value higher than its own base priority.

The base priority is increased by executing the Set Priority system service or the SET PROCESS/PRIORITY command. As a rule, this system service lets a process set its own base priority or the base priority of another process. A process can set the priority of another process, however, only if the other process is its subprocess or if the process setting the priority has process control privilege (GROUP or WORLD) over the process whose priority is being set. For a process to raise its own base priority or to set the base priority of another process to a value higher than its own, the ALTPRI privilege is essential.

The base priority of a process is one of the factors that the scheduler uses to determine the order in which processes are to be run.

Note also that with the same privilege a process can create a process with a base priority higher than its own. Such a process is created by using an optional argument to the Create Process system service or to the RUN command.

The system manager should not grant this privilege widely; if many users have the unrestricted ability to set base priorities, the fair and orderly scheduling of processes for execution could easily be disrupted.

5.3.4 The BUGCHK Privilege

This is the privilege of using the VAX/VMS Bugcheck Facility. The use of this facility is restricted to system software supplied by DIGITAL.

5.3.5 The CMEXEC Privilege

This is the privilege of executing the Change Mode to Executive system service.

If the system manager grants a user the CMEXEC privilege, the user's process is allowed to execute the Change Mode to Executive system service.

This system service lets a process change its access mode to executive, execute a specified routine, and then return to the access mode in effect before the system service was called. While in executive mode, the process is allowed to execute the Change Mode to Kernel system service.

The system manager should grant this privilege only to users who need to gain access to protected and sensitive data structures and internal functions of the operating system.

If many users have unrestricted access to sensitive data structures and functions, the operating system and service to other users could easily be disrupted. Such disruptions could include crashing of the system, destruction of the data base, and exposure of confidential information to unauthorized persons.

5.3.6 The CMKRNL Privilege

This is the privilege of executing the Change Mode to Kernel system service.

If the system manager grants a user the CMKRNL privilege, the user's process is allowed to execute the Change Mode to Kernel system service.

This system service lets a process change its access mode to kernel, execute a specified routine, and then return to the access mode in effect before the system service was called.

The system manager should grant this privilege only to users who need to execute privileged instructions or who need to gain access to the most protected and sensitive data structures and functions of the operating system.

If many users have unrestricted use of privileged instructions and unrestricted access to sensitive data structures and functions, the operating system and service to other users could easily be disrupted. Such disruptions could include crashing of the system, destruction of the data base, and exposure of confidential information to unauthorized persons.

In a sense, the CMKRNL privilege is the "most privileged" privilege.

5.3.7 The DETACH Privilege

This is the privilege of creating detached processes.

If the system manager grants a user the DETACH privilege, the user's process is allowed to create detached processes by executing the Create Process system service. Detached processes remain in existence even after the user who created them has logged off the system.

An example of a detached process is the process created by the system for a user, when the user logs in to the system.

There is no restriction on the UIC that can be specified for a detached process. Thus, there is no restriction on the files and directories to which a detached process can gain access.

5.3.8 The DIAGNOSE Privilege

This is the privilege of issuing diagnostic functions.

If the system manager grants a user the DIAGNOSE privilege, that user is allowed to run on-line diagnostic programs and to intercept and copy all messages that are written to the error log file.

5.3.9 The GROUP Privilege

This is the privilege that lets a process control other processes within its own group and examine the parameters of other processes within its own group.

If the system manager grants a user the GROUP privilege, the user's process is allowed to affect other processes in its own group by executing the following process control system services: the Suspend Process system service, the Resume Process system service, the Delete Process system service, the Set Priority system service, the Wake system service, the Schedule Wakeup system service, the Cancel Wakeup system service, and the Force Exit system service. The user's process is also allowed to examine other processes in its own group by executing the Get Job/Process Information system service.

The user who has this privilege can also execute the STOP (Image or Process) command for any process in the user's group.

The GROUP privilege is not needed for a process to exercise control over, or to examine, subprocesses that it created. The system manager should, however, grant this privilege to users who need to share data and whose processes need to cooperate.

5.3.10 The GRPNAM Privilege

This is the privilege of inserting logical names into the group logical name table and of deleting logical names from that table.

If the system manager grants a user the GRPNAM privilege, the user's process is allowed to insert pairs of names into the logical name table of the group to which the process belongs and to delete names from that table. This is done by the use of the following logical name system services: the Create Logical Name system service and the Delete Logical Name system service.

In addition, the privileged user can use the ASSIGN and DEFINE commands to add names to the group logical name table, and can use the DEASSIGN command to delete names from the table.

To use the /GROUP qualifier of the MOUNT command, the GRPNAM privilege is required.

This privilege should not be granted to all users of the system. There is no limit on the number of group logical names that can be created by a user who has the GRPNAM privilege. Thus, if many users have the unrestricted ability to create group logical names, system performance could be degraded by an excessive use of system dynamic memory.

5.3.11 The LOG_IO Privilege

This is the privilege of issuing logical I/O requests.

If the system manager grants a user the LOG_IO privilege, the user's process is allowed to execute the Queue I/O Request system service to perform logical level I/O operations. Moreover, users who have the LOG_IO privilege have the same access rights to files and to file-structured volumes as system users have.

Usually, users' I/O requests are handled indirectly by use of an I/O package such as the VAX-11 Record Management Services. However, to increase their control over I/O operations and to improve the efficiency of I/O operations, skilled users sometimes prefer to handle directly the interface between their process and a system I/O driver program. They can do this by executing the Queue I/O Request system service; in many instances, the operation called for is a logical level I/O operation.

The system manager should grant this privilege only to users who need it. If this privilege is given to many users who have no need for it, the operating system and service to other users could easily be disrupted. Such disruptions could include the destruction of information on the system device, the destruction of user data, and the exposure of confidential information to unauthorized persons.

5.3.12 The MOUNT Privilege

This is the privilege of executing the mount volume QIO function. The use of this function is restricted to system software supplied by DIGITAL.

5.3.13 The NETMBX Privilege

This is the privilege of performing functions related to a DECnet computer network. In order to perform any network operations, a user must have this privilege.

5.3.14 The OPER Privilege

This is the privilege of performing certain operator's functions.

If the system manager grants a user the OPER privilege, that user is allowed to use the Operator Communication Manager (OPCOM) process, as follows: to reply to users' requests, to broadcast messages to all terminals logged in, to designate terminals as operators' terminals and specify the types of messages to be displayed on these operators' terminals, and to initialize and control the log file of operators' messages. In addition, this privilege lets the user set devices

spooled, create and control both batch queues and print queues, and initialize and mount public volumes.

The system manager should grant this privilege only to special users -- the operators of the system. These are the users who respond to the requests of ordinary users, who tend to the needs of the system's peripheral devices (mounting reels of tape and changing printer forms), and who attend to all the other day-to-day chores of system operation.

5.3.15 The PHY_IO Privilege

This is the privilege of issuing physical I/O requests.

If the system manager grants a user the PHY_IO privilege, the user's process is allowed to execute the Queue I/O Request system service to perform physical level I/O operations. Moreover, users who have the PHY_IO privilege have the same access rights to files and to file-structured volumes as system users have.

Usually, users' I/O requests are handled indirectly by use of an I/O package such as VAX-11 Record Management Services. However, to increase their control over I/O operations and to improve the efficiency of I/O operations, skilled users sometimes prefer to handle directly the interface between their process and a system I/O driver program. They can do this by executing the Queue I/O Request system service; in many instances, the operation called for is a physical level I/O operation.

The system manager should grant the PHY_IO privilege only to users who need it; in fact, this privilege should be granted even more carefully than the LOG_IO privilege (see Section 5.3.11). If this privilege is given to many users who have no need for it, the operating system and service to other users could easily be disrupted. Such disruptions could include the destruction of information on the system device, the destruction of user data, and the exposure of confidential information to unauthorized persons.

5.3.16 The PRMCEB Privilege

This is the privilege of creating or deleting permanent common event flag clusters.

If the system manager grants a user the PRMCEB privilege, the user's process is allowed to create or delete a permanent common event flag cluster by executing the Associate Common Event Flag Cluster system service or the Delete Common Event Flag Cluster system service.

Common event flag clusters enable cooperating processes to communicate with each other and thus provide the means of synchronizing their execution.

This privilege should not be granted to all users of the system. There is no limit on the number of permanent common event flag clusters that can be created by a user who has the PRMCEB privilege, and a permanent cluster remains in the system even after the creating process has been terminated. Such a cluster continues to use up a portion of system dynamic memory. Thus, if many users have the unrestricted ability to create permanent common event flag clusters, system performance could be degraded by an excessive use of system dynamic memory.

5.3.17 The PRMGBL Privilege

This is the privilege of creating permanent global sections.

5.3.18 The PRMMBX Privilege

This is the privilege of creating or deleting permanent mailboxes.

If the system manager grants a user the PRMMBX privilege, the user's process is allowed to create or delete a permanent mailbox by executing the Create Mailbox and Assign Channel system service or the Delete Mailbox system service.

Mailboxes are buffers in virtual memory that are treated as if they were record-oriented I/O devices. A mailbox is used for general interprocess communication.

This is a privilege that should not be granted to all users of the system. Permanent mailboxes are not automatically deleted when the creating processes are deleted. Thus, these mailboxes continue to use up a portion of system dynamic memory.

5.3.19 The PSWAPM Privilege

This is the privilege of changing the process swap mode.

If the system manager grants a user the PSWAPM privilege, the user's process is allowed to control whether or not it can be swapped out of the balance set. Not only must a process have appropriate privilege to lock itself in the balance set (that is, to disable swapping); but, once a process has been locked in the balance set, it also needs the same privilege to unlock itself (that is, to enable swapping).

The Set Process Swap Mode system service controls this changing of the process swap mode.

Note also that with the same privilege a process can create a process that is locked in the balance set (process swap mode disabled). This is done by using an optional argument to the Create Process system service or by using a qualifier of the RUN command, when the RUN command is used to create a process.

The system manager should grant this privilege only to users who need to lock a process in memory for performance reasons. Typically, this will be a time-critical process.

If many users have the unrestricted ability to lock processes in the balance set, system performance could be degraded, as physical memory could be held unnecessarily.

5.3.20 The SETPRV Privilege

This is the privilege of endowing a process with greater privileges than those of the process that created it.

If the system manager grants a user the SETPRV privilege, the user's process is allowed to create processes whose privileges are greater than its own. This is done by executing the Create Process system

service, using an optional argument, or by issuing a RUN command to create a process.

The SETPRV privilege should be guarded closely, because it is an extremely powerful privilege. The process that has it can create subprocesses with the CMKRNL privilege, the "most privileged" privilege. The system is at the mercy of users with these privileges.

5.3.21 The SYSGBL Privilege

This is the privilege of creating system global sections.

5.3.22 The SYSNAM Privilege

This is the privilege of inserting logical names into the system logical name table and deleting logical names from that table.

If the system manager grants a user the SYSNAM privilege, the user's process is allowed to insert pairs of names into the system logical name table and to delete names from that table. This is done by using the following logical name system services: the Create Logical Name system service and the Delete Logical Name system service.

In addition, the privileged user can use the ASSIGN and DEFINE commands to add names to the system logical name table, and can use the DEASSIGN command to delete names from the table.

The system manager should grant this privilege only to the system operator or to system programmers who need to define system logical names (such as, for the user devices, for library directories, and for the system directory). For example, to mount a system volume, which entails defining a system logical name, the system operator must have the SYSNAM privilege.

5.3.23 The TMPMBX Privilege

This is the privilege of creating temporary mailboxes.

If the system manager grants a user the TMPMBX privilege, the user's process is allowed to create a temporary mailbox by executing the Create Mailbox and Assign Channel system service.

Mailboxes are buffers in virtual memory that are treated as if they were record-oriented I/O devices. A mailbox is used for general interprocess communication. Unlike a permanent mailbox, which must be explicitly deleted, a temporary mailbox is deleted automatically when no more channels are assigned to it.

The system manager should usually grant this privilege to all users of the system to facilitate interprocess communication. System performance is not likely to be degraded by permitting the creation of temporary mailboxes, because their number is controlled by limits on the use of system dynamic memory.

5.3.24 The VOLPRO Privilege

This is the privilege of overriding volume protection.

This privilege is needed for the following reasons: 1) to initialize a previously used volume with an owner UIC different from one's own UIC, 2) to override the expiration date on a non-owned tape or disk volume, 3) to mount a non-owned Files-11 volume with the /FOREIGN qualifier, and 4) to override the owner UIC protection of a volume.

5.3.25 The WORLD Privilege

This is the privilege that lets a process control all other processes and examine the parameters of all other processes.

If the system manager grants a user the WORLD privilege, the user's process is allowed to affect other processes both within and outside its group. This is done by executing the following process control system services: the Suspend Process system service, the Resume Process system service, the Delete Process system service, the Set Priority system service, the Wake system service, the Schedule Wakeup system service, the Cancel Wakeup system service, and the Force Exit system service. The user's process is also allowed to examine processes outside its own group by executing the Get Job/Process Information system service.

The user who has this privilege can also execute the STOP (Image or Process) command for any process.

To exercise control over subprocesses that it created or to examine these subprocesses, a process needs no special privilege. To affect or to examine other processes within its own group, a process needs only the GROUP privilege. But to affect or examine processes outside its own group, a process needs the WORLD privilege.

CHAPTER 6

ACCOUNTING FOR THE USE OF SYSTEM RESOURCES

For accounting purposes, the VAX/VMS system keeps records of the use of system resources. These records are kept in the accounting file (ACCOUNTNG.DAT), which is updated each time an accountable process terminates, each time a printing job is completed, and each time a log-in failure occurs. In addition, users can send messages to be inserted into the accounting file.

Accounting records contain cumulative accounts of the resources used either by processes or subprocesses set up for users or by print symbionts that print out files for users. Each accounting record contains two fields -- user name and account name -- that identify the user and establish the connection between the accounting record and a user of the system. These fields correspond to similar fields of the user's account record in the user authorization file (UAF).

Using the detailed accounting records provided by the system, the system manager or a system programmer can devise programs for reporting on the use of system resources and for billing for their use.

Because the users of system resources are identified in two ways, reports on the use of system resources and bills for the use of system resources can be prepared in either of two ways: by user name or by account name.

6.1 THE ACCOUNTING FILE

The accounting file is a sequential file that is created and opened automatically when the operating system is initialized. Records are entered in this file under the following six conditions:

1. When an interactive process terminates
2. When a batch process terminates
3. When a subprocess or a detached process terminates
4. When a printing job is completed
5. When a log-in failure occurs
6. When a user sends a message to the accounting file by use of the Send Message to Accounting Manager system service

Accounting records are arranged chronologically in this file.

To suppress the accounting function and thus to avoid accounting for the use of system resources requires privilege. Only a user who has

the ACNT privilege can create subprocesses or detached processes in which accounting is disabled. The /NOACCOUNTING qualifier of the RUN command is used to disable all accounting in a created process.

A user with OPER privilege can selectively disable various kinds of accounting system wide by using the /DISABLE qualifier of the SET ACCOUNTING command. Usually, this is a task of the system operator. See the VAX/VMS Operator's Guide for a full description of the SET ACCOUNTING command.

As records are entered in the accounting file, all but print job completion records are immediately flushed to disk. This precaution guarantees the integrity of the file and the completeness of accounting data even if the system fails.

Normally, the accounting file is closed at the end of a billing period. The current version of the accounting file is closed and a new version of the file is created and opened. As a rule, this is the system operator's job, done by use of the SET ACCOUNTING command.

If an attempt to write to the accounting file results in an error, the file is closed automatically and a new copy is created and opened.

The following list summarizes the characteristics of the accounting file:

- File name: ACCOUNTNG.DAT (this file is not an ASCII file; hence it must be formatted before it is printed)
- Residence: the system device
- Directory: [SYSMGR]
- File organization: sequential
- Record length: variable length
- Record types: six (see Section 6.2 below)

6.2 RECORDS IN THE ACCOUNTING FILE

The accounting file contains six types of records, corresponding to the six conditions that cause records to be written to the file. These records are:

1. Records written when interactive processes terminate
2. Records written when batch processes terminate
3. Records written when subprocesses or detached processes terminate
4. Records written when printing jobs are completed
5. Records written when log-in failures occur
6. Records written when users' messages are sent to the accounting file

For a detailed description of the records of the accounting file, see the discussion of the Send Message to Accounting Manager system service in the VAX/VMS System Services Reference Manual.

CHAPTER 7

USING THE USER AUTHORIZATION PROGRAM

The User Authorization Program (AUTHORIZE) is a system utility program that the system manager uses to maintain the user authorization file (UAF). As described in Chapter 3, this file contains a default value record, a system management account record, and one record for each user of the system.

The AUTHORIZE program lets the system manager:

- Create the UAF, if one does not exist. A newly created UAF contains only the default value record and the system management account record; no users are yet known to the system.
- Define a new user of the system by creating a record for that user in the UAF and thus granting privileges and specifying limits and priority.
- Take away a user's right to use the system by deleting that user's record from the UAF.
- Change the default record of the UAF.
- Change a user's privileges, limits, or priority by modifying that user's record in the UAF.
- Display all information about a user's account, with the exception of that user's password.
- Make a listing of all records in the UAF.

Once a plan for setting up a meaningful system of accounts has been devised, the system manager proceeds to define users to the system.

Usually, in setting up a user's account, the system manager elects to insert default values into most fields of the user's record in the UAF. Sometimes, however, the system manager may elect not to use these default values. After all, the system manager's job is to regulate system performance, and among the most powerful means of doing so are the abilities to grant special privileges to some users; to withhold privileges from other users; and to specify different limits and priorities, as required.

To override the built-in defaults, the system manager must have a general idea about how the system as a whole works and must understand the significance of each of the fields of the UAF record. Chapter 3 covers the form and content of the records in the UAF; Chapter 5 covers the topics of privilege, limits, and priority. These chapters are required reading for system managers who need to override default values in users' records of the UAF.

Most of this chapter is devoted to the mechanics of using AUTHORIZE to maintain the UAF: Section 7.1 discusses creating the UAF; Section 7.2 discusses changing the UAF; Section 7.3 describes each of the commands that AUTHORIZE accepts.

Anyone who has the proper default device and directory (SYS$SYSTEM:) and who has read, write, and delete access to the UAF (anyone with a system UIC) can use AUTHORIZE. As a rule, however, its use should be restricted to the system manager or to a system operator. AUTHORIZE, which can be run from any terminal, is started up by use of the DCL command RUN AUTHORIZE. AUTHORIZE can be executed either interactively (from commands entered one at a time or from a command procedure) or as a batch job; Sections 7.4 and 7.5 contain examples of interactive and batch use of AUTHORIZE, respectively.

If users defined to the system by use of AUTHORIZE intend to create subdirectories on public volumes and to create and store files on public volumes, the system manager must create user file directories (UFDs) for these users. Usually, the system manager creates UFDs that correspond to the users' default directories, but a user can have more than one directory. To create such a user file directory, the system manager uses the DCL command CREATE, as follows:

```
CREATE/DIRECTORY DBA1:[MILLER]/OWNER=[140,10]
```

The /OWNER qualifier is required so that the system manager will not be the owner of the directory file.

Users can create their own UFDs on private volumes that have been suitably initialized, as explained in the VAX/VMS Command Language User's Guide.

If inadvertently destroyed, the UAF can be reconstructed from back-up the same way as any other system file, or, if the UAF was generated by use of a command procedure, by re-executing that command procedure.

7.1 CREATING THE UAF

When AUTHORIZE is started up, it attempts to gain access to the UAF. If the UAF is not found, AUTHORIZE asks the system manager whether or not a new UAF should be created, as follows:

```
Do you want to create a new file?
```

If the system manager responds with YES (or Y), a new UAF is created. A newly created UAF contains only two records: the default value record and the system management account record.

The default record is simply a user account record with the user name DEFAULT. The initial default values placed in the default record are built into AUTHORIZE. However, the system manager can change this record once the UAF is created. Unlike other user account records, however, the default value record cannot be deleted by the system manager.

Similarly, the initial values of the system management account record are built into AUTHORIZE. When it is first created, the system management account has the user name SYSTEM and the password MANAGER. For security purposes, the system manager should change the initial password after the UAF is created. Like the default record, the system management account record cannot be deleted.

7.2 CHANGING THE UAF

When AUTHORIZE is run, the UAF (SYS$SYSTEM:SYSUAF.DAT) is copied to a temporary file (SYS$SYSTEM:SYSUAF.TMP) for processing. Thus, all modifications of the UAF are performed on the temporary file.

When execution of AUTHORIZE terminates normally, the current UAF is automatically replaced by the temporary file. When, however, the execution of the program terminates abnormally, the current UAF is not affected by any of the attempted changes. In fact, if the system manager aborts the execution of AUTHORIZE at any time, the current UAF remains as it was before attempts were made to change it.

Using AUTHORIZE to change users' records in the UAF has no immediate effect on users who were logged in to the system before execution of AUTHORIZE began and who remained logged in during its execution. Changes made to their account records have no effects on these users until they log out and log in again.

7.3 COMMANDS AND FUNCTIONS OF THE AUTHORIZE PROGRAM

AUTHORIZE accepts DCL-like command lines in response to prompts. The prompt (UAF>) and the general form of command lines are as follows:

```
UAF>command [username][options]
```

command

The command is a required part of the command line. The system manager must respond to the prompt with one of the following commands:

ADD	LIST
DEFAULT	MODIFY
EXIT	REMOVE
HELP	SHOW

These commands can be abbreviated to their initial letters. They are described in detail in Sections 7.3.1 through 7.3.8.

username

A string of up to 12 alphanumeric characters. This optional field is not used with the DEFAULT, EXIT, HELP, and LIST commands; it is required with ADD, MODIFY, REMOVE, and SHOW.

The username field in a command string corresponds to the username field in the UAF record.

options

Table 7-1 lists the options used in AUTHORIZE command strings. These options are used only with the ADD, DEFAULT, and MODIFY commands. An option consists of four parts, arranged as follows:

1. A slash (/).
2. An option name. All option names are unique in four characters; most are unique in three. An option name refers to a field of a UAF record.
3. An equal sign (=) or a colon (:).
4. A value or string that is to be inserted into a field of a UAF record. This field is referred to by the option name. Values are decimal numbers.

The following command lines are acceptable to the AUTHORIZE program. These examples show two equally satisfactory ways of making the same modification to a user record (for the user named USER1) in the UAF.

```
UAF>MODIFY USER1/PASSWORD=MYPASS/PRIORITY=2

UAF>M USER1/PASS:MYPASS/PRIO:2
```

If necessary, a command string can continue onto more than one line. This is done by entering a hyphen (-) as the last character of each line that is to be continued on another line. The following command string illustrates the use of the hyphen as a continuation character.

```
UAF>ADD USER2/OWNER=HENRY/PASSWORD=HANK/UIC=-

_[300,010]/DIRECTORY=[WILLS]/DEVICE=DB2:/ACCOUN-

_T=BUDGE
```

In principle, by using the AUTHORIZE commands and options, the system manager can define or redefine the contents of any field in a UAF record. In practice, however, default values are usually chosen for most fields of users' records in the UAF.

Thus, when creating a new user's record in the UAF (by use of the command ADD), the system manager will often explicitly specify only the options PASSWORD, UIC, and DIRECTORY and define their values but will accept default values for all privileges and limits.

These default values are assigned automatically from the default value record of the UAF whenever the system manager omits explicit reference to an option. That is, the system manager selects a default value by doing nothing.

Similarly, when modifying the default value record of the UAF (by use of the command DEFAULT) or when modifying a user's record in the UAF (by use of the command MODIFY), the system manager will often specify new contents for only a few fields. By default, all other fields will remain unchanged. Again, by doing nothing, the system manager ensures that the contents of these fields do not change. In other words, in modifying a record of the UAF, if the system manager omits explicit reference to an option, the contents of the field corresponding to the option do not change.

Table 7-1
Options of AUTHORIZE
(Arranged Alphabetically)

Option	Description
/ACCOUNT=account-name	Default account name. Except for the hyphen, the string cannot contain embedded blanks, tabs, or special characters. Maximum length of string: 8 characters
/ASTLM=value	Number of AST entries that can be outstanding at the same time

(continued on next page)

Table 7-1 (Cont.)
Options of AUTHORIZE
(Arranged Alphabetically)

Option	Description
/BIOLM=value	Number of buffered I/O operations that can be outstanding at the same time
/BYTLM=value	Amount of dynamic memory available for buffering concurrent I/O requests
/CLI=cli-name	Name of the default command interpreter (DCL or MCR)
/DEVICE=device-name:	Name of default device. Maximum length of string: 15 characters with trailing colon (:); AUTHORIZE adds colon if the system manager does not supply it
/DIOLM=value	Direct I/O limit
/DIRECTORY=directory-name	Name of the default directory. Maximum length of string: 31 characters, including brackets, which the system manager must supply
/FILLM=value	Number of files that can be open and network logical links that can be active at the same time
/FLAGS= (flagname, NOflagname)	Specification of log-in flags. If only a single flag is specified, parentheses are unnecessary. If, however, a list of log-in flags is specified, it must be enclosed in parentheses. Entries in a list of log-in flags must be separated from each other by commas. To set a log-in flag not previously set, enter the name of the flag (either DISCTLY or DEFCLI). To clear a log-in flag previously set, enter the name of the flag prefixed by the word NO (either NODISCTLY or NODEFCLI)

(continued on next page)

Table 7-1 (Cont.)
Options of AUTHORIZE
(Arranged Alphabetically)

Option	Description
/LGICMD=filename	File name of a user's log-in command file. Maximum length of string: 19 characters Device and directory names are the default names and should not be specified; file type is .COM by default and should not be specified.
/OWNER="owner name"	Owner's name. String must be enclosed in quotation marks if it contains blanks. Maximum length of string: 20 characters
/PASSWORD=password	User's password. Maximum length of string: no limit
/PGFLQUOTA=value	Limit on paging file space that the process can use (specified in 256-page blocks)
/PRCLM=value	Limit on number of subprocesses that can exist at any one time
/PRIORITY=value	Default base priority of user's processes
/PRIVILEGE= (privname,NOprivname,...)	Specification of process privileges. If only a single privilege is specified, parentheses are unnecessary. If, however, a list of privileges is specified, it must be enclosed in parentheses. Entries in a list of privileges must be separated from each other by commas. To grant a privilege not previously granted, the system manager enters the name of the privilege (see Table 5-1); for example, CMKRNL. To take away a privilege previously granted, the system manager enters the name of the privilege (see Table 5-1) prefixed by the word NO; for example, NOGRPNAM.

(continued on next page)

Table 7-1 (Cont.)
Options of AUTHORIZE
(Arranged Alphabetically)

Option	Description
/TQELM=value	Number of timer queue entries that can exist at any one time
/UIC=[group,member]	User identification code. Group and member numbers are three-digit octal numbers between 0 and 377, inclusive.
/WSDEFAULT=value	Default working set size in pages
/WSQUOTA=value	Maximum size of working set in pages

After the system manager has successfully added, modified, or removed a user's record in the UAF, AUTHORIZE confirms the fact with a suitable message. Similarly, problems encountered in the execution of AUTHORIZE are reported to the manager. Table 7-2 summarizes the messages that AUTHORIZE displays.

Table 7-2
Messages Displayed by AUTHORIZE
(Arranged Alphabetically)

Message	Response
command is not unique	Enter command name in its entirety
connect error	This message is followed by a VAX-11 RMS error message; see the VAX/VMS System Messages and Recovery Procedures Manual for user action on the RMS error
device name too long to add trailing ":"	Enter device name of proper length, as shown in Table 7-1
Do you want to create a new file?	To create a new UAF, respond YES or Y; to terminate execution of AUTHORIZE, respond NO or N

(continued on next page)

Table 7-2 (Cont.)
Messages Displayed by AUTHORIZE
(Arranged Alphabetically)

Message	Response
error creating listing file	This message is followed by a VAX-11 RMS error message; see the VAX/VMS System Messages and Recovery Procedures Manual for user action on the RMS error
error in UIC specification	Enter two octal numbers [g,m] with values between 000 and 377, inclusive
error in value specification	Enter a proper value for the specified option
****ERROR RENAMING TEMPORARY FILE	This message is followed by a VAX-11 RMS error message; see the VAX/VMS System Messages and Recovery Procedures Manual for user action on the RMS error
get error	This message is followed by a VAX-11 RMS error message; see the VAX/VMS System Messages and Recovery Procedures Manual for user action on the RMS error
invalid command	Enter correct command name: A,D,E,H,L,M,R, or S
invalid option name	Enter correct option name, as shown in Table 7-1
invalid privilege name	Enter correct privilege name, as shown in Table 5-1
invalid response	Enter Y or N to the question "Do you want to create a new file?"
invalid username, username already exists	Enter a new and unique user name

(continued on next page)

Table 7-2 (Cont.)
Messages Displayed by AUTHORIZE
(Arranged Alphabetically)

Message	Response
listing file SYSUAF.LIS complete	Informative message; no response needed
missing argument for option	Enter needed argument for specified option, as shown in Table 7-1
missing username	Enter required user name in ADD, MODIFY, REMOVE, and SHOW commands
no modifications made	Informative message; no response needed
option name not unique	Enter at least four characters of option name
privilege name not unique	Enter at least four characters of privilege name
put error	This message is followed by a VAX-11 RMS error message; see the VAX/VMS System Messages and Recovery Procedures Manual for user action on the RMS error
quoted string missing end quote	Reenter command, supplying required pair of quotes
string too long for field	Enter string of proper length, as shown in Table 7-1
The default record may not be removed	Informative message; no response needed
The SYSTEM record may not be removed	Informative message; no response needed
UAF updates completed	Informative message; no response needed

(continued on next page)

Table 7-2 (Cont.)
Messages Displayed by AUTHORIZE
(Arranged Alphabetically)

Message	Response
unable to add user record	This message is followed by a VAX-11 RMS error message; see the VAX/VMS System Messages and Recovery Procedures Manual for user action on the RMS error
unable to create temporary UAF	This message is followed by a VAX-11 RMS error message; see the VAX/VMS System Messages and Recovery Procedures Manual for user action on the RMS error
unable to delete record	This message is followed by a VAX-11 RMS error message; see the VAX/VMS System Messages and Recovery Procedures Manual for user action on the RMS error
unable to open SYSUAF.DAT	This message is followed by a VAX-11 RMS error message; see the VAX/VMS System Messages and Recovery Procedures Manual for user action on the RMS error
unable to update record	This message is followed by a VAX-11 RMS error message; see the VAX/VMS System Messages and Recovery Procedures Manual for user action on the RMS error
username does not exist	Enter the user name of an existing account
username too long	Enter a user name of no more than 12 alpha-numeric characters
user record removed	Informative message; no response needed

(continued on next page)

Table 7-2 (Cont.)
Messages Displayed by AUTHORIZE
(Arranged Alphabetically)

Message	Response
user record successfully added	Informative message; no response needed
user record updated	Informative message; no response needed
value too large for field	Enter a suitable value for the specified option
writing listing file	Informative message; no response needed

7.3.1 ADD Command

The ADD command is used in creating a new user's record in the UAF.

The form of the ADD command line is as follows:

```
ADD username [options]
```

A unique user name is the only information for the user's record that the system manager must supply. If the system manager specifies a user name that has already been assigned to another user, AUTHORIZE rejects the name and displays an error message.

The default of any option can be selected by simply omitting explicit reference to the option.

Table 7-3 lists the options that are most likely to be referred to explicitly in an ADD command. They are arranged for quick reference in the order in which they are likely to be used. For a complete list of options, see Table 7-1.

Table 7-3
Frequently Used Options of ADD Command

Option	Form of Entry[1]	Example of String
Owner's Name	OWNER="owner name"	"JOHN DOE"
Password	PASSWORD=password	150-20-9000
UIC	UIC=[group,member]	[310,010]
Default Directory	DIRECTORY=directory-name	[JOHN]
Default Device	DEVICE=device-name:	DB2:
Account Name	ACCOUNT=account-name	32V-001

[1] The minimum form of representation of each option name is denoted by an underline.

7.3.2 DEFAULT Command

The DEFAULT command is used in modifying the default value record of the UAF.

The form of the DEFAULT command line is as follows:

```
DEFAULT options
```

To change a field of the default value record, an option must be specified explicitly in this command. Fields whose related options are not referred to explicitly remain unchanged, by default.

7.3.3 EXIT Command

The EXIT command is used in terminating a session with AUTHORIZE and in returning control to the DCL command level.

The form of the EXIT command line is as follows:

```
EXIT
```

If any changes were made in the temporary UAF file, the temporary file becomes the current UAF.

There are two other ways in which a session with AUTHORIZE can be terminated normally: 1) when the control character CTRL/Z (which signals end of file) is entered at a terminal and 2) when a command procedure reaches end of file.

Entering the control character CTRL/Y aborts the execution of AUTHORIZE; no changes are then made to the UAF.

7.3.4 HELP Command

The HELP command is used in displaying summaries of the AUTHORIZE commands and options.

The form of the HELP command line is as follows:

```
HELP
```

7.3.5 LIST Command

The LIST command is used in producing a file (SYSUAF.LIS) which contains a summary of each record of the UAF.

The form of the LIST command line is as follows:

```
LIST
```

The file SYSUAF.LIS can be printed by use of the DCL command PRINT and saved for reference. In the listing, a user is identified as privileged (PRIV) if that user has the CMKRNL privilege and is identified as nonprivileged (NOPRIV) otherwise. Passwords are not revealed in the listing.

7.3.6 MODIFY Command

The MODIFY command is used in modifying the system management record (rarely) or a user's record of the UAF. The only field of the UAF that cannot be changed by use of this command is the user name field.

The form of the MODIFY command line is as follows:

```
MODIFY username options
```

The system manager must supply the user name of the record to be modified.

To change a field of a UAF record, an option must be specified in this command. Fields whose related options are not referred to explicitly remain unchanged, by default.

7.3.7 REMOVE Command

The REMOVE command is used in deleting a user's record from the UAF. As a result, the user whose record is deleted can no longer log in to the system.

The form of the REMOVE command line is as follows:

```
REMOVE username
```

The only information the system manager must supply is the user name of the record being deleted.

The default value record and the system management account record cannot be removed; the AUTHORIZE program permits the deletion of users' records only.

7.3.8 SHOW Command

The SHOW command is used in displaying all fields of a record of the UAF except the password.

The form of the SHOW command line is as follows:

```
SHOW username
```

The only information the system manager must supply is the user name of the record to be displayed.

7.4 EXAMPLES OF THE INTERACTIVE USE OF THE AUTHORIZE PROGRAM

This section presents examples of the use of the AUTHORIZE program from an interactive terminal. Section 7.4.1 illustrates a typical interactive session with AUTHORIZE -- showing how to start it up, how to use typical commands, and how to terminate execution.

Sections 7.4.2 through 7.4.9 illustrate separately the effects of using each of the eight AUTHORIZE commands, as follows:

1. ADD (Section 7.4.2)
2. DEFAULT (Section 7.4.3)
3. EXIT (Section 7.4.4)
4. HELP (Section 7.4.5)
5. LIST (Section 7.4.6)
6. MODIFY (Section 7.4.7)
7. REMOVE (Section 7.4.8)
8. SHOW (Section 7.4.9)

7.4.1 A Typical Interactive Session with AUTHORIZE

This example shows how to start up AUTHORIZE and how to use a number of the commands. Note that each of the commands that appears in this example is fully explained in Section 7.3 and in the examples in the sections that follow.

```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF>ADD ROE/PAS=RICK/UIC=[143,010]/DIR=[RICHARD]
user record successfully added
UAF>SHOW DOE
USERNAME: DOE                 OWNER: JOHN DOE
ACCOUNT: 32V-001              UIC: [310,010]
DIRECTORY: [JOHN]             DEVICE: DB2:
LOGIN COMMAND FILE:           LOGIN FLAGS:
CLI:  DCL          PRCLM:       8  PRIO:          4
ASTLM:      10     BIOLM:       6  BYTLM:       4096
DIOLM:       6     FILLM:      20  TQELM:         10
WSDEFAULT: 150     WSQUOTA:   200  PGFLQUOTA:     40
PRIVILEGES:
   GROUP PRMMBX TMPMBX
UAF>MODIFY DOE/PRIVILEGE=NOPRMMBX
user record updated
UAF>REMOVE SMITH
user record removed
UAF>LIST
writing list file
listing file SYSUAF.LIS complete
UAF>EXIT
UAF updates completed
$
```

The prompt $ indicates that control has returned to the DCL command level.

7.4.2 Using the ADD Command

This example shows how to use the ADD command to create new users' records in the UAF.

In response to the UAF prompt, the ADD command is entered. Note that AUTHORIZE confirms the creation of a new record.

```
UAF>ADD DOE/OWN="JOHN DOE"/PAS=150-20-9000-
_  /UIC=[310,010]/DIR=[JOHN]/DEV=DB2:-
_  /ACCO=32V-001
user record successfully added
```

A record has been created in the UAF for the user Doe. Doe's record is displayed here by use of the SHOW command.

```
UAF>SHOW DOE
USERNAME: DOE                 OWNER: JOHN DOE
ACCOUNT: 32V-001              UIC: [310,010]
DIRECTORY: [JOHN]             DEVICE: DB2:
LOGIN COMMAND FILE:           LOGIN FLAGS:
CLI: DCL          PRCLM:        8  PRIO:          4
ASTLM:       10   BIOLM:        6  BYTLM:       4096
DIOLM:        6   FILLM:       20  TQELM:         10
WSDEFAULT: 150    WSQUOTA:    200  PGFLQUOTA:     40
PRIVILEGES:
   GROUP PRMMBX TMPMBX
```

Compare Doe's record with the default value record of the UAF, displayed here by use again of the SHOW command.

```
UAF>SHOW DEFAULT
USERNAME: DEFAULT             OWNER:
ACCOUNT:                      UIC: [200,200]
DIRECTORY: [USER]             DEVICE:
LOGIN COMMAND FILE:           LOGIN FLAGS:
CLI: DCL          PRCLM:        8  PRIO:          4
ASTLM:       10   BIOLM:        6  BYTLM:       4096
DIOLM:        6   FILLM:       20  TQELM:         10
WSDEFAULT: 150    WSQUOTA:    200  PGFLQUOTA:     40
PRIVILEGES:
   GROUP PRMMBX TMPMBX
```

Note that these two records are the same except for USERNAME and for the fields specified explicitly in the ADD command (OWNER, UIC, DIRECTORY, DEVICE, and ACCOUNT). The password differs, too, but it cannot be displayed.

7.4.3 Using the DEFAULT Command

This example shows how to use the DEFAULT command to modify the default value record of the UAF.

In response to the UAF prompt, the DEFAULT command is entered. Note that AUTHORIZE confirms the updating of the default value record.

```
UAF>DEFAULT/PRIVILEGE=(NOGROUP,NOPRMMBX)
user record updated
```

The default value record of the UAF has been modified. This modified record is displayed here by use of the SHOW command. Compare this record with the original default value record (Section 7.4.2).

```
UAF>SHOW DEFAULT
USERNAME: DEFAULT             OWNER:
ACCOUNT:                      UIC: [200,200]
DIRECTORY: [USER]             DEVICE:
LOGIN COMMAND FILE:           LOGIN FLAGS:
CLI: DCL          PRCLM:        8  PRIO:          4
ASTLM:       10   BIOLM:        6  BYTLM:       4096
DIOLM:        6   FILLM:       20  TQELM:         10
WSDEFAULT: 150    WSQUOTA:    200  PGFLQUOTA:     40
PRIVILEGES:
   TMPMBX
```

Why modify the default value record of the UAF? One reason is to simplify adding a group of users to whom many of the same attributes but not the attributes contained in the original default value record are to be given. Thus, before adding these users, the system manager can modify the default value record. Then, in creating UAF records for these users, the system manager need not specify these attributes to have the newly defined default values assigned to the new users. After adding new users, the system manager restores the original values to the default value record.

7.4.4 Using the EXIT Command

This example shows how to use the EXIT command to terminate a session with AUTHORIZE.

In response to the UAF prompt, the EXIT command is entered. This command does two things: it renames the temporary UAF file, and it returns control to the DCL command level. Note that AUTHORIZE confirms any updating of the UAF.

```
UAF>EXIT
UAF updates completed
$
```

7.4.5 Using the HELP Command

This example shows how to use the HELP command to display a summary of the commands and options of the AUTHORIZE program.

In response to the UAF prompt, the HELP command is entered. Note that the commands and options of AUTHORIZE are displayed.

```
UAF>HELP

Commands are:

    ADD username [/options]    ! define new user
    DEFAULT /options           ! change default record
    EXIT                       ! exit program
    HELP                       ! type this message
    LIST                       ! produce ASCII file of usernames
    MODIFY username /options   ! update a user's record
    REMOVE username            ! delete a user's record
    SHOW username              ! display values in a user record

Options are:

    /ACCOUNT , /ASTLM , /BIOLM , /BYTLM , /CLI ,
    /DEVICE , /DIOLM , /DIRECTORY , /FILLM , /FLAGS ,
    /LGICMD , /OWNER , /PASSWORD , /PGFLQUOTA , /PRCLM ,
    /PRIORITY , /PRIVILEGES , /TQELM , /UIC , /WSDEFAULT ,
    /WSQUOTA
```

7.4.6 Using the LIST Command

This example shows how to use the LIST command to produce a listing of users and their characteristics.

In response to the UAF prompt, the LIST command is entered. Note that AUTHORIZE confirms 1) that the listing file (SYSUAF.LIS) is being written and 2) that the listing file is complete.

```
UAF>LIST
writing listing file
listing file SYSUAF.LIS complete
```

When it is convenient, the listing file can be queued for printing with the DCL command PRINT.

7.4.7 Using the MODIFY Command

This example shows how to use the MODIFY command to change a user's record in the UAF.

In response to the UAF prompt, the MODIFY command is entered. Note that AUTHORIZE confirms the updating of the record.

```
UAF>MODIFY DOE/PRIVILEGE=NOGROUP
user record updated
```

Doe's user record has been modified. The SHOW command can be used to display Doe's updated record.

```
UAF>SHOW DOE
USERNAME: DOE                 OWNER: JOHN DOE
ACCOUNT: 32V-001              UIC: [310,010]
DIRECTORY: [JOHN]             DEVICE: DB2:
LOGIN COMMAND FILE:           LOGIN FLAGS:
CLI: DCL          PRCLM:        8  PRIO:           4
ASTLM:       10   BIOLM:        6  BYTLM:        4096
DIOLM:        6   FILLM:       20  TQELM:          10
WSDEFAULT: 150    WSQUOTA:    200  PGFLQUOTA:      40
PRIVILEGES:
   PRMMBX TMPMBX
```

Compare the updated record with Doe's original record (Section 7.4.2). As expected, Doe no longer has the GROUP privilege.

7.4.8 Using the REMOVE Command

This example shows how to use the REMOVE command to delete a user's record from the UAF.

In response to the UAF prompt, the REMOVE command is entered. Note that AUTHORIZE confirms the removal of the record.

```
UAF>REMOVE DOE
user record removed
```

Remember, neither the default value record nor the system management account record can be deleted. If an attempt is made to remove either of them, AUTHORIZE gives notice of this fact, as follows.

```
UAF>REMOVE DEFAULT
The default record may not be removed
```

7.4.9 Using the SHOW Command

This example shows how to use the SHOW command to display a record of the UAF.

In response to the UAF prompt, the SHOW command is entered. The specified record is displayed. Note that passwords are not revealed by the use of SHOW.

```
UAF>SHOW SYSTEM


USERNAME: SYSTEM          OWNER: SYSTEM MANAGER
ACCOUNT:                  UIC: [001,004]
DIRECTORY: [SYSMGR]       DEVICE:
LOGIN COMMAND FILE:       LOGIN FLAGS:
CLI: DCL          PRCLM:      10  PRIO:          4
ASTLM:       20   BIOLM:       6  BYTLM:     20480
DIOLM:       12   FILLM:      20  TQELM:        20
WSDEFAULT: 150    WSQUOTA:  1024  PGFLQUOTA:    40
PRIVILEGES:
   CMKRNL CMEXEC SYSNAM GRPNAM ALLSPOOL DETACH DIAGNOSE LOG_IO
   GROUP ACNT PRMCEB PRMMBX PSWAPM ALTPRI SETPRV TMPMBX
   WORLD OPER EXQUOTA NETMBX VOLPRO PHY_IO BUGCHK PRMGBL
   SYSGBL MOUNT

UAF>SHOW DOE


USERNAME: DOE             OWNER: JOHN DOE
ACCOUNT: 32V-001          UIC: [310,010]
DIRECTORY: [JOHN]         DEVICE: DB2:
LOGIN COMMAND FILE:       LOGIN FLAGS:
CLI: DCL          PRCLM:      8   PRIO:          4
ASTLM:       10   BIOLM:      6   BYTLM:      4096
DIOLM:        6   FILLM:     20   TQELM:        10
WSDEFAULT: 150    WSQUOTA:  200   PGFLQUOTA:    40
PRIVILEGES:
   GROUP PRMMBX TMPMBX
```

7.5 EXAMPLE OF THE BATCH USE OF THE AUTHORIZE PROGRAM

The UAF can also be produced and manipulated by the batch use of the AUTHORIZE program. To submit a command procedure for batch execution, use the DCL command SUBMIT, as follows:

```
$SUBMIT filename
```

The parameter (filename) in this command line refers to the name of a command procedure.

The following example shows a listing of the file UAF2.COM, which is typical of the kind of command procedure that can be used in the batch execution of AUTHORIZE. This file is submitted by use of the following command line:

```
$SUBMIT  UAF2.COM
```

Remember, in the batch execution of UAF2.COM, AUTHORIZE looks for inputs (valid AUTHORIZE commands) in the command procedure and produces output messages in the batch log file (UAF2.LOG). The log file is printed out on the system line printer.

NOTE

As shown in the example, all letters in a command procedure used in executing AUTHORIZE must be uppercase letters.

```
$ SET VERIFY
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
YES                                                    (1)
ADD USER1/OWN="USER 1"/PAS=32V/UIC=[310,010]/DIR=[USER1]/ACCO=32V-01
ADD USER2/OWN="USER 2"/PAS=32W/UIC=[310,020]/DIR=[USER2]/ACCO=32W-01   (2)
ADD USER3/OWN="USER 3"/PAS=32X/UIC=[310,030]/DIR=[USER3]/ACCO=32X-01
DEFAULT/UIC=[143,010]/DIR=[CLASS]/ACCO=TRAINING/PRIVILEGE=-            (3)
(NOGROUP,NOPRMMBX,NOTMPMBX)
ADD STUDENT1/PAS=565ADI                                (4)
ADD STUDENT2/PAS=325ALY
DEFAULT/UIC=[200,200]/DIR=[USER]/ACCO=DEFAULT/PRIV=(GROUP,-            (5)
PRMMBX,TMPMBX)
LIST                                                   (6)
EXIT                                                   (7)
```

The execution of this command procedure (UAF2.COM) causes the following to happen.

1. The creation of a UAF if none exists.
2. The creation of three new users' records (for USER1, USER2, and USER3).
3. The modification of the default record in anticipation of adding two students' records.
4. The creation of two new users' records for STUDENT1 and STUDENT2. These users share UIC, directory, and account name and have no privileges. The records created for these students are unique only in their user names and their passwords.
5. The restoration of the default record to its original condition (except for account name).
6. The generation of the listing file, which can be printed out at the convenience of the system manager.
7. The termination of the execution of AUTHORIZE.

PART III

MANAGING PUBLIC FILES AND VOLUMES

Typically, the overall planning and management of a system of public files and volumes are among the principal responsibilities of the system manager; carrying out many of the specific tasks of managing public files and volumes is usually a duty of a system operator.

This part of the VAX/VMS System Manager's Guide contains discussions of the following aspects of managing public files and volumes:

- Initializing and mounting public volumes
- Regularly backing up public files and volumes
- Installing commonly used and privileged executable images as known images
- Installing commonly used shareable images as permanent global sections
- Establishing system logical names needed for running the executable images provided by DIGITAL and for running other images available to all users at an installation

Chapters 8 through 11 discuss these aspects of system management, as follows:

- Chapter 8: Initializing and Mounting Public Volumes
- Chapter 9: Backing Up Public Files and Volumes
- Chapter 10: Installing Known Images and Creating Permanent Global Sections
- Chapter 11: Assigning System Logical Names

CHAPTER 8

INITIALIZING AND MOUNTING PUBLIC VOLUMES

Public volumes are file-structured disk volumes that contain public files, which normally must be available to most if not all users of a system. Public volumes may also contain files that users create for their own private use or for general use. Thus, as long as UIC-based file protection permits it, all users have access to public, or system, volumes and to the files contained on these public volumes.

Public volumes contain the following kinds of public files supplied by DIGITAL.

- The operating system itself in executable form and files related to the operating system
- Utility programs in executable form
- Diagnostic and test programs in executable form and files related to these programs
- Various system libraries: macro libraries, object module libraries, and shared run-time libraries
- Text files; for example, the system error message file and help files
- Optional software in executable form, plus related libraries and other files

In addition, the system manager can include on public volumes files that are unique to an installation. These typically are files that must be accessible to many if not all users of the installation.

Finally, the system manager can permit any user to create, catalog, and store files on a public volume. Depending on their file protection, these files may be generally accessible, or access to them may be restricted. This use of a public volume, however, is subject to limitation: a user is free to create, catalog, and store files on a public volume only if volume protection permits and if the user has write access to a directory on the volume. As a rule, the system manager creates a default disk file directory on a public volume for everyone who is authorized to use the system (see Chapter 7).

Knowing how to initialize and mount public disk file volumes is a prerequisite to managing a system of public files and volumes.

8.1 INITIALIZING PUBLIC DISK FILE VOLUMES

The purpose of initializing a disk volume is to delete all information from the volume and to impart to the volume a structure that the operating system recognizes. It is this structure that prepares a volume to receive data and that permits the operating system to locate data that have been stored on the volume.

The following discussion of disk file structure is restricted to Files-11 Structure Level 1 and Files-11 Structure Level 2. Files-11 Structure Level 2 is the default disk structure of the VAX/VMS system, and Files-11 Structure Level 1 is a structure used by DIGITAL's RSX-11M, RSX-11D, and IAS operating systems. Structure Level 1 can be both recognized and created by the VAX/VMS system.

The VAX/VMS Command Language User's Guide contains most of what the system manager needs to know about initializing volumes in 1) the description of the INITIALIZE command in Part II and 2) Chapter 3, "Disk and Tape Volumes."

The following sections contain:

- A general description of Files-11 Structure Level 1 and Files-11 Structure Level 2 (Section 8.1.1)
- A brief discussion of the main differences between the two levels of disk file structure, and what these differences mean to the system manager (Section 8.1.2)
- Special guidelines for initializing disk file volumes for public use (Section 8.1.3)

8.1.1 Files-11 Disk Structure

When a Files-11 Structure Level 2 volume is initialized, 10 files that control the structure of the volume are created. When a Files-11 Structure Level 1 volume is initialized, only five of these files are created. These files, which are referred to as reserved files, are as follows:

1. The index file (levels 1 and 2)
2. The storage bit map file (levels 1 and 2)
3. The bad block file (levels 1 and 2)
4. The master file directory, MFD (levels 1 and 2)
5. The core image file (levels 1 and 2)
6. The volume set list file (level 2 only)
7. The continuation file (level 2 only)
8. The backup log file (level 2 only)
9. The pending bad block log file (level 2 only)
10. The free space file (level 2 only)

All of the files listed above are cataloged in the master file directory, [0,0].

8.1.1.1 **The Index File** - Every Files-11 volume has an index file, which is created when the volume is initialized. This index file identifies the volume to the operating system as a Files-11 structure and contains the access data for all files on the volume. The index file, which is listed in the master file directory (MFD) as INDEXF.SYS;1, contains the following information:

- Bootstrap block

  The volume's bootstrap block is virtual block number 1 of the index file. If the volume is a PDP-11 system device, this block contains a bootstrap program that loads the operating system into memory. If the volume is not a system device, this block contains a PDP-11 program that displays a message that the volume is not the system device but a device that contains user files only. If the volume is a VAX/VMS system device, this block contains the same PDP-11 program that is on volumes that are not system devices. Because the VAX/VMS operating system is loaded by use of a direct file lookup, a bootstrap program is not needed.

- Home block

  The home block establishes the specific identity of the volume, providing such information as the volume name and protection, the maximum number of files allowed on the volume, and the volume ownership information. The home block is virtual block number 2 of the index file.

- Back-up home block

  The back-up home block is a second copy of the home block. It permits the volume to be used even if the primary home block is destroyed.

- Back-up index file header

  The back-up index file header permits recovery of data on the volume if the index file header goes bad.

- Index file bit map

  The index file bit map controls the allocation of file headers and thus the number of files on the volume. The bit map contains a bit for each file header that is allowed on the volume. If the value of a bit for a given file header is 0, a file can be created with this file header. If the value is 1, the file header is already in use. The index file bit map starts at virtual block number 3 of the index file and continues for the number of blocks that are necessary to contain the bit map.

- File headers

  The largest part of the index file is made up of file headers. Each file on the volume has a file header, which describes such properties of the file as file ownership, creation date and time, and file protection. The file header contains all the information needed for gaining access to the file.

8.1.1.2 **The Storage Bit Map File** - The storage bit map file controls the available space on a volume; this file is listed in the MFD as BITMAP.SYS;1. It contains a storage control block, which consists of summary information intended to optimize the Files-11 space allocation, and the bit map itself, which lists the availability of individual blocks.

8.1.1.3 **The Bad Block File** - The bad block file, which is listed in the MFD as BADBLK.SYS;1, contains a list of all the bad blocks on the volume. The system detects bad disk blocks dynamically and prevents their reuse once the files to which they are allocated have been deleted.

8.1.1.4 **The Master File Directory** - The master file directory (MFD) itself is listed in the MFD as 000000.DIR;1. The MFD, which is the root of the volume's directory structure, lists the reserved files that control the volume structure and may list both users' files and users' file directories. Usually, however, the MFD is used to list the reserved files and users' file directories; users seldom enter files in the MFD, even on private volumes. In fact, on a private volume, it is most convenient for a user to create a directory that has the same name as the user's default directory on a system disk. For an explanation of users' file directories and file specifications, see the VAX/VMS Command Language User's Guide.

8.1.1.5 **The Core Image File** - The core image file is listed in the MFD as CORIMG.SYS;1. The use of this file is operating system dependent. In general, it provides a list of certain files used by the operating system, for example, swap areas and overlay areas.

8.1.1.6 **The Volume Set List File** - The volume set list file is listed in the MFD as VOLSET.SYS;1. This file is used only on relative volume 1 of a tightly coupled volume set. The file contains a list of the labels of all the volumes in the set. This file is reserved for future use.

8.1.1.7 **The Continuation File** - The continuation file is listed in the MFD as CONTIN.SYS;1. This file is used as the extension file identifier, when a file crosses from one volume of a loosely coupled volume set to another volume. This file is reserved for future use.

8.1.1.8 **The Back-up Log File** - The back-up log file is listed in the MFD as BACKUP.SYS;1. This file contains a history of the back-ups done to the volume. This file is reserved for future use.

8.1.1.9 **The Pending Bad Block Log File** - The pending bad block log file is listed in the MFD as BADLOG.SYS;1. This file contains a list of suspected bad blocks on the volume that are not listed in the bad block file.

8.1.1.10 **The Free Space File** - The free space file is listed in the MFD as FREFIL.SYS;1. The space this file contains is available for allocation to other files; the file permits implementation of alternative schemes of space allocation. This file is reserved for future use.

8.1.2 Files-11 Structure Level 1 Versus Structure Level 2

Because Files-11 Structure Level 1 and Structure Level 2 are generally compatible, users ordinarily need not concern themselves with the format of a particular disk volume. There are some differences, however, that the system manager should keep in mind.

The main practical difference between structure level 1 volumes and structure level 2 volumes is that structure level 1 volumes are transportable to RSX-11M, RSX-11D, and IAS systems, and structure level 2 volumes are not. Because of the improved performance and reliability of structure level 2, the system manager is, as a rule, likely to initialize a public disk volume as a structure level 2 volume, unless the volume is to be transported to an RSX-11M, RSX-11D, or IAS system.

Still, the system manager or an operator may have to verify or back up structure level 1 volumes that were created on the VAX/VMS system for transport to an RSX-11M, RSX-11D, or IAS system or that were transported to the VAX/VMS system from one of those systems.

Because two levels of disk structure coexist on the VAX/VMS system, the system manager or operator must keep in mind the level of the disks and use the appropriate utilities in backing up disks and verifying them. As explained in both the VAX/VMS Operator's Guide and the VAX-11 Disk Save and Compress User's Guide, the DSC1 utility program is used to back up a disk initialized with Files-11 Structure Level 1, and the DSC2 utility program is used to back up a disk volume initialized with Files-11 Structure Level 2. Similarly, as described in the VAX/VMS Operator's Guide, the VFY1 utility program is used to verify disk volumes initialized with Files-11 Structure Level 1, and the VFY2 utility program is used to verify disk files initialized with Files-11 Structure Level 2.

Table 8-1 summarizes some of the other differences between structure level 1 and structure level 2 volumes that may be of interest to the system manager.

Table 8-1
Differences Between Files-11 Structure Level 1 and Structure Level 2 Volumes

Characteristic	Structure Level 1	Structure Level 2
Hierarchies of directories and subdirectories	No[1]	Yes
Alphabetical directory names	No[1]	Yes

(continued on next page)

[1] Nothing prevents the use of subdirectories or alphabetical directory names on structure level 1 volumes. However, RSX-11M, RSX-11D, and IAS systems cannot process subdirectories and alphabetical directory names.

Table 8-1 (Cont.)
Differences Between Files-11 Structure Level 1 and Structure Level 2 Volumes

Characteristic	Structure Level 1	Structure Level 2
Alphabetized directories	No	Yes
Clustered allocation	No	Yes
Back-up home block	No	Yes
Meaning of protection code E	Extend	Execute

8.1.3 Guidelines for Initializing Public Disk File Volumes

The following guidelines for initializing public disk file volumes supplement information presented in the VAX/VMS Command Language User's Guide.

In initializing a public disk file volume (by using the qualifier /SYSTEM), the system manager may need to use one or all of the following qualifiers of the DCL command INITIALIZE.

- /ACCESSED=n
- /CLUSTER_SIZE=n
- /EXTENSION=n
- /HEADERS=n
- /INDEX=position
- /MAXIMUM_FILES=n
- /WINDOW=n

As described below, selecting appropriate values for n and selecting the appropriate position for the /INDEX qualifier often involve making trade-offs.

8.1.3.1 **The /ACCESSED=n Qualifier** - The /ACCESSED=n qualifier provides an estimate of the number of directories expected to be in use concurrently on a volume. The file system keeps this number of directories in system space for ready access on the basis of which directories were most recently used. The result is a substantial reduction of overhead in directory operations.

8.1.3.2 **The /CLUSTER_SIZE=n Qualifier** - The /CLUSTER_SIZE=n qualifier specifies the fundamental unit of allocation (expressed in blocks) on a volume. In selecting the cluster size, wasted space at the end of files is traded off against the size of the volume storage bit map, which must contain one bit for each cluster on the volume (or one block for each 4096 clusters).

8.1.3.3 **The /EXTENSION=n Qualifier** - The /EXTENSION=n qualifier specifies the default number of blocks allocated for extending files on a volume. This value is less important on the VAX/VMS system than

on the RSX-11M, RSX-11D, and IAS systems, because VAX-11 Record Management Services use an adaptive algorithm maximized against /EXTENSION.

8.1.3.4 **The /HEADERS=n Qualifier** - The /HEADER=n qualifier specifies the number of file headers to be allocated initially to the index file. The primary advantage of preallocating file headers is that they will then be located near the storage map file (generally in the middle of the disk). This placement of file headers helps reduce head motion during file manipulation. This value should be estimated conservatively, because space once allocated to headers cannot later be made available for file storage.

8.1.3.5 **The /INDEX=position Qualifier** - The /INDEX=position qualifier specifies the location of the index file on a volume. The default position (MIDDLE) results in minimum head motion during file processing. The position BEGINNING should be used if the disk is to contain one or a few very large contiguous files.

8.1.3.6 **The /MAXIMUM_FILES=n Qualifier** - The /MAXIMUM_FILES=n qualifier specifies the maximum number of files that a volume can contain. The default value is fairly liberal; a more careful estimate of it helps optimize the dynamic allocation of the index file. Estimates should be liberal, however. Once set, the maximum number of files for a volume cannot be increased. Note that each directory and each extension header of a multiheader file counts as a file against this maximum value.

8.1.3.7 **The /WINDOW=n Qualifier** - The /WINDOW=n qualifier specifies the number of map pointers in a default file access window. This value is the number of extents of a file to which access can be gained without the cost of file system overhead.

8.2 MOUNTING PUBLIC DISK FILE VOLUMES

The purpose of mounting a disk volume is to establish a relationship between the volume, the device on which it is physically mounted, and one or more processes that may gain access to the volume.

The VAX/VMS Command Language User's Guide contains most of what the system manager needs to know about mounting volumes in 1) the description of the MOUNT command in Part II, and 2) Chapter 3, "Disk and Tape Volumes.

The following guidelines for mounting disk file volumes for public use supplement information presented in the VAX/VMS Command Language User's Guide.

In mounting a public disk file volume (by using the qualifier /SYSTEM), the system manager may need to use one or all of the qualifiers /ACCESSED, /EXTENSION, and /WINDOW (described in Section 8.1.3) or the qualifier /PROCESSOR=option (described below).

8.2.1 The /PROCESSOR=option Qualifier

The /PROCESSOR=option qualifier specifies the number of file systems to be used in controlling various public volumes. Selecting an appropriate option for the /PROCESSOR qualifier involves making a trade-off. If the system manager specifies the option SAME, file system parallelism and performance are sacrificed for the sake of saving system space. Conversely, if the system manager specifies the option UNIQUE, system space is sacrificed for the sake of file system parallelism and performance.

CHAPTER 9

BACKING UP PUBLIC FILES AND VOLUMES

To prevent the inadvertent loss or destruction of valuable information stored on disk file volumes, the system manager usually establishes a policy and a schedule for regularly backing up files on public volumes. Once such a policy is established, the system operator usually is responsible for putting it into effect. The VAX/VMS Operator's Guide therefore provides the operating procedures for backing up both selected files and entire disk volumes.

Just as preserving information on public volumes by backing it up is usually considered desirable, preserving files on private volumes is also considered desirable. However, responsibility for backing up the files on private volumes usually is left to the individual owners of those files and volumes.

There are two kinds of back-ups of public disk files and volumes: 1) selective, or partial, back-ups and 2) system, or all-inclusive, back-ups. Either type of back-up can be done either to disk or magnetic tape.

Selective back-ups of files chosen by users of the system are done by use of the VAX/VMS file-copying utility (which is called by use of the COPY command) or by use of the VAX-11 RMS utility Backup. System back-ups, on the other hand, are usually done by use of one of the Disk Save and Compress (DSC) utility programs (DSC1 or DSC2); they can also be done by use of the file-copying utility.

As explained in the VAX/VMS Operator's Guide and in the VAX-11 Disk Save and Compress User's Guide, the difference between the utilities is in the level of the Files-11 disk file structure that they write to a new disk. Thus, DSC1 (which writes disks with Files-11 Structure Level 1) is used in backing up disks that have been initialized with Files-11 Structure Level 1, whereas DSC2 (which writes disks with Files-11 Structure Level 2) is used in backing up disks that have been initialized with Files-11 Structure Level 2.

As a rule, selective back-ups are done more frequently than system back-ups. Normally, the system manager, after consulting with users of the system, decides how frequently to back up files and volumes and how long to retain back-up files and volumes.

The following schedule for backing up public disk volumes on magnetic tapes affords adequate protection of data for many installations.

- Daily, doing an all-inclusive back-up that is retained for seven days. Seven daily tapes that are rotated once a week are needed for this back-up.

- Weekly, doing an all-inclusive back-up that is retained for four weeks. Four weekly tapes that are rotated once every four weeks are needed for this back-up.

- Monthly, doing an all-inclusive back-up that is retained for a year. Twelve monthly tapes that are rotated once a year are needed for this back-up.

In spite of all precautions, there is always the risk of losing a file. Longer retention periods reduce this risk; only infinite retention of back-up files and volumes eliminates it.

CHAPTER 10

INSTALLING KNOWN IMAGES AND CREATING PERMANENT GLOBAL SECTIONS

The system manager can significantly improve system performance by installing certain executable images as known images and by creating permanent global sections. Permanent global sections are created by installing certain shareable images as known images.

There are three general reasons for installing known images:

1. To permit system-wide sharing of images that are frequently used by more than one user at a time

2. To make image files more quickly accessible

3. To permit users who normally have few, if any, privileges to execute (under suitable safeguards) executable images that require either many privileges or sensitive privileges

The INSTALL utility program, which is used in installing image files as known images and in deleting known-image files from the system's list of known images is described in the VAX/VMS Operator's Guide. Usually, the INSTALL program is run at start-up time; see Chapter 12 for an example of the use of INSTALL in a start-up command procedure.

10.1 INSTALLING EXECUTABLE IMAGES AS KNOWN IMAGES

Typically, the kinds of executable images that are installed as known images are:

1. Images that need more privileges than are commonly granted to the users who need to execute the images

2. Images that are executed frequently, often by more than one user at a time

Table 10-1 lists executable images that typically are installed as known images. The shareability and quick accessibility of these images are essential to the high performance of the system.

Table 10-1
Executable Images that Typically Are Installed as Known Images

Image File	Description	Reason File Is a Known Image
BACKTRANS.EXE	Back translator of DCL commands into MCR commands	Frequency of use
COPY.EXE	File copying utility	Frequency of use
DISMOUNT.EXE	Device dismounting utility	Need for privilege
DISPLAY.EXE	Utility used in displaying system performance statistics	Need for privilege
INIT.EXE	Files-11 Structure Level 2 device initialization utility	Need for privilege
LOGINOUT.EXE	Log-in and log-out utility	Need for privilege
PIP.EXE	RSX-11M peripheral interchange utility	Frequency of use
PRTSMB.EXE	Print symbiont	Frequency of use
RSX.EXE	Application migration executive	Frequency of use
SET.EXE	SET command processor	Frequency of use; need for privilege
SHOW.EXE	SHOW command processor	Frequency of use; need for privilege
SOS.EXE	SOS text editor	Frequency of use
TYPE.EXE	File typing utility	Frequency of use
VMOUNT.EXE	Volume mounting utility	Need for privilege

Executable images that are installed as known images can have one or more of the following characteristics.

- They can be permanently open.
- They can be shared.
- Their headers can be permanently resident in memory.
- They can be installed with privilege and may thus amplify the privileges of the users' processes in which they are executed.

10.1.1 Installing a Known Image that Is Permanently Open

If it is permanently open, a known executable image file can be located without the usual search of file directories. Hence, it can be located more quickly than an image file that is not permanently open.

The following example shows how to use the INSTALL utility program to install the executable image file PIP.EXE as a known image file that is permanently open. This file is the executable image of the RSX-11M peripheral interchange program, which all users of the VAX/VMS system frequently use.

```
$ RUN SYS$SYSTEM:INSTALL
* PIP/OPEN
```

The cost of keeping a file permanently open is about 160 (decimal) bytes of nonpaged dynamic memory.

10.1.2 Installing a Known Image that Can Be Shared

If a known executable image can be shared, several users can gain access to its read-only segments at the same time.

The following example shows how to use the INSTALL utility program to install the executable image file PRTSMB.EXE as a known image file that can be shared. Note that to be shared an executable image file must also be permanently open. The file PRTSMB.EXE is the executable image of the print symbiont, which should be shared if the system has more than one line printer.

```
$ RUN SYS$SYSTEM:INSTALL
* PRTSMB/OPEN/SHARED
```

10.1.3 Installing a Known Image with Permanently Resident Header

If the header of a known executable image remains permanently resident in memory, access to the known image can be gained with a saving of one access to disk. Hence, access to a known image with a permanently resident header can be gained more quickly than access to an image file that is not a known image.

The following example shows how to use the INSTALL utility program to install the executable image file BACKTRANS.EXE as a known image file that can be shared and whose header is permanently resident in memory. Note that for the header of an image file to be permanently resident, the image file must also be permanently open. The file BACKTRANS.EXE is the executable image of the back translator of DCL commands to MCR commands, which is used frequently and often by more than one user at a time.

```
$ RUN SYS$SYSTEM:INSTALL
* BACKTRANS/OPEN/HEADER_RESIDENT/SHARED
```

The cost of keeping an image header permanently resident is less than one page of paged dynamic memory.

10.1.4 Installing a Known Image with Privileges

If a known image is installed with privileges, it can if necessary amplify the privileges of the user's process in which it is executed. Hence, the privileges of a user are temporarily increased so that the user's process can execute a known image that requires more privileges than the user ordinarily is allowed.

The following example shows how to use the INSTALL utility program to install the executable image file LOGINOUT.EXE as a known image file. The file LOGINOUT.EXE is the executable image of the log-in and log-out utility, which needs such privileges as CMKRNL and CMEXEC to be executed. Typically, login and logout are executed by users who have these privileges only when they are executing a privileged known image.

```
$ RUN SYS$SYSTEM:INSTALL
* LOGINOUT/PRIVILEGED=(CMKRNL,CMEXEC,TMPMBX,EXQUOTA)
```

10.2 CREATING PERMANENT GLOBAL SECTIONS

Permanent global sections are shareable image files that have been installed as known images and are thus available to all users of the system. Usually, permanent global sections are created by the system manager when the system is started up, and they remain available to all users of the system until explicitly deleted (usually by the system manager). The system-wide sharing of image files by the use of permanent global sections generally tends to reduce the size of programs and thus to reduce the overall memory requirements of a system.

Both native-mode shareable image sections, produced by the VAX-11 Linker, and compatibility-mode commons and libraries, built by the RSX-11M Task Builder, can be installed as permanent global sections. The sharing of native-mode images is the topic of this section; sharing of compatibility-mode images is discussed in the VAX-11/RSX-11M Programmer's Reference Manual.

Shareable image sections produced by the VAX-11 Linker are almost identical with executable image sections, except that they cannot be executed by use of the DCL command RUN . They can, however, be linked with object modules to create executable images. How to create shareable images by use of the VAX-11 Linker and how to link these shareable images with private object modules are explained in detail in the VAX-11 Linker Reference Manual.

Permanent global sections, which are almost identical in structure with private, or process, image sections, are a special kind of shareable image section. Permanent global sections are known by all users of the system, can be referred to by all users of the system, and remain available whether they are being referred to or not. Again, an explanation of how a user links permanent global sections with private object modules is given in the VAX-11 Linker Reference Manual.

The remainder of this chapter contains:

1. An example of how permanent global sections are shared
2. A summary of the advantages of this sharing
3. An example of how the system manager can create permanent global sections from shareable images by using the INSTALL program
4. An example of how to install and use an alternate test copy of a permanent global section

10.2.1 Example of Sharing Permanent Global Sections

A typical example of a shareable image that is normally installed as a permanent global section is the VAX-11 Common Run-Time Procedure Library. The Run-Time Library can be called by users who are writing programs in either high- level languages (VAX-11 FORTRAN IV-PLUS) or assembly language (VAX-11 MACRO).

The VAX-11 Common Run-Time Procedure Library Reference Manual contains details about the contents and use of the Run-Time Library. The following example illustrates the use of this kind of shareable image.

When a VAX-11 FORTRAN IV-PLUS programmer, codes a WRITE statement, the FORTRAN compiler generates calls to the procedures in the Run-Time Library that format the output, issue an I/O request, and handle any errors. When the FORTRAN program is linked to form an executable image, the linker resolves the references to the external procedures that are part of the Run-Time Library shareable image. The linker locates the procedure names in the symbol table of the shareable image.

Once references are resolved, the linker creates the user's executable image file. The image file contains all of the code and data of the original FORTRAN program and descriptors for its various image sections. In addition to descriptors for these sections, the executable image contains descriptors for image sections of the Run-Time Library shareable image. The linker is able to copy the image section descriptors from the run-time procedures into the user's executable image without copying all the code and data of the Run-Time Library.

When the user's executable image is run, the image sections are handled so that the physical memory needed for Run-Time Library procedures is shared among all executable images that are linked to them. Thus, a shareable image needed by an executable image is brought into physical memory only if it is not already in memory. Otherwise, all executable images bound with Run-Time Library procedures that are already in memory share a single copy in memory of the library procedure.

Figure 10-1 illustrates the basic steps in creating a permanent global section and in creating and running an executable image that is bound to a permanent global section. The one step that is the special responsibility of the system manager is that of creating the permanent global section by use of the INSTALL utility program.

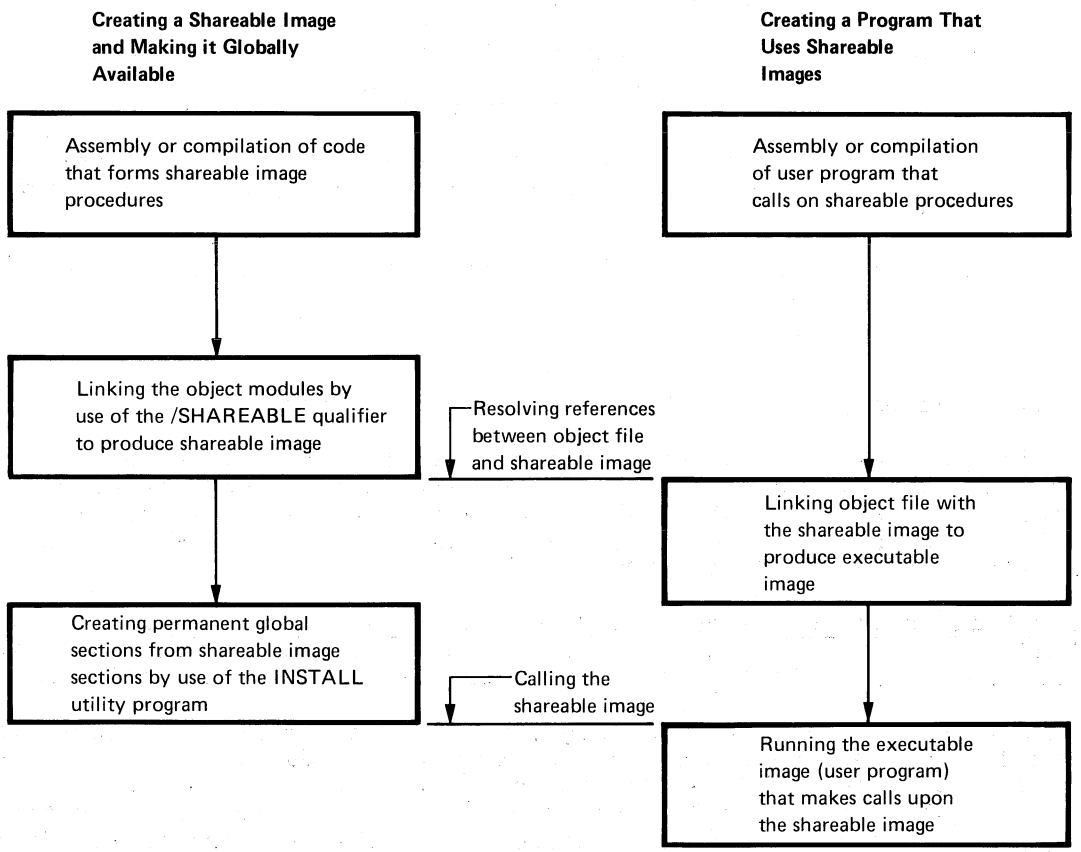


Figure 10-1 Creating and Using a Permanent Global Section

10.2.2 Advantages of Sharing Common Procedures

The sharing of common procedures described in the foregoing example leads to three significant improvements in system performance:

1. Reduction in disk storage requirements
2. Reduction in physical memory requirements
3. Reduction in the amount of paging I/O needed

Use of permanent global sections reduces the amount of disk space needed for storing executable images. When a user's object file is linked with a shareable image to produce an executable image, the linker (by default) copies only the image section descriptors from the shareable image file to the executable image file. Code and data thus remain in the shareable image file. The result is that shareable image code and data are not duplicated for each user's image that is bound to the shareable image.

The sharing of permanent global sections reduces the amount of physical memory needed for running the total number of processes in the balance set. For example, if a physical page of a global section is mapped into the working sets of six processes, five fewer pages of physical memory are needed to execute these processes. The result is that more processes can fit in physical memory at a time.

Sharing of global sections also reduces the amount of paging I/O that is typically needed during image execution. The greater the number of executable images that are mapped to a global section, the greater the chance that the pages of that section are already resident in physical memory when they are needed.

A page fault occurs when a process attempts to gain access to a virtual page that is not in its working set. When the fault occurs, the needed page is either in a disk file (in which case paging I/O is needed) or it is in physical memory (in which case no paging I/O is needed). One of the reasons that a page may be in physical memory is because it is a page of a permanent global section that is already mapped into the address space of another process. Thus, because the page already exists in memory, it does not need to be read into memory from disk.

10.2.3 Using INSTALL to Create Permanent Global Sections

The following example shows how to use the INSTALL program to install a shareable image (the Run-Time Library, SYS$LIBRARY:VMSRTL.EXE) as a permanent global section; that is, a permanently shareable image that is available to all users of the system. Note that to be permanently shareable a shareable image file must also be permanently open.

```
$ RUN SYS$SYSTEM:INSTALL
* SYS$LIBRARY:VMSRTL/OPEN/SHARED
```

10.2.4 Installing and Using a Test Copy of a Permanent Global Section

System programmers often want to test experimental versions of shareable images that have been installed as permanent global sections. This section explains how the system manager can install a test copy of such a shareable image and how programs previously linked with the original shareable image can use the test copy without having to be relinked.

For example, a system programmer may want to test an experimental version of a statistical subroutine package that was installed as a permanent global section and with which users' programs have been linked. The file specification of the original shareable image file for this statistical subroutine package is SYS$LIBRARY:STATSHR.EXE; the shareable image was installed as follows:

```
$ RUN SYS$SYSTEM:INSTALL
* SYS$LIBRARY:STATSHR/OPEN/SHARED
```

Programs linked with this shareable image are linked to SYS$LIBRARY:STATSHR.

To make a new experimental version of this statistical package available to users already linked to SYS$LIBRARY:STATSHR, the system programmer must create a new package in a shareable image file with a file specification, for example,, of SYS$LIBRARY:STATSHRX.EXE. Note that the device and directory name need not be SYS$LIBRARY, but the file name (here STATSHRX) must be different from the file name of the original file.

Then, the system manager installs this shareable image as a permanent global section as follows:

```
$ RUN SYS$SYSTEM:INSTALL
* SYS$LIBRARY:STATSHRX/OPEN/SHARED
```

Programs linked with SYS$LIBRARY:STATSHR need not be relinked with SYS$LIBRARY:STATSHRX to use the experimental version of the statistical package. Rather, a user who wants to use the new version need only issue the following DCL command:

```
$ ASSIGN STATSHRX LIB$STATSHR
```

In this way, all the user's programs that are linked with STATSHR will use STATSHRX.

CHAPTER 11

ASSIGNING SYSTEM LOGICAL NAMES

Making sure that all needed system logical names have been assigned to equivalence names is still another task of the system manager.

A logical name is a user-specified name that may be equivalent to a file specification or to some portion (such as a device name) of a file specification. A system logical name is simply a logical name that can be referred to by all users of the system and by all processes created for these users.

The use of logical names to make programs independent of specific devices is explained in Chapter 2, "File Specifications and Logical Names," of the VAX/VMS Command Language User's Guide and in Chapter 5, "Logical Names: Files for Program I/O," of the VAX/VMS Primer. The ASSIGN and DEFINE commands, which are used in establishing logical names, are fully described in the VAX/VMS Command Language User's Guide.

Some system logical names are needed by all or nearly all VAX/VMS installations; others are needed only by certain installations.

Except for such default system logical names as SYS$SYSTEM and SYS$DISK, system logical names that are needed by all or nearly all VAX/VMS installations are assigned in the start-up command procedure file STARTUP.COM, which DIGITAL provides as part of all software release distribution kits. Chapter 12 of this guide contains examples of the ASSIGN commands that are part of this start-up file.

The following paragraphs list examples of the system logical names that are established in the start-up file STARTUP.COM and the equivalence names to which they are assigned.

The VAX-11 Symbolic Debugger requires the following logical names.

Equivalence Name	Logical Name
SYS$INPUT:	DBG$INPUT:
SYS$OUTPUT:	DBG$OUTPUT:

The following logical names are needed for running FORTRAN programs.

Equivalence Name	Logical Name
SYS$INPUT:	FOR005:
SYS$OUTPUT:	FOR006:
SYS$INPUT:	FOR$ACCEPT:
SYS$INPUT:	FOR$READ:
SYS$OUTPUT:	FOR$PRINT:
SYS$OUTPUT:	FOR$TYPE:

To execute such RSX-11M compatibility mode images as BAD, SOS, PIP, F4V, and MAR, the following logical names are required.

Equivalence Name	Logical Name
'F$LOG("SYS$DISK")'	LB:
'F$LOG("SYS$DISK")'	LB0:
'F$LOG("SYS$DISK")'	WK:
'F$LOG("SYS$DISK")'	WK0:
'F$LOG("SYS$DISK")'	SP:
'F$LOG("SYS$DISK")'	SP0:

The VAX-11 language processors, the VAX-11 Linker, and the HELP command require the following logical names.

Equivalence Name	Logical Name
'F$LOG("SYS$DISK")'[SYSLIB]	SYS$LIBRARY:
'F$LOG("SYS$DISK")'[SYSHLP]	SYS$HELP:

Note that the lexical function 'F$LOG("SYS$DISK")' used above returns the name of the system device. Chapter 5 of the VAX/VMS Command Language User's Guide describes lexical functions in greater detail.

Any user who has the privilege of entering logical names into the system logical name table (the SYSNAM privilege) can assign the system logical names that are needed only by certain installations. Usually, however, the system manager is responsible for establishing the system logical names that are unique to an installation. As a rule, these names are assigned by use of ASSIGN commands in the command procedure SYSTARTUP.COM, which is called by the command procedure STARTUP.COM. DIGITAL provides the command procedure SYSTARTUP.COM as part of the software release distribution kit; as explained in Chapter 12 of this guide, the system manager places site-specific start-up commands in this file.

PART IV

OVERALL CONTROL OF THE SYSTEM

This part of the VAX/VMS System Manager's Guide covers two important ways in which the system manager exerts control over the behavior of the VAX/VMS system:

- By maintaining command procedures of initialization commands that are essential to the proper operation of the system
- By establishing output spooling and setting up and controlling batch queues, print queues, and terminal queues

The following two chapters discuss these topics:

- Chapter 12: Maintaining Start-Up Command Procedures
- Chapter 13: Spooling, Batch Queues, Print Queues, and Terminal Queues

CHAPTER 12

MAINTAINING START-UP COMMAND PROCEDURES

The software release distribution kit contains two start-up command procedures: STARTUP.COM, in directory [SYSEXE], and SYSTARTUP.COM, in directory [SYSMGR].

The first of these files (STARTUP.COM) contains site-independent initialization commands that, in general, must be executed for any VAX/VMS system to run properly. The file STARTUP.COM calls the file SYSTARTUP.COM.

Unlike STARTUP.COM, SYSTARTUP.COM is furnished as an empty file in the software release distribution kit. Although the system manager can tailor STARTUP.COM to the needs of a particular installation, it is not customary to do so. Instead, the system manager usually places site-specific initialization commands in SYSTARTUP.COM. The system manager should, nevertheless, be acquainted with the contents of STARTUP.COM as he or she may sometimes need to alter them.

12.1 THE SITE-INDEPENDENT START-UP FILE STARTUP.COM

The command procedure STARTUP.COM is a start-up file that is automatically executed immediately after the VAX/VMS operating system has been booted. This start-up file contains commands for performing site-independent operations that must be performed if the system is to run properly. These operations include assigning system logical names (see Chapter 11), installing executable images as known images (see Chapter 10), creating permanent global sections (see Chapter 10), and building the I/O data base and loading I/O drivers.

The commands and functions described in the following sections are typical of those found in STARTUP.COM. The order in which these commands and functions are presented is also typical.

12.1.1 Housekeeping Functions

Three commands are used in performing miscellaneous housekeeping functions:

```
$ SHOW TIME
$ SET VERIFY
$ SET NOON
```

The first command, SHOW TIME, displays the current time on the console terminal. The second, SET VERIFY, causes commands that are found in indirect command files to be echoed (typed) on the terminal. The last command, SET NOON, sets up condition handling so that the command file continues processing even if an error occurs.

12.1.2 Mounting the Floppy Disk

A floppy disk, which is mounted as a foreign volume, drives the console program. As described in the VAX-11 Software Installation Guide, the system manager can use this floppy disk in updating the bootstrap and bootstrap procedures. The following command is used in mounting the floppy disk:

```
$ MOUNT/SYSTEM/FOR/PROT=(SYSTEM:RWLP) DX1:  CONSOLE
```

The /SYSTEM and /PROT qualifiers protect the floppy disk from access by unprivileged users. Generally, only the system manager needs to write to it.

12.1.3 Setting Default Directory Name

The following command is used in setting the default directory to [SYSEXE], which is the location of the executable images of the VAX/VMS operating system and utilities.

```
$ SET DEF [SYSEXE]
```

12.1.4 Assigning Logical Names

The VAX-11 Symbolic Debugger needs two system logical names for normal input and output functions. The following two commands establish these logical names.

```
$ ASSIGN/SYSTEM SYS$INPUT: DBG$INPUT:
$ ASSIGN/SYSTEM SYS$OUTPUT: DBG$OUTPUT:
```

Systems that run FORTRAN programs need a series of system logical names that are assigned as follows.

```
$ ASSIGN/SYSTEM SYS$INPUT: FOR005:
$ ASSIGN/SYSTEM SYS$OUTPUT: FOR006:
$ ASSIGN/SYSTEM SYS$INPUT: FOR$ACCEPT:
$ ASSIGN/SYSTEM SYS$INPUT: FOR$READ:
$ ASSIGN/SYSTEM SYS$OUTPUT: FOR$PRINT:
$ ASSIGN/SYSTEM SYS$OUTPUT: FOR$TYPE:
```

All sites that run any RSX-11M compatibility mode images (such as BAD, SOS, PIP, F4V, or MAR) need certain system logical names that are assigned as follows.

```
$ ASSIGN/SYSTEM 'F$LOG("SYS$DISK")' LB:
$ ASSIGN/SYSTEM 'F$LOG("SYS$DISK")' LB0:
$ ASSIGN/SYSTEM 'F$LOG("SYS$DISK")' WK:
$ ASSIGN/SYSTEN 'F$LOG("SYS$DISK")' WK0:
$ ASSIGN/SYSTEM 'F$LOG("SYS$DISK")' SP:
$ ASSIGN/SYSTEM 'F$LOG("SYS$DISK")' SP0:
```

Logical names that are used by the VAX-11 language processors, by the VAX-11 Linker, and by the HELP command are assigned as follows.

```
$ ASSIGN/SYSTEM 'F$LOG("SYS$DISK")'[SYSLIB] SYS$LIBRARY:
$ ASSIGN/SYSTEM 'F$LOG("SYS$DISK")'[SYSHLP] SYS$HELP:
```

12.1.5 Installing Known Images and Creating Permanent Global Sections

Images that must be privileged in order to run, images whose quick accessibility is critical to system performance, and images that should be permanently sharable system wide are installed as known images and as permanent global sections by use of the INSTALL program, which is usually run at start-up time. The following example illustrates the use of the INSTALL program in the start-up file.

```
$ RUN SYS$SYSTEM:INSTALL
PRTSMB /OPEN /SHARED
INIT /PRIVILEGED=(CMKRNL,PHY_IO,LOG_IO)
VMOUNT /PRIVILEGED=(CMKRNL,DETACH,LOG_IO,SETPRV,ALTPRI,-
        TMPMBX,WORLD,GROUP,EXQUOTA,ACNT,PHY_IO,BUGCHK,MOUNT)
DISMOUNT /PRIVILEGED=(CMKRNL,EXQUOTA,BUGCHK)
DISPLAY /PRIVILEGED=(CMKRNL,CMEXEC) /OPEN /HEADER_RESIDENT
 /SHARED
LOGINOUT /PRIVILEGED=(CMKRNL,CMEXEC, TMPMBX,EXQUOTA)
SET /PRIVILEGED=(CMKRNL,TMPMBX) /OPEN /HEADER_RESIDENT /SHARED
SHOW /PRIVILEGED=(CMKRNL) /OPEN /HEADER_RESIDENT /SHARED
PIP /OPEN
SOS /OPEN /SHARED
SYS$LIBRARY:RSXSHR /SHARED /OPEN
RSX /OPEN /HEADER_RESIDENT /SHARED
BACKTRANS /OPEN /HEADER_RESIDENT /SHARED
SYS$LIBRARY:DEBUG /OPEN /SHARED
SYS$LIBRARY:TRACE /OPEN /SHARED
SYS$LIBRARY:VMSRTL /SHARED
```

12.1.6 Building the I/O Data Base and Loading I/O Drivers

The SYSGEN program is used to automatically build an I/O data base and to load the I/O drivers needed by the standard I/O devices on the system. The following example illustrates the use of the SYSGEN program in the start-up file.

```
$ RUN SYS$SYSTEM:SYSGEN
AUTOCONFIGURE ALL
```

The SYSGEN utility program is described in the VAX-11 Software Installation Guide.

12.1.7 Calling a Site-Specific Start-Up File

The following command is used in calling the site-specific start-up file.

```
$ @[SYSMGR]SYSTARTUP.COM
```

12.1.8 Logging Out

The final command in STARTUP.COM is the following log-out command.

```
$ LOG
```

12.2 THE SITE-SPECIFIC START-UP FILE SYSTARTUP.COM

The command procedure SYSTARTUP.COM is a start-up file that the system manager tailors to the needs of a specific installation. Because this file is furnished empty in the software release distribution kit, the version of SYSTARTUP.COM that is described in the following paragraphs is (though typical) only a model.

Typically, this start-up file contains commands for performing such operations as: mounting system disks (see Chapter 8), assigning logical names (see Chapter 11), establishing and starting queues (see Chapter 13), establishing spooled printers (see Chapter 13), installing known images and creating permanent global sections (see Chapter 10), setting the characteristics of terminals and other devices, purging the operator's log file (see Chapter 15), submitting batch jobs that are run at the time the system is initialized and that are periodically resubmitted, and announcing that the system is up and running.

12.2.1 Mounting System Disks

The following commands are typical of those used in mounting system disks.

```
$ MOUNT /PROCESS = UNIQUE /SYSTEM DBA1: BUILDDATA
$ MOUNT /PROCESS = UNIQUE /SYSTEM DBA2 BILLING
$ MOUNT /SYSTEM DBB2: FORTRAN
```

12.2.2 Initializing and Starting Queues

Usually, at start-up time, batch queues and output queues are initialized and started. Thereafter when the system is rebooted, each queue often needs only to be started again. A sample of the commands that should be used in the start-up file to initialize and to start queues follows. Note that only if an attempt to start a queue fails (if $STATUS is not true) is that queue initialized and then started.

```
$ QLPA0:        START /QUEUE LPA0
$               IF $STATUS THEN GOTO QLPB0
$               INITIALIZE /QUEUE /FLAG LPA0
$               START /QUEUE LPA0
$ QLPB0:        START /QUEUE LPB0
$               IF $STATUS THEN GOTO QLPC0
$               INITIALIZE /QUEUE /FLAG LPB0
$               START /QUEUE LPB0
$ QLPC0:        START /QUEUE LPC0
```

12.2.3 Installing Known Images and Creating Permanent Global Sections

The system manager often installs a number of utility programs as known images or permanent global sections, so that they can be located quickly or shared. These programs are installed as follows.

```
$ RUN SYS$SYSTEM:INSTALL
COPY /OPEN
MAR /OPEN
LIB /OPEN
LINK /OPEN
```

12.2.4 Setting the Characteristics of Terminals and Other Devices

The system manager uses a series of SET commands to establish the characteristics of the terminals and other devices on the system, as follows.

```
$ SET TERMINAL TTC2: /SPEED=300/LA36/PERM
$ SET TERMINAL TTDl: /SPEED=9600/PREM
$ SET TERMINAL TTD4: /SPEED=1200/PERM
$ SET PRINTER LPA0: /LOWER/NOCR
$ SET DEVICE LPA0: /SPOOLED
```

Note that the /SPEED qualifier sets both transmit and receive speeds to the same value.

12.2.5 Purging the Operator's Log File

Every time the system is booted, it is convenient to purge all but the last two or three versions of the operator's log file, as follows.

```
$ PURGE /KEEP=2 [SYSMGR]OPERATOR.LOG
```

12.2.6 Submitting Standard Batch Jobs

Some sites may have batch jobs that are submitted at system start-up time and that resubmit themselves to run at intervals as long as the system is running. For such jobs, the SUBMIT command is used in the start-up file as follows.

```
$ SUBMIT SYS$SYSTEM:LOGJOBS
```

12.2.7 Announcing that the System Is Up and Running

The last command in SYSTARTUP.COM typically announces to all terminals that the system is up and running.

```
$ REPLY /ALL /BELL "VAX/VMS System Initialized"
```

CHAPTER 13

SPOOLING, BATCH QUEUES, PRINT QUEUES, AND TERMINAL QUEUES

System performance is greatly influenced by how well the system manager establishes spooled devices and creates and controls queues. Typically, the system manager of a VAX/VMS operating system is responsible for performing the following four closely related functions:

1. Establishing spooling of input and output. The VAX/VMS operating system supports input spooling of batch job files from card readers and transparent spooling of output files for line printers and terminals. Using DCL commands, the system manager specifies which output devices are to be spooled. Section 13.1 describes spooling and the use of DCL commands to establish spooled devices.

2. Creating and controlling batch queues. Section 13.2 describes batch processing and the use of DCL commands to create and control batch queues.

3. Creating and controlling print queues. Section 13.3 describes queuing output to line printers and the use of DCL commands to create and control print queues.

4. Creating and controlling terminal queues; see Section 13.4.

The system manager need not learn all the inner workings of spooling and queuing. However, a working knowledge of how to establish spooled devices and how to create and control queues is essential for the manager to keep the system running efficiently. For this reason, Sections 13.5 and 13.6 and Figures 13-1 through 13-4 contain guidelines for setting up batch queues, print queues, and spooled line printers.

The kind of working knowledge that the system manager needs presupposes a familiarity with the DCL commands listed in Table 13-1. The use of these commands is restricted to users who have operator privilege (OPER), typically, the system manager and system operators. The VAX/VMS Operator's Guide fully describes these commands.

In addition, three other VAX/VMS commands play a role in the control of batch and output queues:

1. SHOW QUEUE -- displays information about a file (or files) queued for batch execution or for output. No privilege is needed to use this command.

2. SET QUEUE -- changes the attributes of a file (or files) queued for batch execution or for output.

Ordinarily, no privilege is needed to use this command; operator privilege (OPER) is needed, however, to use the command to:

a. Modify queued elements entered by a member of another group

b. Increase the queue priority of a job

3. DELETE/ENTRY -- deletes jobs from queues.

   Ordinarily, no privilege is needed to use this command; operator privilege (OPER) is needed, however, to use this command to delete a queued job entered by a member of another group.

These commands are described fully in the VAX/VMS Command Language User's Guide.

Table 13-1
Operator Commands Used in Regulating Spooling and Queuing

Command	Use
SET DEVICE/SPOOLED	To establish spooled printers or terminals and to assign queues to them
SET DEVICE/NOSPOOLED	To turn off spooling of printers or terminals
INITIALIZE/QUEUE	To create queues
DELETE/QUEUE	To delete queues
START/QUEUE	To start queues
STOP/QUEUE	To stop queues
ASSIGN/QUEUE	To assign a queue to a device
DEASSIGN/QUEUE	To deassign a queue from a device
ASSIGN/MERGE	To empty a queue of jobs and to place them in another queue
STOP/ABORT	To abort printing of a file that is currently being printed
STOP/REQUEUE	To stop the printing of a job that is currently being printed and to requeue that job at the end of the queue

13.1 SPOOLING

Spooling is the technique of using a high-speed mass storage device to buffer data passing between low-speed I/O devices and high-speed main memory. The low-speed devices, which can be either the ultimate sources or the ultimate destinations of buffered I/O data, are called spooled devices; the high-speed mass storage devices are called intermediate devices.

The system manager establishes the spooled devices; to all other users, and their programs, the mechanism of spooling is transparent.

There are two kinds of spooling: input spooling and output spooling.

Input spooling makes input from a low-speed I/O device (such as a card reader) available for processing by placing it into a file on a high-speed device (such as a disk). Input spooling is used principally to create, from card reader input, batch input files on disk. After they are spooled to disk, batch jobs are queued for processing according to their priority.

Output spooling, on the other hand, makes output from the processor available for transmission to a low-speed I/O device (such as a line printer) by placing it into files on a high-speed device (such as a disk). Output spooling is used principally to create printer output files on disk. After they are spooled to disk, print jobs are queued for printing according to their priority.

As a rule, programs demand input and produce output at irregular intervals during their execution. If programs were allowed to read directly from low-speed devices and to write directly to low-speed devices, two undesirable results would occur:

1. Utilization of low-speed devices would be impaired. For example, if two users wanted to use a low-speed device, such as a line printer, at the same time, one would have to wait until the device was free, even though there were substantial periods in which no I/O operation was actually taking place.

2. Valuable system resources (such as balance set slots, physical memory, and swapping space) would be tied up unnecessarily as processes competing for the use of the same I/O device became bogged down in long I/O waiting periods.

Spooling alleviates these undesirable results: inputs from low-speed devices and outputs to low-speed devices are stored temporarily (buffered) on high-speed mass storage devices.

Then, when needed for processing, an input file is available directly from a high-speed intermediate device. Likewise, an output file is produced directly on a high-speed intermediate device. In this way, the I/O waiting time of a process is reduced, and system throughput is improved.

The actual transfer of inputs from a spooled device to an intermediate device or the transfer of outputs from an intermediate device to a spooled device is carried out by processes called symbionts.

Input symbionts read input at the speed of the input spooled device and buffer it in a file on the intermediate device. Later, when the input is needed, it is read directly from the file on the intermediate device rather than from the spooled device. While one set of input data is being processed, the input symbiont is free to read another set of input data into another file on the intermediate device. The use of the low-speed input device is thus optimized.

Output symbionts read data from an intermediate device and write the data to an output spooled device at the speed of that output device. The data on the intermediate device are generated by programs that produce outputs directly into files on the intermediate device. The I/O waiting time of programs is thus minimized. When an output file is complete, it is queued for printing by an output symbiont. As with input symbionts, there is an overlapping here too. While an output symbiont is printing a file stored temporarily on an intermediate device, another program can be producing another output file on the intermediate device. Again, the use of the low-speed output device is optimized.

13.1.1 Establishing Spooled Devices

Card readers are spooled by default. To use a card reader without spooling, users must allocate the reader before making it ready to read a card deck. By default, also, the queue SYS$BATCH is assigned to spooled input devices. Thus, no special command is needed to establish card readers as spooled devices.

On the other hand, the operator command SET DEVICE must be used to establish a line printer or a terminal as a spooled device. The use of this command is restricted to users with the OPER privilege to execute operator functions. As a rule, only the system manager and the system operators have this privilege. The VAX/VMS Operator's Guide describes the SET DEVICE command in detail.

Typically, the system manager must decide which low-speed peripheral devices to include in the system's basic complement of spooled devices. Often, the system manager sets up devices for spooling by making entries in the system start-up command procedure.

At a minimum, the system manager should see that at least one line printer is set spooled when the system is started up. In a system with only one line printer, this is the default system printer. The system manager need not set a card reader spooled, because card readers are spooled by default.

Depending on system configuration and anticipated operational needs, more spooled devices can be established at start-up. Moreover, in the course of normal operations (to meet special operational needs) the manager or the operator can define still other devices as spooled devices, without having to reboot the system. Normally, all line printers should be spooled.

Finally, and most important, on a system with both spooled input devices and spooled output devices, the system manager must create and start at least one batch queue to handle spooled input and one output queue to handle output for each spooled output device.

13.1.2 Turning Off Spooling

The system manager or operator can, as necessary, turn off spooling to spooled printers and terminals by use of the SET DEVICE command.

13.2 BATCH QUEUES

Batch jobs can enter the VAX/VMS system and be queued for initiation in two ways:

1. As batch job files submitted by use of the $JOB command (see the VAX/VMS Command Language User's Guide) from a card reader. These batch job files are spooled onto disk by an input symbiont and placed in a batch queue according to their priority. Unless the $JOB card specifies otherwise, the name of this batch queue is SYS$BATCH (by default). From the batch queue, batch jobs are selected for execution.

2. As command procedure disk files submitted by use of the SUBMIT command (see the VAX/VMS Command Language User's Guide). These files are also placed in a batch queue and selected for execution according to their priority. Again, by default, the name of this batch queue is SYS$BATCH.

Batch jobs cannot be executed unless at least one batch queue has been created on the system and unless that queue has been started. By default, this is the batch queue SYS$BATCH.

In the VAX/VMS system, many jobs, or streams, can be executed at the same time from each of several batch queues. Thus, the system manager can create and start several batch queues at once and can specify the number of jobs, or streams, that can be executed at the same time from each queue.

Among the jobs in a batch queue that has been started, the one with the highest priority is the first candidate for initiation. Whether or not that job is actually started up, however, depends on an evaluation of the following limits and conditions:

- The maximum number of batch jobs allowed to be executed from the queue at the same time. The system manager specifies this limit with either the INITIALIZE/QUEUE command or the START/QUEUE command.

- The maximum number of all jobs allowed to be executed in the system at the same time.

- The number of jobs currently being executed in the system.

Hence, the highest priority batch job in a queue is started up only if both of the following conditions are satisfied:

- Fewer than the maximum number of batch jobs allowed are currently running from the queue.

- The system is not saturated with other jobs.

13.2.1 Creating Batch Queues

The operator command INITIALIZE/QUEUE is used in creating, or initializing, a batch queue. The use of this command is restricted to users with the OPER privilege to execute operator functions. As a rule, only the system manager and the system operators have this privilege. The VAX/VMS Operator's Guide describes the INITIALIZE/QUEUE command in detail.

Typically, the system manager must decide on the number of batch queues for an installation, on the job limit of each queue, on the priority of each queue, and on the swap mode of each queue. Often, the system manager creates batch queues by making entries in the system start-up command procedure.

Setting up batch queues is not restricted to start-up time. In the course of normal operations, the system manager or operator can create batch queues as operational needs dictate.

13.2.2 Deleting Batch Queues

The system manager or operator can delete batch queues, as necessary, by use of the DELETE/QUEUE command. The VAX/VMS Operator's Guide describes this command in detail.

13.2.3 Starting Batch Queues

The initiation of batch jobs from a batch queue (dequeuing) can only take place if the batch queue has been started. The operator command START/QUEUE starts a batch queue. The use of this command is restricted to users with the OPER privilege to execute operator functions. As a rule, only the system manager and the system operators have this privilege. The VAX/VMS Operator's Guide describes the START/QUEUE command in detail.

Typically, the system manager must see that batch queues created by use of the INITIALIZE/QUEUE command are started. Often, the system manager starts batch queues by making entries in the system start-up command procedure.

Starting batch queues is not restricted to start-up time. In the course of normal operation, the system manager or operator can start queues as operational needs dictate.

13.2.4 Stopping Batch Queues

The system manager or operator can, as necessary, abort a job in a batch queue or disable all processing from the queue until the queue is restarted by use of the START/QUEUE command. The STOP/QUEUE command is used to stop batch queues. The VAX/VMS Operator's Guide describes this command in detail.

13.3 PRINT QUEUES

Unless a line printer is associated with a physical queue (a queue that has the same name as the line printer) and unless that queue has been started, no queued output can occur on that line printer.

Print jobs are queued for processing by an output symbiont in one of two ways: without the direct intervention of a user (that is, implicitly) or with the direct intervention of a user (that is, explicitly).

When an implicitly spooled print file destined for a spooled printer is closed, the file is placed in a print queue. Both the spooling of

the output file to an intermediate device and the subsequent queuing of a job consisting of this file occur without the direct intervention of a user.

By use of the PRINT command, a user can explicitly queue a disk file or several files for printing. The VAX/VMS Command Language User's Guide describes the PRINT command in detail. The disk file or files specified in the PRINT command are queued as a print job; if several files make up a print job, they will be printed together.

Print jobs are placed in queues according to their priority. These queues can be any one of the following:

- Physical queues -- queues associated with (that is, named for) a specific line printer.
- Generic queues -- queues from which files can be printed out on any available line printer that has correctly matching characteristics.
- Named, or logical, queues -- queues that are not associated, even indirectly, with any device. To obtain printed output from a named, or logical, queue, the system manager or operator must explicitly assign the queue to a printer. The command ASSIGN/QUEUE is used for this purpose.

From these queues, print jobs are selected for initiation. Among the jobs in a print queue for a particular printer at any given time, the job with the highest priority is the one chosen for printing.

By default, print jobs queued by use of the PRINT command are placed in the queue named SYS$PRINT. Thus, to use the default version of the PRINT command in a system with only one line printer, the name SYS$PRINT is equated with the name of the physical line printer. To use the default version of the PRINT command in a system with several line printers of matching characteristics, SYS$PRINT is normally established as the name of a generic queue.

13.3.1 Creating Print Queues

The operator command INITIALIZE/QUEUE is used in creating, or initializing, a print queue. The use of this command is restricted to users with the OPER privilege to execute operator functions. As a rule, only the system manager and the system operators have this privilege. The VAX/VMS Operator's Guide describes the INITIALIZE/QUEUE command in detail.

Typically, the system manager must decide on the number of print queues for an installation and on their attributes. Often, the system manager creates print queues by making entries in the system start-up command procedure. Chapter 12 provides examples of such entries.

Setting up print queues is not restricted to start-up time. In the course of normal operations, the system manager or operator can create print queues as operational needs dictate.

13.3.2 Deleting Print Queues

The system manager or operator can delete print queues, as necessary, by use of the DELETE/QUEUE command. The VAX/VMS Operator's Guide describes this command in detail.

13.3.3 Starting Print Queues

The initiation of print jobs from a print queue (dequeuing) can only take place if the print queue has been started. The operator command START/QUEUE starts a print queue. The use of this command is restricted to users who have the OPER privilege to execute operator functions. As a rule, only the system manager and system operators have this privilege. The VAX/VMS Operator's Guide describes the START/QUEUE command in detail. All options that can be specified in the INITIALIZE/QUEUE command can also be specified in the START/QUEUE command.

Typically, the system manager must see that print queues created by use of the INITIALIZE/QUEUE command are started. Often, the system manager starts print queues by making entries in the system start-up command procedure. Chapter 12 provides examples of such entries.

Starting print queues is not restricted to start-up time. In the course of normal operations, the system manager or operator can start queues as operational needs dictate.

13.3.4 Stopping Print Queues

The system manager or operator can abort a job in a print queue, suspend the printing of a job currently being printed, or disable processing from the queue entirely until the queue is restarted by use of the START/QUEUE command. The STOP/QUEUE command is used to stop print queues and to suspend printing of jobs. The VAX/VMS Operator's Guide describes this command in detail.

13.3.5 Assigning a Named, or Logical, Print Queue to a Printer

The operator command ASSIGN/QUEUE is used in assigning, or redirecting, a named, or logical, print queue to a printer. The use of this command is restricted to authorized users with the OPER privilege to execute operator functions. As a rule, only the system manager and system operators have this privilege. The VAX/VMS Operator's Guide describes the ASSIGN/QUEUE command in detail.

To produce printer output, a logical queue must first be assigned to a printer and then started.

Typically, the print files of a group of low-priority users can be placed in a logical queue and held there until off-peak hours. Then, to print the files, the system operator can assign the queue to a line printer and start the queue.

13.3.6 Deassigning a Named, or Logical, Print Queue from a Printer

The operator command DEASSIGN/QUEUE is used in deassigning a named, or logical, print queue from a printer. The use of this command is restricted to authorized users with the OPER privilege to execute operator functions. As a rule, only the system manager and system operators have this privilege. The VAX/VMS Operator's Guide describes the DEASSIGN/QUEUE command in detail.

13.4 TERMINAL QUEUES

Terminal queues are created and controlled by use of the same commands that are used in creating and controlling print queues. For details, see the VAX/VMS Operator's Guide.

13.5 GUIDES TO SETTING UP BATCH QUEUES

The following rules of thumb are useful in setting up a batch queue for a system that is predominantly interactive:

1. Set up one batch queue named SYS$BATCH, the name of the default batch queue.
2. Give SYS$BATCH the following characteristics:
   a. Job limit -- 2 to 4 (2, for example)
   b. Priority -- 3 or 4 (3, for example)
   c. Swapping mode -- swapping enabled (by default)

The system manager issues the following commands to create and start this queue:

```
$ INITIALIZE/QUEUE/BATCH/JOB_LIMIT=2/PRIORITY=3  SYS$BATCH
$ START/QUEUE  SYS$BATCH
```

Normally, these commands are contained in the start-up command procedure (see Chapter 12).

The following rules of thumb are useful in setting up batch queues for a system that is predominantly a batch system and in which editing is the principal interactive activity.

1. Set up three batch queues as follows:
   a. SYS$BATCH -- the default batch queue
   b. FAST -- a high-priority queue for executing high-priority jobs that should not be swapped out of memory
   c. SLOW -- a low-priority background queue for processing low-priority jobs. Typically, these are large jobs with large requirements for physical memory. Usually, it is uneconomical to swap such jobs out of memory. Normally, the system operator adjusts the system workload by stopping and restarting background queues as needed.

2. Give SYS$BATCH the following characteristics:

   a. Job limit -- 6 to 10 (6, for example)

   b. Priority -- 4 (by default)

   c. Swapping mode -- swapping enabled (by default)

3. Give FAST the following characteristics:

   a. Job limit -- 1 (by default)

   b. Priority -- high (10, for example)

   c. Swapping mode -- swapping disabled

4. Give SLOW the following characteristics:

   a. Job limit -- 1 (by default)

   b. Priority -- low (2, for example)

   c. Swapping mode -- swapping disabled

The system manager issues the following commands to create and start these queues:

```
$ INITIALIZE/QUEUE/BATCH/JOB_LIMIT=6  SYS$BATCH
$ START/QUEUE  SYS$BATCH
$ INITIALIZE/QUEUE/BATCH/PRIORITY=10/DISABLE_SWAPPING  FAST
$ START/QUEUE  FAST
$ INITIALIZE/QUEUE/BATCH/PRIORITY=2/DISABLE_SWAPPING  SLOW
$ START/QUEUE  SLOW
```

Normally, these commands are contained in the start-up command procedure (see Chapter 12).

13.6 GUIDES TO SETTING UP PRINT QUEUES AND SPOOLED LINE PRINTERS

The following rules of thumb are useful in setting up and regulating print queues and spooled line printers.

1. Normally, set all line printers spooled.

2. To produce output on a spooled line printer, initialize a print queue with the same name as the spooled printer and start that queue.

3. If more than one line printer is on the system, enable generic printing from as many print queues as possible, and make at least one print queue (SYS$PRINT) a generic queue. Queues for line printers that have unique characteristics or that are in remote locations should not be enabled for generic printing.

4. For special printing jobs (to print checks, for example) that are not normally done on a spooled printer, stop the queue associated with the spooled printer on which the special printing is to be done, allocate the spooled printer (this takes the ALLSPOOL privilege), and print the special job. After the job is completed, deallocate the spooled printer and restart the queue.

Figures 13-1 through 13-4 illustrate some of the most common arrangements of spooled line printers and print queues. These figures can be used, with the rules of thumb listed above, as guidelines in setting up spooled line printers and print queues.

Figure 13-1 illustrates a typical spooling and queuing configuration for a system with one line printer. The commands listed in this figure produce the following results:

1. The line printer LPA0 is set spooled.
2. System wide, the logical name SYS$PRINT is equated with the name LPA0. The equivalence of these names is recorded in the system logical name table.
3. The print queue LPA0 is initialized and started.
4. All print jobs explicitly directed to the printer LPA0 are placed in the queue LPA0 and are printed from that queue.
5. All print jobs that normally would be placed by default in a queue named SYS$PRINT (if that queue existed) are actually placed in the queue LPA0 (in this case, the system default print queue) and are printed from that queue.

COMMANDS

```
$ SET DEVICE/SPOOLED=LPA0 LPA0
$ ASSIGN/SYSTEM LPA0  SYS$PRINT
$ INITIALIZE/QUEUE/FLAG  LPA0
$ START/QUEUE  LPA0
```

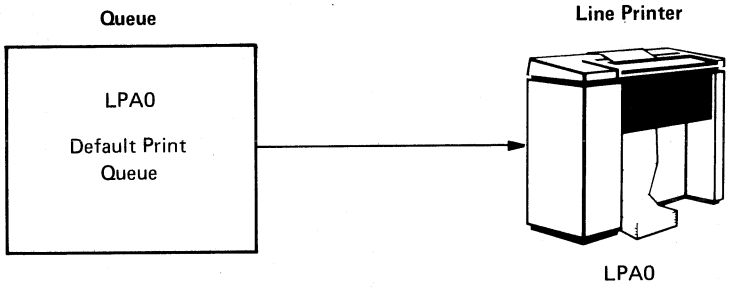


Figure 13-1 Setting Up a Spooled Printer and a Print Queue on a System with One Line Printer

Figure 13-2 illustrates a typical spooling and queuing configuration for a system with two line printers that have the same characteristics. The commands listed in this figure produce the following results:

1. The line printer LPA0 is set spooled.
2. The line printer LPB0 is set spooled.
3. The generic queue SYS$PRINT is initialized and started.
4. Physical queues LPA0 and LPB0 are initialized and started, with generic printing enabled by default.

5. All print jobs explicitly directed by use of the PRINT command to one of the two printers are placed in the queue associated with the specified printer.

6. Print jobs queued by use of the PRINT command without device specification are placed by default in the generic queue SYS$PRINT. From the generic queue, jobs are printed on whichever printer is free, by way of either of the two physical queues, LPA0 or LPB0.

7. Spooled print files destined either for LPA0 or for LPB0 are placed in the generic queue SYS$PRINT, which was associated with both these printers. From the generic queue, these jobs are printed on whichever printer is free.

COMMANDS

```
$ SET DEVICE/SPOOLED LPA0
$ SET DEVICE/SPOOLED LPB0
$ INITIALIZE/QUEUE/FLAG/GENERIC SYS$PRINT
$ START/QUEUE SYS$PRINT
$ INITIALIZE/QUEUE/FLAG LPA0
$ START/QUEUE LPA0
$ INITIALIZE/QUEUE/FLAG LPB0
$ START/QUEUE LPB0
```

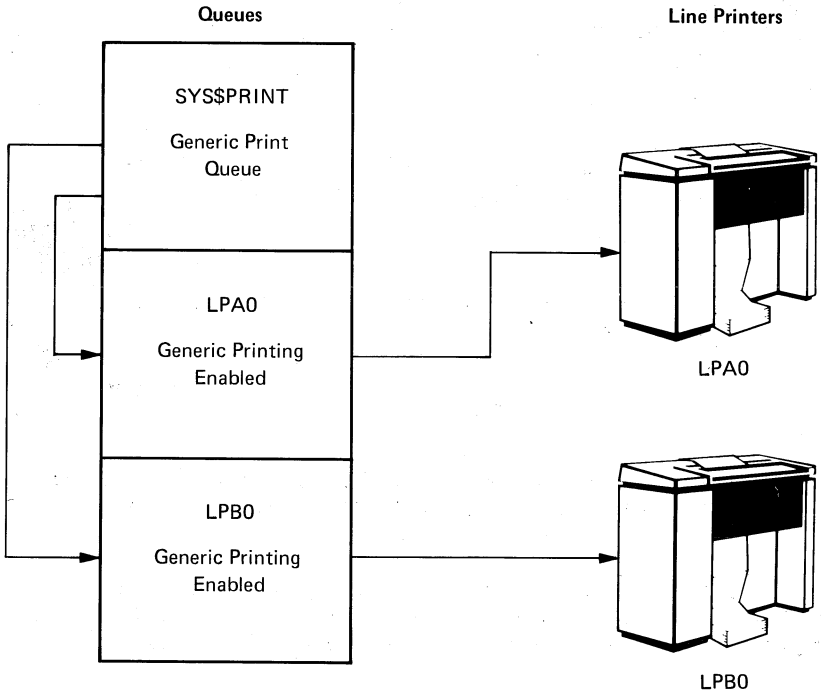


Figure 13-2 Setting Up Spooled Printers and Print Queues on a System with Two Line Printers with the Same Characteristics

Figure 13-3 illustrates a typical spooling and queuing configuration for a system with three line printers: two that have the same characteristics and one that either has special characteristics or is in a remote location. The configuration shown in Figure 13-3 is basically the same as the one in Figure 13-2, with the addition of the spooled printer LPC0 and the creation and starting of the queue LPC0. Because of some special characteristics or because of its remote location, printer LPC0 is not suited for general printing.

COMMANDS

```
$ SET DEVICE/SPOOLED LPA0
$ SET DEVICE/SPOOLED LPB0
$ SET DEVICE/SPOOLED=LPC0 LPC0
$ INITIALIZE/QUEUE/FLAG/GENERIC SYS$PRINT
$ START/QUEUE SYS$PRINT
$ INITIALIZE/QUEUE/FLAG LPA0
$ START/QUEUE LPA0
$ INITIALIZE/QUEUE/FLAG LPB0
$ START/QUEUE LPB0
$ INITIALIZE/QUEUE/FLAG/NOENABLE_GENERIC_PRINTING LPC0
$ START/QUEUE LPC0
```

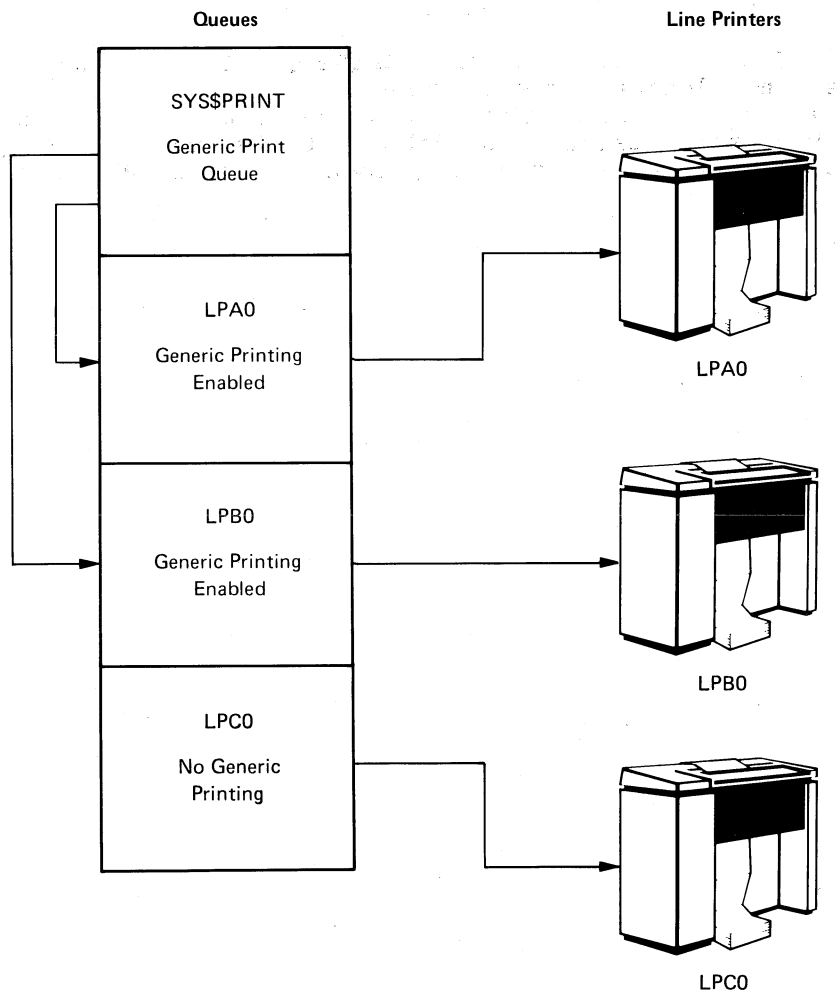


Figure 13-3 Setting Up Spooled Printers and Print Queues on a System with Three Line Printers; Two with the Same Characteristics and One With Special Characteristics or in a Remote Location

Thus, with the following exceptions, the commands listed in Figure 13-3 produce the same results as the commands listed in Figure 13-2.

1. The line printer LPC0 is set spooled.

2. The physical queue LPC0 is initialized and started with generic printing disabled.

3. Only print jobs explicitly directed to the printer LPC0 are ever placed in the queue LPC0; no generic printing is ever done on printer LPC0 by way of the queue LPC0.

Figure 13-4 adds still another feature to the configuration illustrated in Figure 13-3. This is a logical queue, which is a named, nongeneric queue that is not directly associated with any line printer. When a logical queue is assigned to a line printer and started, however, output to a line printer can occur.

Logical queues can be used, for example, to hold print jobs of low-priority users for printing during off-peak hours. To channel the print jobs of these users into a logical queue, the name of the logical queue (HOLD, for example) must be assigned to the name of their default print queue (SYS$PRINT).

As shown in Figure 13-4, the INITIALIZE/QUEUE command is used to initialize the logical queue HOLD. This queue is initialized with generic printing disabled. When the queue HOLD is assigned to the printer LPB0 and started, the jobs in the queue HOLD are printed on the line printer LPB0 by way of the physical queue LPB0.

COMMANDS

```
$ SET DEVICE/SPOOLED  LPA0
$ SET DEVICE/SPOOLED  LPB0
$ SET DEVICE/SPOOLED=LPC0  LPC0
$ INITIALIZE/QUEUE/FLAG/GENERIC  SYS$PRINT
$ START/QUEUE  SYS$PRINT
$ INITIALIZE/QUEUE/FLAG  LPA0
$ START/QUEUE  LPA0
$ INITIALIZE/QUEUE/FLAG  LPB0
$ START/QUEUE  LPB0
$ INITIALIZE/QUEUE/FLAG/NOENABLE_GENERIC_PRINTING  LPC0
$ START/QUEUE  LPC0
$ INITIALIZE/QUEUE/FLAG/NOENABLE_GENERIC_PRINTING  HOLD
$ ASSIGN/QUEUE  LPB0  HOLD
$ START/QUEUE  HOLD
```

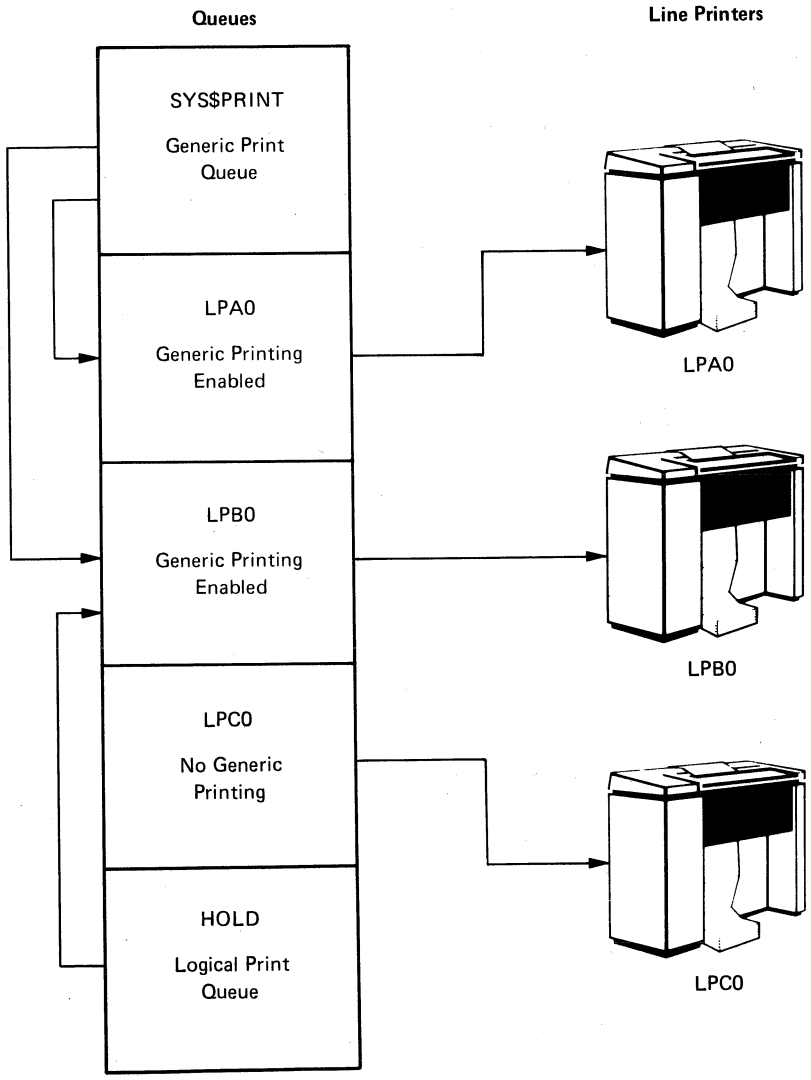


Figure 13-4 Setting Up Spooled Printers and Print Queues -- Adding a Logical Queue to the System with Three Line Printers

PART V

MONITORING THE ACTIVITY OF THE SYSTEM

This part of the VAX/VMS System Manager's Guide describes two tools used to monitor the activity of the system: the Display Utility Program (DISPLAY) and the operator's log file (OPERATOR.LOG).

Part V contains two chapters:

- Chapter 14: Using the Display Utility Program
- Chapter 15: The Operator's Log File

In addition, the commands SHOW SYSTEM and SHOW DEVICES are useful in monitoring the system. The VAX/VMS Command Language User's Guide fully describes these commands.

CHAPTER 14

USING THE DISPLAY UTILITY PROGRAM

The Display Utility Program (DISPLAY) can display system performance measurement statistics on a DECscope VT52 video display terminal or on a DECscope VT55 video display terminal. These displays can take the form of either graphs or tables.

The VAX/VMS system collects statistics on how the system is being used and on how the system responds to users' requests. These statistics are collected for two purposes: to aid system developers in understanding how the system operates and to aid system managers in improving system performance.

The types of information collected and displayed are as follows:

- File system statistics
- I/O system activity
- Use of processor modes (the computational workload)
- Page management statistics
- Nonpaged pool statistics
- Activity in scheduler state queues
- Principal users of CPU time
- System process activity

The VAX/VMS system starts accumulating a new set of performance measurement statistics each time the system is booted.

14.1 DCL COMMAND LINE USED IN RUNNING DISPLAY

The DISPLAY program is usually installed as a known image. Thus, ordinarily, its use is not limited to the system manager. Any user can run DISPLAY by entering the following VAX/VMS command line at the keyboard of a VT52 or VT55 video display terminal:

```
$ RUN SYS$SYSTEM:DISPLAY
```

DISPLAY returns the following prompt:

```
DISPLAY NAME?
```

To this prompt the user responds with one of the DISPLAY program commands. These commands are described in the following section.

14.2 COMMANDS OF THE DISPLAY PROGRAM

Table 14-1 summarizes the commands of the DISPLAY program. With the exception of CTRL/C , CYCLE, EXIT, and HELP, these commands are all names of displays that are updated periodically.

After the user types in the name of a display that is updated periodically, DISPLAY returns the following prompt:

```
UPDATE INTERVAL IN SECONDS?
```

The proper response to this question is either a decimal integer of the user's choice or a carriage return. By default, the carriage return gives an update interval of three seconds. After this response, the desired display appears.

After the user types in the command CYCLE, DISPLAY returns the following prompt:

```
INTERVAL BETWEEN DISPLAYS IN SECONDS?
```

The proper response to this question is either a decimal integer of the user's choice or a carriage return. By default, the carriage return gives an interval between different displays of 30 seconds. After this response, the cycling through the various displays begins; the update interval for each display is fixed.

Table 14-1
Commands of the DISPLAY Program

Command	Abbreviation	Display Produced
CTRL/C	None	The prompt DISPLAY NAME? This command is used to terminate any display and thus to return the user to the DISPLAY command level.
CYCLE	C	This command causes DISPLAY to continually cycle through the displays of performance measurement statistics.
EXIT	E	None. This command is used in terminating the DISPLAY program and returning the user to the DCL command level.
FCP	F	File Primitive Statistics. See Figure 14-1.
HELP	H	A summary of DISPLAY commands. See Figure 14-2.
IORATES	I	I/O System Rates. See Figure 14-3.
M2	M	Time in Processor Modes, VT52 and VT55. See Figure 14-4.

(continued on next page)

Table 14-1 (Cont.)
Commands of the DISPLAY Program

Command	Abbreviation	Display Produced
M5	None	Time in Processor Modes, VT55 only. See Figure 14-5.
PAGE	PA	Page Management Statistics. See Figure 14-6.
POOL	PO	Nonpaged Pool Statistics. See Figure 14-7.
S2	S	Number of Processes in Scheduler States, VT52 and VT55. See Figure 14-8.
S5	None	Number of Processes in Scheduler States, VT55 only. See Figure 14-9.
TOPUSERS	T	Top CPU Time Users. See Figure 14-10.
USERS	U	VAX/VMS Processes. See Figure 14-11.

14.3 DISPLAYS

DISPLAY produces displays of system performance measurement statistics, which are illustrated in Figures 14-1 through 14-11. Sections 14.3.1 through 14.3.9 describe these displays.

14.3.1 File Primitive Statistics

Figure 14-1 shows the display produced in response to the FCP (F) command. This display breaks down the types of file system activity taking place on the system. Three figures are given for each type of file system activity:

- Value -- the activity count during the last timed interval
- Rate/sec -- the activity rate per second over the last timed interval
- Avg rate -- the activity rate per second since the display was started

The types of file system activity monitored are:

- FCP calls -- number of QIO requests received by the file system
- Allocations -- number of calls that caused allocation of disk space
- Creates -- count of new files created
- Disk reads -- read requests performed on disks by file system

- Disk writes -- write requests performed on disks by file system
- Cache hits -- number of times disk block was found in file system's cache
- CPU tics -- count of CPU time used by the file system (in 10 millisecond tics)
- Window turns -- count of file mapping window misses
- File lookups -- count of the number of times file names were looked up in file directories
- File opens -- count of files that were opened

FILE PRIMITIVE STATISTICS
14:27:27

NAME	VALUE	RATE /SEC	AVG RATE	NAME	VALUE	RATE /SEC	AVG RATE
FCP CALLS	7	1.70	3.92	CACHE HITS	8	1.94	2.87
ALLOCATIONS	0	0.00	0.04	CPU TICS	10	2.43	2.85
CREATES	0	0.00	0.01	WINDOW TURNS	1	0.24	1.01
DISK READS	4	0.97	0.85	FILE LOOKUPS	7	1.70	2.64
DISK WRITES	0	0.00	0.15	FILE OPENS	1	0.24	0.97

Figure 14-1 Display of File Primitive Statistics Produced by the FCP Command

14.3.2 Display Produced by HELP

Figure 14-2 shows the display produced in response to the HELP (H) command. This display summarizes all DISPLAY commands. Note that the parts of commands enclosed in parentheses are optional.

```
                    DISPLAY


DISPLAY NAME? H

  COMMANDS ARE:

      CTRL-C - back to DISPLAY command level
      C(YCLE) - cycle between displays
      E(XIT) - exit display program
      F(CP) - file primitive statistics
      H(ELP) - type this message
      I(ORATES) - I/O system rates per second
      M(2) - time in processor modes (VT52)
      M5 - time in processor modes (VT55)
      PA(GE) - page fault statistics
      PO(OL) - nonpaged pool statistics
      S(2) - count of processes in scheduling states (VT52)
      S5 - count of processes in scheduling states (VT55)
      T(OPUSERS) - top CPU users by percent
      U(SERS) - list of system processes and status

DISPLAY NAME?
```

Figure 14-2 Display of Commands Produced by the HELP Command

14.3.3 I/O System Rates

Figure 14-3 shows the display produced in response to the IORATES (I) command. This display breaks down the types of I/O activity taking place on the system. Three figures are given for each type of I/O activity:

- Value -- the activity count during the last timed interval
- Rate/sec -- the activity rate per second over the last timed interval
- Avg rate -- the activity rate per second since the display was started

The types of I/O activity monitored are:

- Free list -- pages on the free page list
- Modify list -- pages on the modified page list
- Direct I/Os -- direct I/O operations performed
- Buffered I/Os -- buffered I/O operations performed
- Mailbox writes -- write requests performed on mailboxes

- Window turns -- count of file system mapping window misses
- Logname trans -- logical name translations performed
- File opens -- count of files that were opened
- Page faults -- page faults, system wide
- Pages read -- pages read from disk as a result of page faults
- Read I/Os -- number of I/O operations used to read the pages specified by pages read
- Pages written -- pages written to the paging file
- Write I/Os -- number of I/O operations used to write the pages specified by pages written
- Total inswaps -- swaps of processes and process headers to memory from disk

FREE LIST: 2016 — I/O SYSTEM RATES — 16:17:09 — MODIFY LIST: 35

NAME	VALUE	RATE /SEC	AVG RATE	NAME	VALUE	RATE /SEC	AVG RATE
DIRECT I/Os	32	7.30	1.50	PAGE FAULTS	65	14.84	1.83
BUFFERED I/Os	29	6.62	3.24	PAGES READ	4	0.91	0.11
MAILBOX WRITES	0	0.00	0.00	READ I/Os	2	0.45	0.07
WINDOW TURNS	3	0.68	0.14	PAGES WRITTEN	0	0.00	0.00
LOGNAME TRANS	39	8.90	0.98	WRITE I/Os	0	0.00	0.00
FILE OPENS	3	0.68	0.07	TOTAL INSWAPS	0	0.00	0.00

Figure 14-3 Display of I/O System Rates Produced by the IORATES Command

14.3.4 Time in Processor Modes

Figures 14-4 and 14-5 show the displays produced in response to the M2 (M) and M5 commands, respectively. These displays are records of the percent of total CPU time spent by the processor in each of seven processing modes. Data are displayed for the last timed interval. The result is a profile of the computational workload.

The processor modes monitored are:

- Time on interrupt stack (INTER) -- percent of time processor was executing on interrupt stack
- Time in kernel mode (KERNEL) -- percent of time processor was executing in kernel mode (does not include time on interrupt stack)
- Time in exec mode (EXEC) -- percent of time processor was executing in executive mode
- Time in super mode (SUPER) -- percent of time processor was executing in supervisor mode
- Time in user mode (USER) -- percent of time processor was executing in user mode (does not include time in compatibility mode)
- Time in compatibility mode (COMPAT) -- percent of time processor was executing compatibility mode user images
- Idle time (IDLE) -- percent of time processor was executing the null process

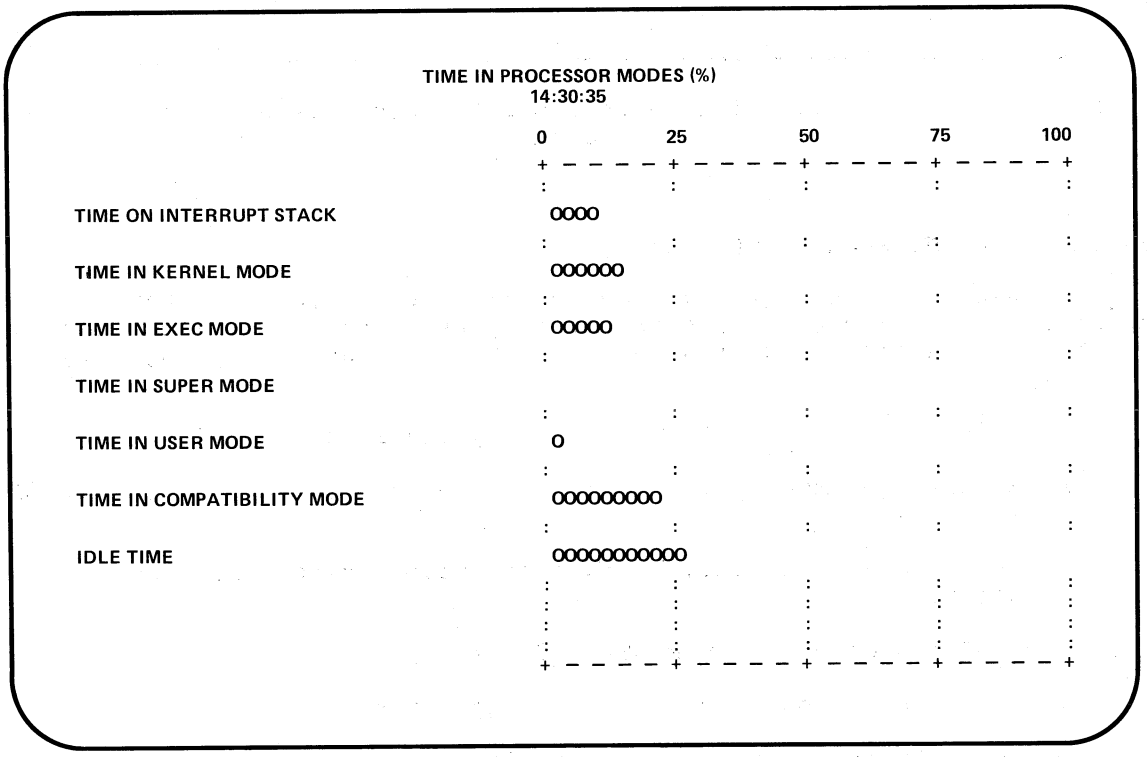


Figure 14-4 Display of Time in Processor Modes Produced by the M2 Command

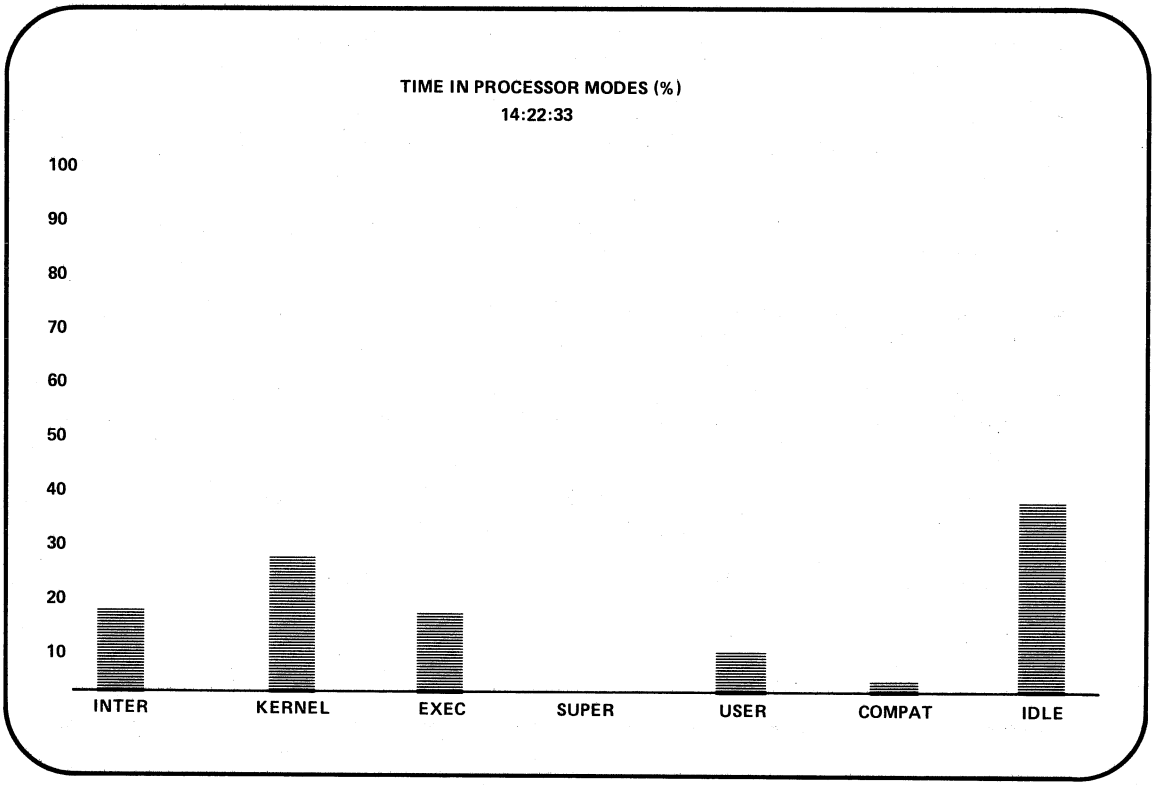


Figure 14-5 Display of Time in Processor Modes Produced by the M5 Command

14.3.5 Page Management Statistics

Figure 14-6 shows the display produced in response to the PAGE (PA) command. This display breaks down the types of paging activity taking place on the system. Two figures are given for each type of paging activity:

- Value -- the activity count during the last timed interval
- Rate/sec -- the activity rate per second over the last timed interval
- Avg rate -- the activity rate per second since the display was started

The types of paging activity monitored are:

- Free list -- pages on the free page list
- Modify list -- pages on the modified page list
- Page faults -- page faults, system wide
- Pages read -- pages read from disk as a result of page faults
- Read I/Os -- number of I/O operations used to read the pages specified by pages read

- Pages written -- pages written to the paging file
- Write I/Os -- number of I/O operations used to write the pages specified by pages written
- Free list -- faults for pages on the free page list
- Modified list -- faults for pages on the modified page list
- Demand zero -- faults for demand-allocated, zero-filled pages
- Write in prog -- faults for pages in the process of being written to backing store
- System faults -- faults in system space

FREE LIST: 181 PAGE MANAGEMENT STATISTICS MODIFY LIST: 19
14:25:19

NAME	VALUE	RATE /SEC	AVG RATE	NAME	VALUE	RATE /SEC	AVG RATE
PAGE FAULTS	50	11.96	8.04	FREE LIST	1	0.23	0.10
PAGES READ	9	2.15	2.08	MODIFIED LIST	3	0.71	1.55
READ I/Os	3	0.71	0.46	DEMAND ZERO	8	1.91	2.44
PAGES WRITTEN	0	0.00	0.42	WRITE IN PROG	0	0.00	0.00
WRITE I/Os	0	0.00	0.02	SYSTEM FAULTS	0	0.00	0.00

Figure 14-6 Display of Page Management Statistics Produced by the PAGE Command

14.3.6 Nonpaged Pool Statistics

Figure 14-7 shows the display produced in response to the POOL (PO) command. This display is an analysis of the condition of the nonpaged pool of memory. Two figures are given for each item displayed:

- Value -- a count over the last timed interval
- Avg -- the average value since the display was started

The items monitored are:

- I/O request packets left -- the number of entries in the special list of fixed-size I/O request packets
- Number of holes in pool -- the number of blocks of memory in the dynamically allocated pool

- Total space left -- total bytes in the remaining blocks of memory
- Largest block -- largest remaining block
- Smallest block -- smallest remaining block
- Number of blocks LEQ 32 bytes -- number of blocks less than or equal to 32 bytes

NONPAGED POOL STATISTICS
14:28:26

	VALUE	AVG
I/O REQUEST PACKETS LEFT	64	64.00
NUMBER OF HOLES IN POOL	27	27.00
TOTAL SPACE LEFT	12048	12048.00
LARGEST BLOCK	11136	11136.00
SMALLEST BLOCK	16	16.00
NUMBER OF BLOCKS LEQ 32 BYTES	18	18.00

Figure 14-7 Display of Nonpaged Pool Statistics Produced by the POOL Command

14.3.7 Number of Processes in Scheduler States

Figures 14-8 and 14-9 show the displays produced in response to the S2 (S) and S5 commands, respectively. These displays show the current number of processes in each of 14 process scheduling states.

These scheduling states are:

- Collided page wait (CPG) -- a wait state for resident and nonresident processes that have faulted a page currently in transition
- Mutex and misc. resource wait (MWT) -- a wait state for resident and nonresident processes awaiting the availability of a mutex semaphore or a dynamic resource
- Common event flag wait (CEF) -- a wait state for resident and nonresident processes waiting for some combination of event flags to be set in a common event block
- Page fault wait (PFW) -- a wait state for resident processes that have initiated the reading of a page as the result of a page fault

- Local event flag wait (LEF) -- a wait state for resident processes waiting for some combination of local event flags
- Local event flag wait, out of balance set (LFO) -- a wait state for nonresident processes waiting for some combination of local event flags
- Hibernate wait (HIB) -- a wait state for resident processes that have made a hibernate request
- Hibernate wait, out of balance set (HBO) -- a wait state for nonresident processes that have made a hibernate request
- Suspended wait (SSP) -- a wait state for suspended processes currently resident in the balance set
- Suspended wait, out of balance set (SPO) -- a wait state for suspended processes not currently in the balance set
- Free page wait (FPG) -- a wait state for resident and nonresident processes that need a free page of memory
- Compute (COM) -- a state for executable processes contained in the balance set
- Compute, out of balance set (CMO) -- a state for executable processes not currently in the balance set
- Current process (CUR) -- the state of a process actively being executed by the processor

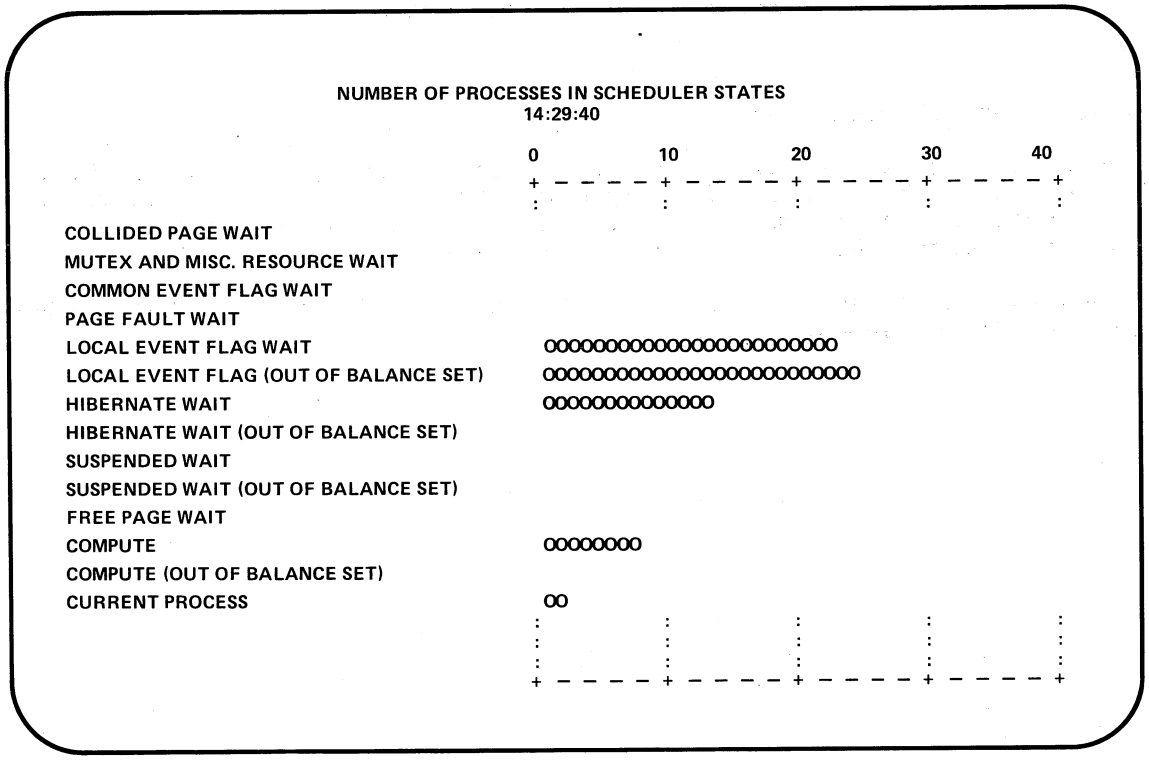


Figure 14-8 Display of Number of Processes in Scheduler States Produced by the S2 Command

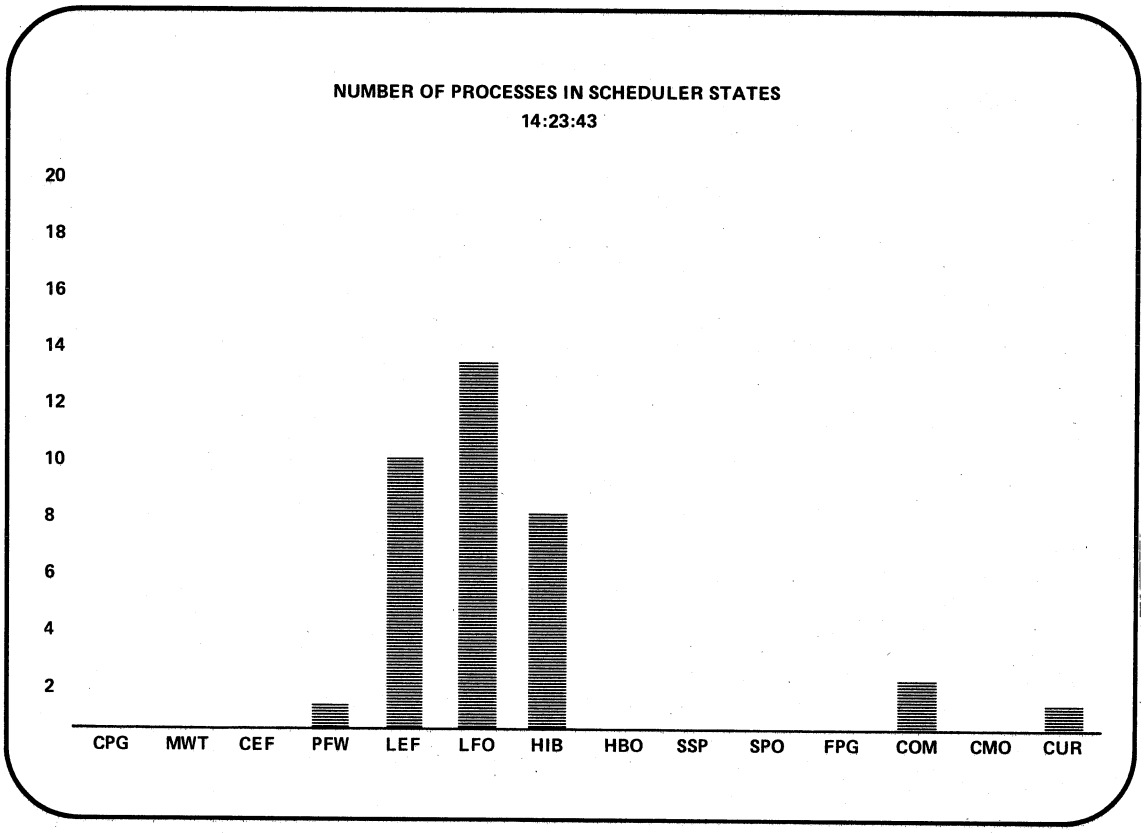


Figure 14-9 Display of Number of Processes in Scheduler States Produced by the S5 Command

14.3.8 Top CPU Time Users

Figure 14-10 shows the display produced in response to the TOPUSERS (T) command. This display is a record over the last timed interval of the percent of total CPU time taken by the principal users of the CPU. Users are identified by UIC and process name.

Processes not in memory at both the beginning and end of the timed interval do not appear in the display.

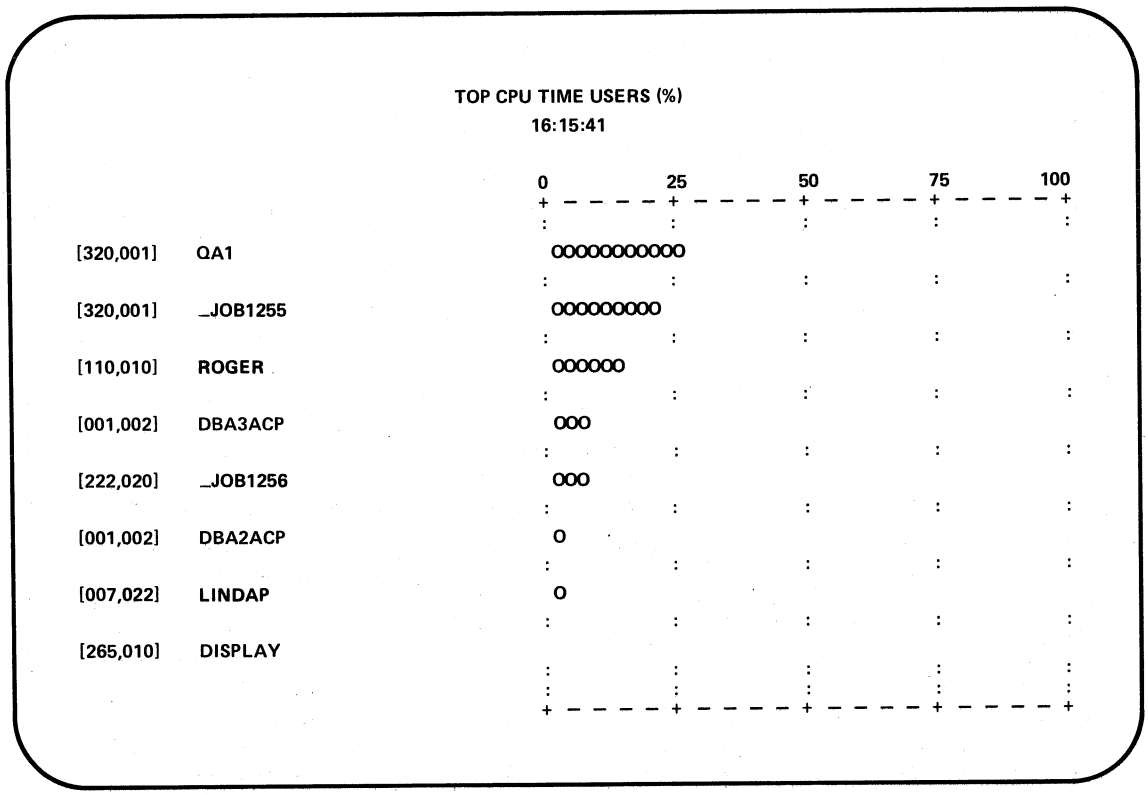


Figure 14-10 Display of Top CPU Time Users Produced by the TOPUSERS Command

14.3.9 VAX/VMS Processes

Figure 14-11 shows the display produced in response to the USERS (U) command. This display presents various facts about VAX/VMS processes, as follows:

- Processes -- the number of processes currently in the system
- Uptime -- the time (in days, hours, and minutes) since the system was last booted
- PID -- the unique 32-bit process identification assigned to a process
- UIC -- the user identification code assigned to a process
- State -- the current scheduling state of the process (see Section 14.3.7)
- PRI -- the current processing priority of a process
- Name -- process name

- Size -- the physical memory use (in pages) of a process (expressed in the form x/y, where x is the number of sharable pages and y is the total number of physical pages in use, the current working set)
- DIOCNT -- the cumulative count of direct I/O operations performed by a process since the process was created
- Faults -- the cumulative count of page faults that occurred for a process since the process was created
- CPU time -- the CPU time consumed by a process since the process was created

PROCESSES: 37 VAX/VMS PROCESSES 14:20:12 UPTIME: 3 00:48

PID	UIC	STATE	PRI	NAME	SIZE	DIOCNT	FAULTS	CPU TIME
001D002C	[101,010]	LEF	8	ANDREW	61/100	1624	3370	00:00:35.10
0017002D	[201,011]	LEFO	4	RMSTST	14/43	SWAPPED OUT		
0009002E	[105,020]	COM	5	RICHARDP	29/150	4496	3569	00:02:39.29
0031002F	[011,010]	LEF	5	LINDAP	14/100	6378	4778	00:01:57.99
00130030	[251,020]	LEF	9	JAMESP	18/128	12126	8254	00:03:57.25
00060031	[010,040]	LEFO	6	_TTC6:	0/32	SWAPPED OUT		
00170032	[007,007]	LEFO	4	HANKP	20/48	SWAPPED OUT		
00010033	[001,002]	HIB	8	DBA2ACP	31/71	99114	47	00:23:04.62
00350034	[360,015]	LEFO	4	JONES	15/44	SWAPPED OUT		
00220035	[221,020]	LEF	7	KATHYP	31/102	9044	9734	00:03:36.99
00010036	[007,377]	LEFO	4	SYSTEM	3/35	SWAPPED OUT		
00150037	[241,010]	LEF	7	WILLIAM_B	38/170	20771	8904	00:08:15.74
00010038	[001,002]	LEF	9	DBA1ACP	31/71	44381	47	00:05:24.83
00020039	[001,002]	HIB	8	PRTSYMB2	8/31	20470	21	00:11:40.62
000B003A	[010,010]	LEF	7	DAVIDP	37/114	452875	212260	14:51:45.25
0001003B	[001,006]	HIB	7	ERRFMT	0/26	900	15	00:00:08.25
0001003C	[002,001]	LEFO	8	OPERATOR	0/30	SWAPPED OUT		

Figure 14-11 Display of VAX/VMS Processes Produced by the USERS Command

CHAPTER 15

THE OPERATOR'S LOG FILE

The operator's log file (OPERATOR.LOG) is a system management tool that is useful in anticipating and preventing hardware and software failures. By regularly examining the operator's log file, the system manager can often detect tendencies, or trends, toward failures and can thereby see that corrective action is taken before these failures occur.

The system operator should, therefore, print out copies of the operator's log file regularly, and the system manager should retain these copies for reference. See the VAX/VMS Operator's Guide for descriptions of the messages that appear in this file and for instructions on printing copies of it. Figure 15-1 shows some typical messages of the operator's log file.

```
Opcom, 13-APR-1978 20:09:43.07, Logfile initialized, operator=_OPA0:
Opcom, 06:57:53.70, Device offline, LPA0:
Opcom, 06:58:25.70, Device offline, LPA0:
Opcom, 06:58:57.70, Device offline, LPA0:
Opcom, 06:59:29.70, Device offline, LPA0:
Opcom, 07:00:01.70, Device offline, LPA0:
Opcom, 07:00:33.70, Device offline, LPA0:
Opcom, 07:01:05.70, Device offline, LPA0:
```

```
Opcom, 11:30:47.70, Device offline, LPA0:
Opcom, 11:31:19.70, Device offline, LPA0:
Opcom, 11:31:51.70, Device offline, LPA0:
Opcom, 14-APR-1978 13:59:30.27, Terminal enabled, operator=_TTC3:
    Opcom, 13:59:41.88, ROGERP      Accnt=VMS
Opcom, TTC3:, 'TEST

Opcom, 15:26:42.73, Device offline, CRA0:
```

Figure 15-1 The Operator's Log File (OPERATOR.LOG)

PART VI

RECOGNIZING AND DEALING WITH ERRORS AND FAILURES

This part of the VAX/VMS System Manager's Guide deals with two topics related to reliable system performance:

- The error logging facilities, which can detect and log a variety of hardware and software errors
- The use of Software Performance Reports (SPRs) to report software problems

The following two chapters cover these topics:

- Chapter 16: Error Logging
- Chapter 17: Reporting Software Problems

CHAPTER 16

ERROR LOGGING

This chapter describes the VAX/VMS error logging facility. Section 16.1 explains the purpose of error logging and its role in system management. Section 16.2 explains how error logging works. Section 16.3 discusses the error log file.

16.1 THE PURPOSE OF ERROR LOGGING

The purpose of the error logging facility is to gather and maintain information on system errors and events as they occur; this information provides a detailed record of system activity. By running the report generator program SYE, the system manager or a DIGITAL field service representative can obtain a report of errors and events that have occurred within a specified period of time.

16.1.1 The Errors and Events Detected

The error logging facility detects a variety of hardware and software errors. When an error occurs, the facility gathers significant information about the current state of the system; the type of information gathered depends on the type of error detected. In addition to detecting actual errors, the facility monitors events that reflect other aspects of system performance. The recording of such events helps to define the system context in which actual errors occur.

The errors detected include:

- Device errors
- Interrupt timeouts
- Corrected read data
- Synchronous backplane interconnect (SBI) errors
- Fatal hardware errors such as parity errors in cache memory or in a translation buffer (machine checks)
- Fatal software errors (bugchecks)

The system events recorded include:

- Normal system start up (cold start)
- Start up after a power failure (warm start)

- Start up after a crash (crash restart)
- Volume mounts and dismounts
- Error log messages sent by an operator or by a process that issues a Send Message to Error Logger system service ($SNDERR)
- Time stamps that indicate that no errors or events have occurred within a given period of time

The error logging facility stores information on all these errors and events in a file on the system disk. This file becomes input to the report generator program SYE. Depending on how a user invokes it, SYE reports either on all the errors in the file or on a specific subset of errors. Parameters to SYE also determine the level of detail to be included in a report.

16.1.2 Using Error Reports

The error reports generated by SYE are useful tools in two basic ways:

- Reports aid preventive maintenance by identifying areas within the system that show potential for failure
- Reports speed the diagnosis of a failure by documenting the errors and events that led up to them

The detailed contents of the reports are most meaningful to DIGITAL field service personnel. However, the system manager should use the reports as an important indicator of the system's reliability. For example, when a report shows that a particular device is producing a relatively high number of errors, the system manager can consult DIGITAL field service. By running a diagnostic program to investigate the device, field service can attempt to isolate the source of the errors. Once identified, the source of the errors can possibly be eliminated and a failure averted.

In the event that a system component does fail, a field service representative can study error reports of system activity leading up to and including the failure. For example, if a device fails, the system manager can generate error reports immediately after the failure. One report might describe in detail all the errors associated with the failed device and occurring within the last 24 hours; another report might summarize all types of errors that occurred within the same time period. The summary report can put the device errors into a system-wide context. The field service representative can then run the appropriate diagnostic program for a thorough analysis of the failed device. Using the combined error logging and diagnostic information, the field service representative can proceed to correct the device.

The information made available by the error logging facility is essential to efficient maintenance of a VAX/VMS system. Error reports allow the system manager to track system performance and to anticipate failures. In turn, field service personnel rely on the reports as an aid to both preventive and corrective maintenance. Overall, effective use of the error logging facility, in conjunction with diagnostic programs, can significantly reduce the amount of time wasted by system downtime.

16.2 HOW ERROR LOGGING WORKS

The error logging facility consists of three working parts:

- A set of executive routines that detects errors and events and writes relevant information into error log buffers in memory
- A process called ERRFMT that periodically empties the error log buffers, transforms the descriptions of the errors into standard formats, and stores the formatted information in a file on the system disk.
- A process called SYE that generates readable reports from the information formatted by ERRFMT

The executive routines and the process ERRFMT operate continuously without user intervention. The routines fill up the error log buffers in memory with raw data on every detected error and event. When one of the available buffers becomes full, or when a time allotment expires, ERRFMT automatically empties the buffers. Sometimes a burst of errors can cause the buffers to fill up before ERRFMT can empty them. In this case, the system merely assigns a sequence number to the errors and events that occur, without preserving further information. As soon as ERRFMT frees the buffer space, the executive routines resume preserving error information in the buffers.

The ERRFMT process writes the error information into the error log file, ERRLOG.SYS, which is listed in the directory [SYSERR] on the system disk. The report generator SYE uses ERRLOG.SYS as its default input file and creates an error report as its output file. To control the number of versions of the error log file, it is best not to use ERRLOG.SYS as input to SYE. A user should rename the error log file and use the renamed file as input to SYE.

Unlike the executive routines and the ERRFMT process, a user must specifically invoke SYE and define several parameters. Mandatory parameters, which can be defaulted, include an input file specification, an output file specification, and the type of report to be generated. Optionally, a user can specify that SYE report on a particular device type or device unit and that SYE limit the report to a designated time period. The VAX/VMS Operator's Guide contains detailed instructions for running the SYE report generator.

Because the file ERRLOG.SYS can be renamed and the renamed error log file can then be copied to a removable volume, error reports can be generated either at the site where the errors occurred or at any other VAX/VMS installation. For example, a field service representative can rename and copy the error log file to take back to the field service office, where another VAX/VMS system can be used to generate error reports. Alternatively, a system manager can rename and copy to a disk file a version of the error log file that covers a crucial period of system activity. When a field service representative arrives on site, he or she can generate one or more reports from the copied file as well as from the current version of ERRLOG.SYS.

16.3 THE ERROR LOG FILE (ERRLOG.SYS)

The error logging process ERRFMT writes information on all detected errors into the file ERRLOG.SYS, which is listed in the directory [SYSERR] on the system disk. While SYE is accessing ERRLOG.SYS, ERRFMT cannot write any error information into it. Therefore, if SYE is accessing the highest version of ERRLOG.SYS when ERRFMT needs to

log an error, ERRFMT creates a new version of the file. The new version picks up logging errors where the previous version left off. All the versions of the ERRLOG.SYS file remain on the system disk until a user explicitly manipulates them in some way.

The fewer the log files, the simpler and more efficient it is to generate log reports. The system manager or operator can take steps to minimize or control the number of versions created.

For example, when generating several reports from the current error log file, a user should first rename the error log file and then use the renamed file as input to SYE. In this way, only one new error log file is created, and SYE does not prevent ERRFMT from accessing the new file. In addition, the user ensures that SYE is accessing the same error log file for each report. Separate SYE runs that use ERRLOG.SYS as the input file probably access different versions of that file.

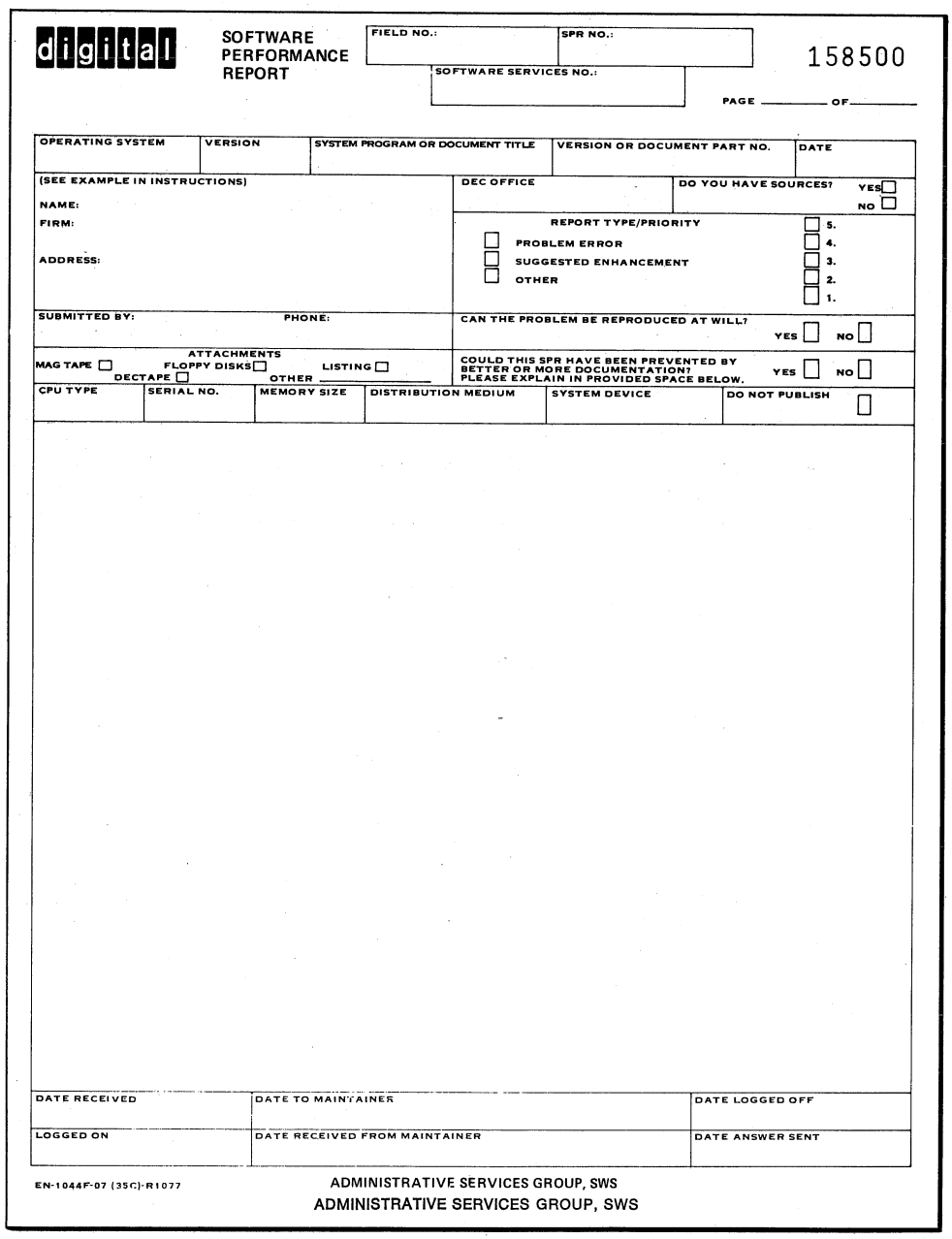
A responsibility of the system operator or system manager is to devise a plan for maintaining the versions of the error log file. One way to do this is to rename the highest version of ERRLOG.SYS on a daily basis. This causes a new error log file to be created and allows the old file (which was renamed) to be copied to a back-up volume where it can be kept as long as needed. For example, an operator could rename the current copy of ERRLOG.SYS every morning at 9:00 A.M. The file ERRLOG.SYS might be renamed ERRLOG.OLD. To free space on the system disk, the operator could then back up the renamed version of the error log file on a different volume and delete the renamed file from the system disk. Note that caution should be taken to ensure that error log files are not deleted inadvertently: the information contained in error log files is essential to efficient maintenance of a VAX/VMS system.

CHAPTER 17

REPORTING SOFTWARE PROBLEMS

To inform DIGITAL about problems with the VAX/VMS operating system or about errors in VAX/VMS software documents, the system manager should use the Software Performance Report (SPR), which is illustrated in Figure 17-1. Complete directions for completing the SPR form accompany the form itself.

A supply of SPR forms is included in the VAX/VMS software release distribution kit; more forms can be obtained from an SPR center. The addresses of these centers are listed on the backs of the forms.



digital

SOFTWARE PERFORMANCE REPORT

FIELD NO.: | SPR NO.: | SOFTWARE SERVICES NO.:

158500

PAGE _____ OF _____

OPERATING SYSTEM	VERSION	SYSTEM PROGRAM OR DOCUMENT TITLE	VERSION OR DOCUMENT PART NO.	DATE

(SEE EXAMPLE IN INSTRUCTIONS)
NAME:
FIRM:
ADDRESS:

DEC OFFICE

DO YOU HAVE SOURCES? YES ☐ NO ☐

REPORT TYPE/PRIORITY
☐ PROBLEM ERROR
☐ SUGGESTED ENHANCEMENT
☐ OTHER
☐ 5. ☐ 4. ☐ 3. ☐ 2. ☐ 1.

SUBMITTED BY: PHONE:

CAN THE PROBLEM BE REPRODUCED AT WILL? YES ☐ NO ☐

ATTACHMENTS
MAG TAPE ☐ FLOPPY DISKS ☐ LISTING ☐
DECTAPE ☐ OTHER _____

COULD THIS SPR HAVE BEEN PREVENTED BY BETTER OR MORE DOCUMENTATION? PLEASE EXPLAIN IN PROVIDED SPACE BELOW. YES ☐ NO ☐

CPU TYPE	SERIAL NO.	MEMORY SIZE	DISTRIBUTION MEDIUM	SYSTEM DEVICE	DO NOT PUBLISH
					☐

DATE RECEIVED	DATE TO MAINTAINER	DATE LOGGED OFF
LOGGED ON	DATE RECEIVED FROM MAINTAINER	DATE ANSWER SENT

EN-1044F-07 (35C)-R1077

ADMINISTRATIVE SERVICES GROUP, SWS

ADMINISTRATIVE SERVICES GROUP, SWS

Figure 17-1 Software Performance Report (SPR)

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

VAX/VMS
System Manager's Guide
AA-D027A-TE

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Did you find errors in this manual? If so, specify the error and the page number.

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Please indicate the type of reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify)_____

Name_____ Date_____

Organization_____

Street_____

City_____ State_____ Zip Code_____
or
Country

Please cut along this line.

Do Not Tear - Fold Here and Tape

digital

No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

RT/C SOFTWARE PUBLICATIONS TW/A14
DIGITAL EQUIPMENT CORPORATION
1925 ANDOVER STREET
TEWKSBURY, MASSACHUSETTS 01876

Do Not Tear - Fold Here

Cut Along Dotted Line

digital

PRINTED IN USA