# VAX/VMS
## System Management and Operations Guide

Order No. AA-M547A-TE

**May 1982**

This document describes the following tasks of system management with the aim of providing required information and guidelines for efficient operations: 1) setting up users' accounts, 2) managing public files and volumes, 3) controlling printing and batch operations, 4) monitoring and tuning system activity, 5) recognizing and responding properly to system errors and failures.

**REVISION/UPDATE INFORMATION:** This document supersedes the VAX/VMS System Manager's Guide (AA-D027B-TE) and the VAX/VMS Operator's Guide (AA-D025B-TE).

**SOFTWARE VERSION:** VAX/VMS Version 3.0

digital equipment corporation · maynard, massachusetts

First Printing, May 1982



Printed in U.S.A.

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

The following are trademarks of Digital Equipment Corporation:

| | | |
|---|---|---|
| DEC | DIBOL | RSX |
| DEC/CMS | EduSystem | UNIBUS |
| DECnet | IAS | VAX |
| DECsystem-10 | MASSBUS | VMS |
| DECSYSTEM-20 | PDP | VT |
| DECUS | PDT | digital |
| DECwriter | RSTS | |

ZK2156

**HOW TO ORDER ADDITIONAL DOCUMENTATION**

In Continental USA and Puerto Rico call 800-258-1710

In New Hampshire, Alaska, and Hawaii call 603-884-6660

In Canada call 613-234-7726 (Ottawa-Hull)
800-267-6146 (all other Canadian)

**DIRECT MAIL ORDERS (USA & PUERTO RICO)***

Digital Equipment Corporation
P.O. Box CS2008
Nashua, New Hampshire 03061

*Any prepaid order from Puerto Rico must be placed with the local Digital subsidiary (809-754-7575)

**DIRECT MAIL ORDERS (CANADA)**

Digital Equipment of Canada Ltd.
940 Belfast Road
Ottawa, Ontario K1G 4C2
Attn: A&SG Business Manager

**DIRECT MAIL ORDERS (INTERNATIONAL)**

Digital Equipment Corporation
A&SG Business Manager
c/o Digital's local subsidiary or approved distributor

Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Northboro, Massachusetts 01532

CONTENTS

CONTENTS

CONTENTS

Page

CONTENTS

Page

FIGURES



TABLES

# PREFACE

## MANUAL OBJECTIVES

The VAX/VMS System Management and Operations Guide is a reference book for all users responsible for system management and operations. This guide has the following objectives:

- To give the reader an understanding of the reasons for performing VAX/VMS system management tasks and to show how to perform these tasks.
- To provide general information about day-to-day operations of the VAX/VMS operating system.
- To serve as a one-volume reference source of information, procedures, and examples that pertain to the operation of the VAX-11 single-processor system.
- To introduce utilities and commands used in VAX/VMS system management and operation, and to provide cross-references to other books in the document set that contain reference information on the subject.

## INTENDED AUDIENCE

This guide is intended for users of the VAX/VMS system who must perform the functions of a system manager or operator on one or more VAX/VMS single-processor systems. Section 1.1 outlines and briefly describe these duties. The reader of this guide is most likely a data processing generalist, not necessarily a programmer, and probably not a systems programmer.

Note that this manual does not address any of the issues of managing a VAX-11/782 attached-processor system. The VAX-11/782 User's Guide presents this information.

## STRUCTURE OF THIS DOCUMENT

This document covers all aspects of controlling the operations of a VAX/VMS installation, in twelve chapters and one appendix, as follows:

- Chapter 1 briefly describes the functions of system management and operations and the duties entailed
- Chapter 2 explains how to add and remove users from the system, using the Authorize Utility (AUTHORIZE) to maintain the user authorization file
- Chapter 3 describes how various system resources are protected through the mechanism of the user identification code (the UIC)

- Chapter 4 describes the use of limits, quotas, and privileges in controlling the system resources
- Chapter 5 describes how to handle files and volumes
- Chapter 6 describes how to install images as known images
- Chapter 7 describes how to shut down and restart the system
- Chapter 8 describes how to control print and batch queues
- Chapter 9 describes the tools available to detect and correct errors
- Chapter 10 describes the system parameters, their default values, and information needed to adjust them
- Chapter 11 describes the System Generation Utility (SYSGEN) and its role in generating and maintaining a well-running system
- Chapter 12 describes how to monitor the system and tune it for improved performance

## ASSOCIATED DOCUMENTS

The VAX-11 Information Directory and Index lists and describes all the documents that you may need to refer to in the course of performing system management and operations, and contains a master index of all topics discussed in the VAX/VMS document set.

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

The following documents may also be useful:

- VAX/VMS Command Language User's Guide
- VAX/VMS Release Notes
- VAX-11 Utilities Reference Manual
- VAX/VMS System Messages and Recovery Procedures Manual
- VAX-11/RSX-11M User's Guide
- RMS-11 User's Guide (For those users who continue to use the RMSBCK and RMSRST utilities to back up and restore files. This book is no longer part of the VAX/VMS document set since new VAX/VMS utilities offer improved functionality)
- VAX/VMS Magnetic Tape User's Guide

You should also consult the software installation guide for your VAX-11 processor.

For hardware operating instructions, refer to the appropriate hardware manual for VAX-11 users.

For managing network operations, refer to the DECnet-VAX System Manager's Guide.

For managing VAX-11/782 attached-processor systems, refer to the VAX-11/782 User's Guide.

## CONVENTIONS USED IN THIS DOCUMENT

| Convention | Meaning |
|---|---|
| `$ RUN SYS$SYSTEM:SYSGEN`<br>`SYSGEN>` | prompt characters that the system prints or displays in black letters. All user-entered commands are shown in red letters. |
| Uppercase words and letters | Uppercase words and letters, used in examples, indicate that you should type the word or letter exactly as shown. |
| Lowercase words and letters | Lowercase words and letters, used in format examples, indicate that you are to substitute a word or value of your choice. |
| Quotation marks<br>Apostrophes | The term "quotation marks" refers to double quotation marks ("). The term "apostrophe" refers to a single quotation mark ('). |
| [ ] | Square brackets indicate that the enclosed item is optional. However, square brackets are not optional in:<br>- The syntax of a directory name<br>- A file specification<br>- The syntax of a substring specification<br>- The SET UIC command |
| ... | A horizontal ellipsis indicates that the preceding item(s) can be repeated one or more times. For example:<br>`file-spec[,...]` |
| .<br>.<br>. | A vertical ellipsis indicates that not all the statements in an example or figure are shown. |
| `$ RUN`<br>`$_File:` | In examples of commands you enter and system responses, all output lines and prompting characters that the system prints or displays are shown in black letters. All the lines you type are shown in red letters. |
| RET <RET> | A symbol with a 1- to 3-character abbreviation indicates that you press a key on the terminal. |
| CTRL/Y or CTRL/x | The symbol CTRL/x indicates that you must press the key labeled CTRL while you simultaneously press another key, for example, CTRL/C, CTRL/Y, CTRL/O. |

Unless otherwise noted, all numeric values are represented in decimal notation.

Unless otherwise specified, you terminate commands by pressing the RETURN key.

# SUMMARY OF CHANGES

While this is a new manual in title and format, it contains much material from the earlier manuals it replaces: the VAX/VMS System Manager's Guide and the VAX/VMS Operator's Guide. Thus, it is possible to identify technical areas in this new manual where major changes have occurred since the Version 2.0 publication of the corresponding information.

## 1.0 INFORMATION RELOCATED WHEN THE MANUALS WERE COMBINED

You will find that reference material for the following utilities formerly found in the VAX/VMS System Manager's Guide has been moved to the VAX-11 Utilities Reference Manual:

- Authorize Utility (AUTHORIZE)
- Disk Quota Utility (DISKQUOTA)
- Display Utility (DISPLAY) -- replaced by the Monitor Utility (MONITOR), which is also described in the VAX-11 Utilities Reference Manual
- Install Utility (INSTALL)
- RMS Share Utility (RMSSHARE)
- SYE Utility
- System Generation Utility (SYSGEN)

Note also that the error messages for each of these utilities have been relocated to the VAX/VMS System Messages and Recovery Procedures Manual.

The descriptions of all commands requiring the operator privilege (OPER) that were formerly found in the VAX/VMS Operator's Guide have been moved to the VAX/VMS Command Language User's Guide, where they appear in alphabetical order with the other DCL commands.

## 2.0 TECHNICALLY CHANGED INFORMATION

Technical changes in the following major areas reflect changes in the Version 3.0 software:

- MONITOR -- All references to the Display Utility have been updated to refer to the new Monitor Utility.
- BACKUP -- All references to the Disk Save and Compress Utilities have been updated to show the new Backup Utility as the preferred utility. Also, new descriptions featuring BACKUP facilities have been added.

- Quotas -- Two new quotas, the enqueue quota (ENQLM) and the working set extent quota (WSEXTENT), have been added.

- Privileges -- The new privilege SYSLCK has been added.

- Logical names for the system device, system disk, and system directories -- The new logical names are reflected throughout the manual.

- Mounting disks -- The new features of mount verification and operator-assisted mounts have been described.

- Operator messages -- The messages have changed significantly. All OPCOM messages now include the full date. Also, the technique for numbering requests has changed. As a result, request id numbers are not immediately reused after a request completes.

- System start-up and shutdown -- The procedures STARTUP.COM and SHUTDOWN.COM have been modified. See Chapter 7.

- Restarting the job controller, error logger, and OPCOM -- The method for restarting these processes has changed.

- Emptying the queue file -- See Chapter 8 for a description of how to use the new system parameter REINITQUE to clear a corrupted queue file.

- Login procedures -- Login sequence changes have been made. See Chapter 2.

- Concealed device names -- Examples generally conceal physical device names with logical names, in keeping with the new VAX/VMS conventions.

- Automatic generation of parameter files -- The default parameter file is now automatically generated using a procedure named AUTOGEN.COM, which means that DIGITAL no longer ships the nUSER.PAR files such as 32USER.PAR. You can still create you own parameter files, if necessary. However, the file AUTOGEN creates should be a more accurate reflection of your configuration than the DIGITAL-supplied files of previous versions were.

- Tuning -- The system permits growth in a number of key areas where the system parameters formerly provided fixed limits. As a result, the system can do more to tune itself, so you should have even less need now to perform tuning. The chapter on tuning the system has been revised to reflect this philosophy and explain the changes.

- Nonpaged dynamic pool size -- The rules for configuring the size of nonpaged pool have been greatly simplified. In addition, each of the nonpaged pool areas can grow beyond its current size. There is little penalty for underconfiguring one of the size parameters.

## 3.0 NEW INFORMATION

This manual contains information on the following new subjects:

- Accounting Utility -- This new utility is introduced here as are the older system management utilities; however, reference material for all the utilities now resides in the VAX-11 Utilities Reference Manual.

- Mounting disks -- The new features of mount verification and operator-assisted mounts are described in Chapter 5.

- New System Parameters -- Chapter 10 includes descriptions of the new system parameters:

| | | |
|---|---|---|
| BORROWLIM | MPW_WAITLIMIT | SCSBUFFCNT |
| DEADLOCK_WAIT | MVTIMEOUT | SCSCONNCNT |
| DISMOUMSG | NPAGEVIR | SCSFLOWCUSH |
| FREEGOAL | PAGFILCNT | SCSMAXDG |
| FREELIM | PAPOLLINTERVAL | SCSRESPCNT |
| GBLPAGFIL | PAPOOLINTERVAL | SCSSYSTEMID |
| GROWLIM | PASTDGBUF | SRPCOUNT |
| IRPCOUNTV | PASTIMOUT | SRPCOUNTV |
| JOBQUEUES | PASTRETRY | SRPSIZE |
| LOCKIDTBL | PQL_DENQLM | SWPFILCNT |
| LOGGHASHTBL | PQL_DWSEXTENT | TIMEPROMPTWAIT |
| LOGPHASHTBL | PQL_MENQLM | TTY_ALTALARM |
| LOGSHASHTBL | PQL_MWSEXTENT | TTY_ALTYPAHD |
| LONGWAIT | REINITQUE | TTY_CLASSNAME |
| LRPCOUNT | RESHASHTBL | TTY_DEFCHAR2 |
| LRPCOUNTV | RJOBLIM | TTY_DIALTYPE |
| LRPSIZE | RMS_EXTEND_SIZE | TTY_DMASIZE |
| MOUNTMSG | RMS_FILEPROT | TTY_SILOTIME |
| MPW_THRESH | RMS_PROLOGUE | UDABURSTRATE |

- New User-Defined System Parameters -- You can define any of the following reserved system parameters for site-specific purposes: USERD1, USERD2, USER3, and USER4.

- Announcements -- You can define the logical names SYS$ANNOUNCE and SYS$WELCOME to provide site specific announcements and welcome messages.

- Primary and secondary days -- You can define days of the week and periods within those days as enabled or disabled for user logins, on a user-by-user basis. See Chapter 2.

- Uses of the new DCL commands are suggested wherever appropriate, and behavior changes in the existing commands are shown in the descriptions.

## 4.0 MISCELLANEOUS IMPROVEMENTS

Chapter 2, User Accounts, has been expanded to describe more procedures for establishing and deleting user accounts and for writing login command procedures.

The chapter entitled Protection and Control has been rewritten to be more comprehensive. (See Chapter 3, UIC-Based Protection.)

Likewise, the description of Software Performance Reports (SPRs) has been expanded to provide guidelines for supplying the information required by DIGITAL with your report.

The chapter entitled Print and Batch Queues has been reorganized. (See Chapter 8.)

# CHAPTER 1

# INTRODUCTION

System management of a VAX/VMS installation entails two main responsibilities: system performance and system operation. These responsibilities require that you:

- Make decisions that relate to optimizing the overall performance and efficiency of the system
- Perform procedures that relate to the overall management and control of the system

To make good decisions, you must understand both the needs of users and the capabilities of the VAX/VMS operating system. To perform system management procedures well, you must 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 you 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 a VAX/VMS system. System management cannot be done by rote or from a cookbook. Rather, you must -- by combining an understanding of users' needs and system capabilities with a working knowledge of the functions of VAX/VMS -- work out your own strategy for effective system management.

## 1.1 OVERVIEW OF 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 can be performed by one individual, or by a single system manager who is assisted by one or more system operators, or it can be shared by several persons, some or all of whom may also serve as system operators. Since the designation of these roles varies from site to site, this guide does not make job distinctions between system managers and operators. For the most part, as you read this document it will appear that one person is performing all the functions. Bear in mind that this is not always the case.

A computer installation exists 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.

### 1.1.1 System Management Tasks

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

- Bringing the system up
- Setting up users' accounts
- Controlling the operation of the system (see Section 1.2)
- Configuring the system for good performance
- Planning to meet future requirements

The first topic (bringing the system up) is the subject of the software installation guide for your VAX-11 processor. The next three topics are the subjects of this system management and operations guide. The final topic is beyond the scope of this manual.

### 1.1.2 Operational Tasks

The VAX/VMS system runs, to a great extent, without operator intervention. However, in many installations, one or more operators keep the system running smoothly by performing some of the following tasks:

- Physically mounting magnetic tapes and disks at the request of the users who own them
- Initializing and mounting system volumes
- Backing up public files and volumes
- Carrying out user requests
- Sending messages to specific users
- Broadcasting messages to all users
- Controlling print and batch queues
- Tending line printers
- Tending card readers
- Monitoring the system
- Observing and responding to emergencies
- Printing copies of the operator's log file and the error log file
- Shutting down and restarting the system
- Bringing up and shutting down network components

To carry out these tasks, you will probably have to interact with:

- Other users of the system

- The VAX/VMS operating system
- The VAX-11 processor on which the operating system runs

As a VAX/VMS system operator, you can perform many of your duties from user terminals. However, one of your chief functions, communicating with other users, must be performed at a terminal that has been defined as an operator's terminal. You can define a terminal to be an operator's terminal by using the privileged command REPLY/ENABLE (described in Section 9.2.4.3). Your relationship with the users and their programs is based on messages that pass between you and other users. A record of these messages is displayed on the operator's terminal; the messages are also entered in the operator's log file for later reference.

Other operator functions can be performed only from the system console terminal. These functions include bootstrapping and communicating with the VAX-11 processor's console subsystem.

The interactions between you and the VAX/VMS operating system are based on your ability to use the entire set of VAX/VMS commands and on your understanding of the system messages displayed or printed on the operator's terminal. (System messages are also entered in the operator's log file.)

Interactions between you and the VAX-11 processor require you to operate and maintain the peripheral devices supported by the system. For detailed instructions on operating and maintaining these peripheral devices, see the appropriate hardware manual for VAX-11 users.

## 1.2 MANAGING THE VAX/VMS ENVIRONMENT

In managing the environment of a VAX/VMS system, you need to understand the functions to be performed and the tools available. You have at your command powerful utilities and commands to monitor and control the system resources.

### 1.2.1 VAX/VMS Components

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

You will also have to customize some of the operations of the operating system and incorporate optional software into the system. Optional software, which runs under control of the VAX/VMS operating system, includes languages such as VAX-11 FORTRAN and data management systems such as VAX-11 DBMS.

For an explanation of the steps that you may have to take to install a VAX/VMS system, see the software installation guide for your VAX-11 processor.

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

- SYS$LIBRARY or SYS$SHARE
  This directory contains various macro and object libraries and shareable images.

- SYS$MESSAGE
  This directory contains system message files.

- SYS$MANAGER
  This directory contains files used in managing the operating system.

- SYS$HELP
  This directory contains text files and help libraries for the Help Utility.

- SYS$ERRORLOG
  This is the directory for the error log file (ERRLOG.SYS).

- SYS$TEST
  This directory contains files used in testing the functions of the operating system.

- SYS$MAINTENANCE
  This directory contains system diagnostic programs.

- SYS$UPDATE
  This directory contains files used in applying system updates.

- SYS$EXAMPLES
  This directory contains sample driver programs, user-written system services, and other source programs.

- SYS$SYSTEM
  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 software installation guide for your VAX-11 processor.

### 1.2.2 Privileges

System management functions require privileges that are denied to most users. Table 1-1 summarizes the privileges you need to use certain procedures and commands documented in this manual. Chapter 2 describes how to set up authorization records that grant these privileges. Chapter 4 provides more detailed information about the privileges and who should receive them.

### 1.2.3 DIGITAL Command Language Commands

This manual contains numerous references to the DIGITAL Command Language (DCL) commands used most often to keep a VAX/VMS system running smoothly. Most of these commands require the OPER user privilege. For detailed descriptions of these commands, see the VAX/VMS Command Language User's Guide.

**1.2.3.1 Command Line Format** - The general format of a DCL command is:

```
command-name[/qualifiers...] parameter[/qualifiers...][...]
```

Because a command can be continued on more than one line, the term "command string" is used to define the entire command that is passed

to the system. A command string is the complete specification of a command, which includes the command name, command qualifiers, parameters, and parameter qualifiers. See the VAX/VMS Command Language User's Guide for a detailed description of command syntax.

**Table 1-1: System Management Privileges**

| Privilege | Function |
|---|---|
| ACNT | Create a process for which no accounting records are made |
| ALLSPOOL | Allocate spooled devices |
| ALTPRI | Increase the base execution priority for any process |
| BYPASS | Bypass user identification code (UIC) protection in accessing files |
| CMKRNL | Change execution mode to kernel |
| GROUP | Affect processes within the same group |
| GRPNAM | Insert logical names into the group logical name table |
| LOG_IO | Issue logical I/O requests |
| NETMBX | Create network devices |
| OPER | Execute operator functions |
| PHY_IO | Issue physical I/O requests |
| PRMCEB | Create or delete permanent common event flag clusters |
| PRMMBX | Create permanent mailboxes |
| PSWAPM | Change process swap mode |
| SHMEM | Create or delete data structures in shared memory |
| SYSGBL | Create system global sections |
| SYSNAM | Insert logical names into the system logical name table |
| SYSPRV | Take system UIC protection when accessing files |
| TMPMBX | Create temporary mailboxes |
| VOLPRO | Override volume protection |
| WORLD | Control the execution of any process in the system |

**1.2.3.2 Summary of DCL Commands -** Table 1-2 briefly describes the DCL commands you will use most frequently.

Table 1-2: DCL Commands Commonly Used

| Command | Function |
|---|---|
| ACCOUNTING | Provides reports on system resource utilization based on data recorded in the accounting log file |
| ALLOCATE | Reserves a device for use by a single user and, optionally, assigns a logical name to the device |
| ANALYZE/DISK_STRUCTURE | Checks the readability and validity of Files-11 Structure Level 1 and Files-11 Structure Level 2 disk volumes |
| ASSIGN/MERGE | Removes all jobs from one queue and places them in another queue |
| ASSIGN/QUEUE | Assigns a logical queue to a specific device |
| BACKUP | Saves, copies, restores, and compares files; lists file information in a save set |
| COPY | Copies one or more files into one or more additional files |
| DEALLOCATE | Relinquishes use of a previously allocated device, thus making the device available to other users |
| DEASSIGN/QUEUE | Deassigns a queue from a specific device |
| DELETE/ENTRY[1] | Deletes an entry from a print or batch queue or stops processing of the current job |
| DELETE/QUEUE | Deletes batch and print queues |
| DIRECTORY | Displays information about a file or group of files |
| DISMOUNT | Releases the connection between a user and a disk or tape volume that is currently mounted on a device |
| INITIALIZE | Readies a mass storage volume by deleting any existing data and writing a label on the volume |

1. Allows a user with either operator (OPER) or world (WORLD) privilege to affect any job in the system.

(continued on next page)

Table 1-2 (Cont.): DCL Commands Commonly Used

| Command | Function |
|---|---|
| INITIALIZE/QUEUE | Creates batch and print queues |
| MONITOR | Monitors system-wide performance data |
| MOUNT | Makes a disk or tape volume available for the reading or writing of files and assigns a logical name to the device on which the volume is mounted |
| PRINT | Queues a file for printing on a specific device |
| REPLY | Allows the operator to communicate with system users, selectively enable and disable operator status, and examine the operator's log file |
| SET ACCOUNTING | Selectively enables and disables the recording of particular kinds of accounting information |
| SET DAY | Overrides the default day type associated with the user authorization file |
| SET DEVICE | Establishes the spooling and error-logging status of a specific device |
| SET DIRECTORY | Modifies the characteristics of a directory |
| SET FILE | Modifies the characteristics of a file |
| SET LOGINS | Establishes the maximum number of users able to log in to the system |
| SET PRINTER | Establishes the characteristics of a specific line printer |
| SET PROCESS | Changes the execution characteristics of the curent process |
| SET PROTECTION/DEVICE | Establishes the protection for a non-file-structured device |

(continued on next page)

Table 1-2 (Cont.): DCL Commands Commonly Used

| Command | Function |
|---|---|
| SET QUEUE/ENTRY[1] | Changes the status or attributes of jobs in print or batch queues that have not yet been processed by the system |
| SET TIME | Resets the system time |
| SET UIC | Establishes a new user identification code (UIC) as the process UIC |
| SET VOLUME | Modifies the characteristics of one or more mounted Files-11 volumes |
| SHOW DEFAULT | Displays the current default directory and disk device |
| SHOW DEVICES | Displays the status of devices in the system |
| SHOW ERROR | Displays the error count for the CPU, memory, and all physical devices with error counts greater than 0 |
| SHOW MEMORY | Displays the availability and utilization of memory resources |
| SHOW QUEUE | Displays the names, job identification numbers, and status of current and pending jobs in print and batch job queues |
| SHOW TIME | Displays the current date and time on the terminal |
| SHOW USERS | Displays the current users of the system |
| START/QUEUE | Starts batch and print queues |
| STOP[1] | Halts execution of a command procedure, program, subprocess, or detached process |
| STOP/QUEUE | Suspends or controls batch and print queues |
| SUBMIT | Queues one or more command procedure(s) to a batch job queue |
| TYPE | Displays the contents of a file or files at the current output device |

1. Allows a user with either operator (OPER) or world (WORLD) privilege to affect any job in the system.

## 1.3 UTILITIES

Table 1-3 lists those utilities frequently employed primarily as system management tools and gives a brief description of the purpose of each. Further details on each of these utilities, as well as other utilities you will want to become familiar with, can be found in the VAX-11 Utilities Reference Manual.

Table 1-3 System Management Utilities

| Utility Name | Function |
|---|---|
| AUTHORIZE | Modifies the existing user authorization file or creates a new one |
| DISKQUOTA | Controls the usage of disk volumes |
| INSTALL | Installs and maintains known images |
| RMSSHARE | Enables the VAX-11 Record Management Services (RMS) file sharing capability and displays figures on allowable and actual usage |
| SYE | Reports the contents of the system error log file |
| SYSGEN | Performs tasks associated with system generation such as loading and connecting drivers, creating or extending swapping and paging files, displaying or modifying the values of the system parameters, and enabling multiport memory units |

# CHAPTER 2

# AUTHORIZING SYSTEM USERS

You identify users who may log in to the system and place controls on their activities by maintaining a record for each user in the user authorization file (UAF). The file specification of the UAF is SYS$SYSTEM:SYSUAF.DAT. You maintain the UAF with the Authorize Utility (AUTHORIZE). AUTHORIZE is further described in the VAX-11 Utilities Reference Manual.

## 2.1 THE USER AUTHORIZATION FILE

Each record in the UAF includes the following information:

- Name and password -- Identifies a user to the system at login time
- User identification code (UIC) -- Identifies a user by a group number and a member number (see Chapter 3 for detailed information)
- Default file specification -- Provides default device and directory names for file access
- Default command language -- Names the default command interpreter as DCL or MCR
- Login command file -- Names a command procedure to be executed automatically at login time
- Login flags -- Allow you to inhibit the use of the CTRL/Y function, restrict users to their default command interpreters, control the time of day and days of the week when logins are permitted, and/or lock user passwords
- Priority -- Specifies the base priority of the process created for the user at login time (see Chapter 4 for more detailed information)
- Privileges -- Limits activities the user may perform (see Chapter 4 for more detailed information)

The system uses the UAF to validate each login attempt. The fields in each UAF record control this process in a number of ways, as described in this chapter and in the description of AUTHORIZE in the VAX-11 Utilities Reference Manual. Section 2.2 describes the sequence of events that occurs during a user login and how the UAF is used during login.

The software distribution kit provided with a new VAX/VMS system contains a UAF of four records:

- DEFAULT -- Serves as a template in creating user records in the UAF. A new user record is assigned the values of the DEFAULT record except where you explicitly override those values. The DEFAULT record can be modified but cannot be renamed or deleted from the UAF.

- SYSTEM -- Provides a means for you to log in with full privilege. The SYSTEM record can be modified but cannot be renamed or deleted from the UAF. Note that if you change the SYSTEM record, particularly the default device and directory and the privileges, you could prevent successful installation of optional software products or future VAX/VMS maintenance releases.

- FIELD -- Permits DIGITAL field service personnel to check out a new system. The FIELD record can be deleted once the system is installed.

- SYSTEST -- Provides an appropriate environment for running the User Environment Test Package (UETP). The SYSTEST record can be deleted once the system is installed.

AUTHORIZE can run concurrently with user login operations as long as VAX-11 RMS file sharing is in effect (that is, the RMS Share Utility (RMSSHARE) has been run). A slight chance exists that a user login operation might fail on a record lock, but the user need only try again to remedy the situation.

## 2.2 LOGIN SEQUENCE

When a terminal is activated (by turning it on and pressing RETURN if directly connected, or by dialing in to a system and observing the remote connect protocol), and that terminal is not allocated by a user process, the system prompts for a name and password. The person using the terminal must type a name and password that exist in a UAF record or further access to the system is denied.

If the password is accepted, then the login flags are examined. The DISUSER flag is the first to be checked. If DISUSER is set, the login attempt fails. Note that setting this flag for powerful, infrequently used accounts (such as SYSTEM, SYSTEST, and FIELD) virtually eliminates the risk of guessed passwords for those accounts.

If the DISUSER flag is not set, the next check is for primary or secondary day restrictions. You can define certain days of the week as primary days and the remaining days as secondary days (with the AUTHORIZE qualifier /PRIMEDAYS). The current day of the week is checked for which type of day it is. Then, checks are made for which hours during this type of day logins are permitted (as defined by the /P_RESTRICT and /S_RESTRICT qualifiers). If the current hour has no restriction against it, the login is one step closer to success. Otherwise, it fails immediately.

Finally, the DISNETWORK and DISDIALUP flags are examined. If DISNETWORK has been set for the appropriate type of day (primary or secondary), the login (if attempted through the DCL command SET HOST) is not allowed. Similarly, the login fails if the DISDIALUP flag is set and the user has attempted to dial in to the system. (To implement the feature to disable dialups, you must also issue the DCL command SET TERMINAL/PERMANENT/MODEM, for all dial-in terminal lines,

but only for the dial-in lines. A user whose DISDIALUP flag is set will be unable to log in on any terminal with the modem characteristic.)

If the login is successful, control passes to the command interpreter (for example, DCL) named in the user's record of the UAF. The system checks whether or not SYS$SYLOGIN has been defined. If it has, the logical name is translated (in most cases to SYS$MANAGER:SYLOGIN.COM) and that procedure is executed. When the procedure completes, another check is made for the name of a login command procedure in that user's record of the UAF. If a command procedure is specified in the LGICMD field and that procedure exists, it is executed. Otherwise, if the LGICMD field is blank, then the user's command file named LOGIN is executed automatically (if it exists). The command interpreter prompts for user input (DCL displays a dollar sign) and the user responds with commands acceptable to the command interpreter. (DCL accepts those commands documented in the VAX/VMS Command Language User's Guide.) However, the system prohibits activities that violate the user's privilege allowance or exceed resource quotas, and accords the user processor time as regulated through the base priority.

## 2.3 GENERAL MAINTENANCE OF THE UAF

Typically, you use the UAF supplied with the distribution kit. (You can, however, rename the UAF with the DCL command RENAME, then create a new UAF with AUTHORIZE.) You should limit any kind of access to this file to just the system account (see Chapter 3 for guidelines on protecting system files). Furthermore, each time you modify the file, you should create a back-up copy (see Chapter 5 for guidelines on backing up files).

The UAF is accessed with file sharing enabled, and updates to the UAF are made on a per-record basis, which eliminates the need for both a temporary UAF and a new version of the UAF after each AUTHORIZE run. Updates become effective as soon as AUTHORIZE commands are entered, not after the termination of AUTHORIZE. (For this reason, you should not enter temporary values with the intent of fixing them later in the run.)

Initially, you should make the following modifications to the UAF:

- SYSTEM, FIELD, and SYSTEST accounts -- Change the passwords on these accounts immediately. Use obscure passwords of six characters or more and keep changing them on a regular basis. These accounts permit access to the system that the general user should not have. As an alternative to changing the password, you could specify /FLAGS=DISUSER with AUTHORIZE, especially if the account usage is infrequent. To enable the account when it is needed, run AUTHORIZE and specify /FLAGS=NODISUSER.

- SYSTEM account -- You use this account only for system functions such as performing back-ups and installing maintenance updates. The account comes to you with full privilege, so you must exercise caution in using it. For example, because you have BYPASS privilege, the system will not prevent you from deleting any file no matter what its protection. If you type an incorrect name or spurious asterisk, you may destroy files that you need to keep. For this reason, you should use another account with fewer privileges for day-to-day miscellaneous use of the system.

If you want to receive mail on this account, you can define a system-wide logical name in the site-specific start-up command procedure (see Chapter 7) to equate SYSTEM to the user name of the system manager's account.

As a general rule, do not make any other changes in this UAF record; installation of VAX/VMS maintenance releases and optional software products depends on certain values in this record.

- DEFAULT record -- You may want to change several fields in this account, as demonstrated below:

```
UAF>MODIFY DEFAULT/DEVICE=DISK$USER/PGFLQUOTA=25000
```

The default device is set to the name most commonly used for user accounts that will be added. Likewise the page file quota value is set to a typical value for most users at the site.

## 2.4 ADDING A USER ACCOUNT

Accounts are of two general types:

- Interactive -- A person using an interactive account has access to the system software (command interpreters, compilers, utilities, and so on) and can perform work of a general nature (program development, text editing, and so on). Normally, such an account is considered individual -- that is, only one person can use it.

- Turnkey -- A person using a turnkey account has access only to user software and can only perform strictly limited work. Normally, such an account is considered functional -- that is, anyone who needs to perform the particular work can use it. As an example, you might develop an inventory system. Anyone whose job entails inventory control can access your system, but cannot access other systems or the base software.

The suggested procedures for adding a user account are as follows:

1. Determine a user name and password
2. Determine a user identification code (UIC)
3. Decide where the account's files will reside
4. Use the Disk Quota Utility (DISKQUOTA) to add a disk quota entry for this UIC, if disk quotas are in effect
5. Create a first-level directory on the appropriate volume
6. Establish any login/logout command files
7. Invoke the Authorize Utility and add the account

Perform the procedures in the order shown.

For consistent results, you can incorporate the steps into a command procedure. Figure 2-1 illustrates a command procedure for adding a new user.

### 2.4.1 Name and Password

The usual conventions for naming an account are as follows:

- Interactive -- The last name of the person using the account
- Turnkey -- A word that describes the function of the account

For example, an interactive account for Robert Jones would typically be named JONES while a turnkey account for the inventory system would typically be called INVENTORY.

For interactive accounts, it is best to let the person using the account control the password. You provide a simple password (USER or the person's first name, for example) and tell the person to change the password with the DCL command SET PASSWORD. Only the person using the account need know the password. You should encourage persons with sensitive accounts to set obscure passwords of six characters or more and to change them occasionally.

```
$ !
$ !   ADD A NEW USER TO THE SYSTEM AUTHORIZATION FILE
$ !
$ USERDISK = "WRKD$:"                    ! DEFAULT DISK FOR NEW USERS
$ UAF = "$AUTHORIZE"
$ ON CONTROLY THEN GOTO CLEANUP
$ ON WARNING THEN GOTO CLEANUP
$ OLDDIR = F$LOGICAL("SYS$DISK") + F$DIRECTORY()'
$ PREVPRIV = F$SETPRV("SYSPRV")
$ IF .NOT. F$PRIVILEGE("SYSPRV") THEN GOTO NOPRIV
$ SET DEFAULT SYS$SYSTEM
$ INQUIRE USERNAME "Username"
$ INQUIRE FULLNAME "Full name"
$ SET TERMINAL/NOECHO
$ INQUIRE PASSWORD "Password [''Username']"
$ SET TERMINAL/ECHO
$ IF PASSWORD .EQS. "" THEN PASSWORD = USERNAME
$GET_GRP:
$ INQUIRE GRP "UIC Group Number"
$ IF GRP .EQS. "" THEN GRP = "*"
$ WRITE SYS$OUTPUT ""
$ WRITE SYS$OUTPUT "Determine the UIC from the following listing:"
$ WRITE SYS$OUTPUT ""
$ UAF SHOW ['GRP',*]/BRIEF
$ INQUIRE UIC
$ IF UIC .EQS. "" THEN GOTO GET_GRP
$ IF F$LOCATE("[",UIC) .EQ. F$LENGTH(UIC) .AND. -
     F$LOCATE("<",UIC) .EQ. F$LENGTH(UIC) THEN UIC = "[" + UIC + "]"
$ INQUIRE ACCOUNT "Account Name [VMS]"
$ IF ACCOUNT .EQS. "" THEN ACCOUNT = "VMS"
$ INQUIRE PRIVS "Privileges [NONE]"
$ IF PRIVS .NES. "" THEN PRIVS = "/PRIV=(" + PRIVS + ")"
$ USERDIR = F$EXTRACT(0,9,USERNAME)
$ INQUIRE TMP "Login Directory [''USERDIR']"
$ IF TMP .NES. "" THEN USERDIR = TMP
$ INQUIRE TMP "Login Device [''USERDISK']"
$ IF TMP .NES. "" THEN USERDISK = TMP
$ DQUOTA = 0
$ IF F$SEARCH("''USERDISK'[0,0]QUOTA.SYS") .EQS. "" THEN GOTO NQ0
$ DQUOTA = 1
```

**Figure 2-1: Command Procedure Template for Adding a User**

```
$ INQUIRE QUOTA "Disk Quota [1000]"
$ IF QUOTA .EQS. "" THEN QUOTA = 1000
$ INQUIRE OVERDRAFT "Overdraft Quota [100]"
$ IF OVERDRAFT .EQS. "" THEN OVERDRAFT = 100
$ OPEN/WRITE FILE SYS$LOGIN:ADDQUOTA.TMP
$ WRITE FILE "RUN SYS$SYSTEM:DISKQUOTA"
$ WRITE FILE "USE ''USERDISK'"
$ WRITE FILE "ADD ",UIC,"/PERMQUOTA=",QUOTA,"/OVERDRAFT=",OVERDRAFT
$ CLOSE FILE
$ @SYS$LOGIN:ADDQUOTA.TMP
$ DELETE SYS$LOGIN:ADDQUOTA.TMP;*/NOLOG
$NQO:
$ CREATE/DIRECTORY/OWNER_UIC='UIC'/PROTECTION=(S=RWE,O=RWE,G=RE,W) -
        'USERDISK'['USERDIR']/LOG
$ IF F$SEARCH("SYS$MANAGER:LGISAMPL.COM") .EQS. "" THEN GOTO ADDACC
$ COPY SYS$MANAGER:LGISAMPL.COM 'USERDISK'['USERDIR']LOGIN.COM
$ SET FILE/OWNER_UIC='UIC' 'USERDISK'['USERDIR']LOGIN.COM;
$ADDACC:
$ OPEN/WRITE FILE SYS$LOGIN:ADDUAF.TMP
$ WRITE FILE "RUN SYS$SYSTEM:AUTHORIZE"
$ WRITE FILE "ADD ",USERNAME,"/OWNER=""",FULLNAME,"""/ACCOUNT=",ACCOUNT,-
      "/DEVICE=''USERDISK'/DIRECTORY=[''USERDIR']/UIC=",UIC,PRIVS,=
        "/PASSWORD=",PASSWORD
$ CLOSE FILE
$ @SYS$LOGIN:ADDUAF.TMP
$ DELETE SYS$LOGIN:ADDUAF.TMP;*/NOLOG
$CLEANUP:
$ SET TERMINAL/ECHO
$ PREVPRIV = F$SETPRV(PREVPRIV)
$ SET DEFAULT 'OLDDIR'
$ EXIT
$NOPRIV:
$ WRITE SYS$OUTPUT "You need SETPRV or SYSPRV privilege to run this procedure"
$ GOTO CLEANUP
```

**Figure 2-1 (Cont.): Command Procedure Template for Adding a User**

For turnkey accounts, the sensitivity of the account should determine the type of password. For example, the password for a payroll application should be obscure, while the password for a suggestions account might not even be required; it could be null. For all turnkey accounts, you should prohibit users from changing the password by specifying /FLAG=LOCKPWD when you add the account with the Authorize Utility. You should change the password whenever you feel it might be compromised (for example, if a person using the account moves to another job).

### 2.4.2 User Identification Code

See Chapter 3 for a detailed discussion of the UIC and its use. In general, you should assign each account a unique UIC, and you should assign accounts the same group number if they perform similar work, access the same files frequently, and/or use many of the same logical names.

### 2.4.3 User Directory and Default File Specification

If disk quotas are in effect for the volume, run the Disk Quota Utility to add an entry for the new UIC (see Chapter 5).

For each interactive account, you should create a first-level directory (using the DCL command CREATE/DIRECTORY) under which the interactive user can create and maintain files and subdirectories. Make the owner of the directory the UIC you have decided upon for the new account. Typically, the name of the account is also used for the first-level directory. For example, if you have decided upon an account name of JONES and a UIC of [014,006], you would issue the following DCL command to create a first-level directory for the account on the volume DISK$USER:

```
$ CREATE/DIRECTORY DISK$USER:[JONES]/OWNER_UIC=[014,006]-
$_/PROTECTION=(S:RWE,G:RE,O:RWE,W)
```

All access is denied to world users -- the typical protection specification for first-level directories. Users can further protect their files and subdirectories on an individual basis with the DCL command SET PROTECTION.

The volume on which the directory is established depends, of course, on which devices you reserve for interactive accounts and the available space on each.

The default file specification you provide the new account (when you run AUTHORIZE) should be the name of the device and the name of the first-level directory you used in the DCL command CREATE/DIRECTORY.

A turnkey account may or may not require the creation of a first-level directory, depending on the nature of the user system. Where the user system does use files in a particular directory, that directory should be made the default directory specification. For example, if the inventory system uses the files DISK$DATA:[INV]STOCK1.DAT and DISK$DATA:[INV]STOCK2.DAT, the default device specification should be DISK$DATA: and the default directory specification should be [INV].

### 2.4.4 Login Command Procedures

For interactive accounts, login command procedures contain commands commonly executed at the beginning of every user session, such as defining symbols for commands and command procedures, displaying messages and the time of day, and setting terminal characteristics. They are useful both for saving keystrokes and standardizing operations. In establishing login command procedures for interactive accounts, you have several choices:

- System -- You create and maintain a standard login command procedure in the system directory file possibly named SYS$MANAGER:SYLOGIN.COM. You then equate the logical name SYS$SYLOGIN to it so that whenever a user logs in, this procedure is executed, provided it exists.

- Individual -- For any or all accounts, you can name a separate login command procedure with the /LGICMD qualifier of AUTHORIZE. You can name the login command procedure anything you want. Once this definition is made, whenever the user logs in the procedure is executed, provided it exists.

- User-specified command file -- If individual login command procedures are not implemented, then by default, the system tries to execute the command file named LOGIN. This command file is developed and maintained by the user and should follow the following conventions:

  - Device and directory names should take the default file specification for the account

- File name should be LOGIN

- File type should be COM (or CMD, for accounts using the MCR command interpreter)

As an aid to new users, you might copy a login command procedure template into newly created first-level directories. However, to ensure proper ownership of the file, you must change the owner UIC of the file to that of the user. You do this with the DCL command SET FILE/OWNER_UIC.

Figures 2-2 and 2-3 illustrate typical system and user-specified login command procedures. Note that the user-specified login command files must all have the same name, LOGIN.

You can disable the CTRL/Y function (which suspends execution of the current image and invokes the command interpreter) to force execution of the complete login command procedure whenever the user logs in. You can do this with the DCL command SET NOCONTROL=Y. For interactive accounts, however, the login command procedure should, at some point, reset the CTRL/Y function with the DCL command SET CONTROL=Y.

```
$ V = F$VERIFY(0)
$START:
$ !
$ ON CONTROL_Y THEN GOTO START    ! Don't allow control/y out
$ SET NOON
$ !
$ !     Set default file protection back to the old default
$ !
$ SET PROTECTION=(SY:RWED,OW:RWED,GR:RWED,WO:RE)/DEFAULT
$ !
$ !     Make network jobs start faster
$ !
$ IF F$MODE() .EQS. "NETWORK" THEN GOTO EXIT
$ !
$ !     Enable CTRL/T handling by DCL
$ !
$ SET CONTROL=T
$ !
$ !     DEFINE FOREIGN COMMANDS FOR INSTALLED UTILITIES
$ !
$ SDA               :==    ANALYZE/CRASH_DUMP
$ USERS             :==    SHOW USERS
$ DISPLAY           :==    MONITOR PROCESSES/TOPCPU
$ NCP               :==    $NCP
$ INFO              :==    SHOW PROCESS/CONTINUOUS
$ SUSPEND           :==    SET PROCESS/SUSPEND
$ RESUME            :==    SET PROCESS/RESUME
$ SETNAME           :==    SET PROCESS/NAME
$ !
$ !     Define a symbol indicating whether the terminal
$ !     is on a dialup port
$ !
$ TT == F$GETDVI("TT","DEVNAM")-"-"
$ DIALUP == ((TT .GES. "TTG0:" .AND. TT .LES. "TTG4:") -
        .OR. (TT .GES. "TTH1:" .AND. TT .LES. "TTH4:") -
        .OR. (TT .EQS. "TTI5:"))
$ IF DIALUP THEN SET TERMINAL/INQUIRE
$ !
$EXIT:
$ IF V THEN SET VERIFY
$ EXIT
```

**Figure 2-2: Sample SYS$MANAGER:SYLOGIN.COM Login Command Procedure**

```
$ SET NOON
$ SET PROTECTION=(S=RD,O=RWED,G=R,W=R)/DEFAULT
$ !
$ ! Define abbreviations for often-used commands
$ !
$ DIR*ECTORY     :==     DIRECTORY/DATE/SIZE
$ PH*ONE         :==     PHONE/SCROLL
$ !
$ !
$ ! Other useful abbreviations
$ !
$ SHP            :==     SHOW PROCESS/PRIVILEGES
$ PRI*NT         :==     PRINT/NOTIFY
$ SHD            :==     SHOW DEFAULT
$ UP             :==     SET DEFAULT [-]
$ SP             :==     SET PROCESS/PRIVILEGES='P1
$ SQ             :==     SHOW QUEUE/BATCH/ALL/DEVICE
$ H*OME          :==     SET DEFAULT SYS$LOGIN
$ SUB*MIT        :==     SUBMIT/NOTIFY
$ SPC            :==     SHOW PROCESS/CONTINUOUS 'P1
$ SYS            :==     SHOW SYSTEM
$ DAY            :==     SHOW TIME
$ !
$ ! Set /LOG for all commands
$ !
$ BACK*UP        :==     BACKUP/LOG
$ DEL            :==     DELETE/LOG
$ LIB            :==     LIBRARY/LOG
$ PUR            :==     PURGE/LOG
$ REN            :==     RENAME/LOG
$ !
$ ! End of login.com processing
$ !
$ GOTO F$MODE()
$NETWORK:
$ EXIT
$INTERACTIVE:
$ VN             :==     SET TERMINAL/WIDTH=80
$ VW             :==     SET TERMINAL/WIDTH=132
$ EXPERT         :==     SET MESSAGE/NOFACIL/NOSEVER/NOIDENT
$ NOVICE         :==     SET MESSAGE/FACILITY/SEVERITY/IDENTIF
$ NOVICE
$ !
$ ! Network logins and users
$ !
$ SYSA           :==     SET HOST SYSA
$ SYSB           :==     SET HOST SYSB
$ SYSC           :==     SET HOST SYSC
$ EXIT                                  ! End of interactive login
$BATCH:
$ SET VERIFY                            ! End of batch login
$ EXIT
```

**Figure 2-3: Sample LOGIN.COM Login Command Procedure**

### 2.4.5 Logout Command Procedures

The system does not provide for automatic execution of a command procedure at logout time. However, you can provide for one as follows:

1. Create a command procedure for use at logout time (for example, SYS$MANAGER:SYLOGOUT.COM) or instruct your interactive users to create such a command procedure in their directories using a standard name such as LOGOUT.COM.

2. In the system-wide login command procedure, symbolically equate the abbreviation of the LOGOUT command most commonly used (for example, LO) to the execution of the logout command procedure, as demonstrated below:

```
$ LO*GOUT:==@SYS$MANAGER:SYLOGOUT
```

   The last line of the logout command procedure then uses an alternate form of the LOGOUT command, such as a LOGOUTNOW command. (You create any command name you like beginning with LO.) You cannot use the same abbreviation as used for the symbol (in this case LO) because it will start the procedure again. As an alternative, you could make the next to the last line of the procedure:

```
$ DELETE/SYMBOL/GLOBAL LOGOUT
```

   Note that the DCL command STOP, which is normally used to delete runaway processes, does not invoke any command procedure.

For turnkey accounts, the login command procedure normally directs the account user into the user system and logs the user out upon termination of the user system. Also, a turnkey login command procedure should explicitly assign the default disk device in case the user specifies another with the /DISK qualifier at login time. The login command procedure for the inventory system, for example, might consist of the following commands:

```
$ DEFINE SYS$DISK DISK$INVENT
$ RUN INVENTORY
$ LOGOUTNOW
```

The user program INVENTORY assumes control transparently to the person logging into the account.

For turnkey accounts, you should normally disable the CTRL/Y function and prevent the user from specifying an alternate command interpreter with the /CLI qualifier at login time (by specifying /FLAGS=CAPTIVE in AUTHORIZE). This action locks the person using the account into the user software. You may also want to disable mail and welcome notices with the /FLAGS=(DISNEWMAIL,DISWELCOME) qualifier.

### 2.4.6 Authorize Utility

Invoke AUTHORIZE and add the new account, using the information you have determined in the preceding steps. An example for an interactive user follows:

```
UAF>ADD JONES/PASSWORD=ROBERT/UIC=[014,006]-
_/DEVICE=DISK$USER/DIRECTORY=[JONES]-
_/LGICMD=DISK$USER:[NEWPROD]GRPLOGIN
_/OWNER="ROBERT JONES"/ACCOUNT=NEWPROD
```

The /OWNER and /ACCOUNT specifications are primarily for accounting purposes and can be omitted. The unspecified qualifiers typically take the defaults:

- Limits (/ASTLM, /BIOLM, /CPUTIME, /DIOLM, /ENQLM, /FILLM, /PGFLQUOTA, /PRCLM, /TQELM, /WSDEFAULT, /WSEXTENT, /WSQUOTA) -- See Chapter 4 for a discussion of limits; the default limits are normally adequate

- Priority (/PRIORITY) -- See Chapter 4 for a discussion of the processor priorities; the default is normally adequate for accounts not running real-time processes

- Privileges (/PRIVILEGES) -- See Chapter 4 for a discussion of privileges; the default privileges (TMPMBX, NETMBX) are normally adequate

- Primary and Secondary Login Times (/PRIMEDAYS, /P_RESTRICT, /PFLAGS, /S_RESTRICT, and /SFLAGS). By default, users are allowed to log in at any hour of any day. To override the setting of a particular day, you can use the DCL command SET DAY. Use this command if a holiday occurs on a day that would normally be treated as a primary day and you want it treated as a secondary day

- Quotas -- Control the amount of system resources the user can consume

- Command language interpreter -- Specify MCR if running in RSX-11M compatibility mode

An example for a turnkey account follows:

```
UAF>ADD INVENTORY/PASSWORD=CRAYONZ/UIC=[033,066]/DEVICE=DISK$INVENT-
_/DIRECTORY=[INV]/LGICMD=LOGIN/FLAGS=CAPTIVE/P_RESTRICT=18-8-
_/S_RESTRICT=0-23
```

## 2.5 DELETING A USER ACCOUNT

The main problem in deleting an account, especially an interactive account, is cleaning up the files used by the account. The following steps are suggested:

1. Copy (or have the outgoing user of the account copy) any files of value to the ownership of another account.

2. Delete the account's files and directories from the bottom up by:

   a. Locating and examining all subdirectories with the DCL command DIRECTORY [default...], where default is the name of the account's default directory;

   b. Deleting the files in each subdirectory and then deleting the subdirectory;

   c. Deleting the account's first-level directory. You can not delete a subdirectory without first deleting the files in it. Figure 2-4 demonstrates a command procedure that deletes an account's files from the bottom up. Note that if the user had the necessary privilege to write in other directories, you may need to employ the /USAGE qualifier with the Verify Utility to locate all the user's files.

3. Delete the account with the Authorize Utility.

4. Remove the user's disk quota entry from the disk quota file, if one existed, with the Disk Quota Utility.

```
$ !     DELTREE.COM - delete a complete directory tree
$ !
$ !     P1 = pathname of root of tree to delete
$ !
$ !     All files and directories in the tree, including
$ !     the named root, are deleted.
$ !
$ IF "'DELTREE'" .EQS. "" THEN DELTREE = "@SYS$LIBRARY:DELTREE"
$ ON CONTROL_Y THEN GOTO DONE
$ ON WARNING THEN GOTO DONE
$ DEFAULT = F$LOGICAL("SYS$DISK") + F$DIRECTORY()
$10:
$ IF P1 .NES. "" THEN GOTO 20
$ INQUIRE P1 "Root"
$ GOTO 10
$20:
$ IF F$PARSE(P1) .EQS. "" THEN OPEN FILE 'P1' ! Report error, exit
$ SET DEFAULT 'P1'
$LOOP:
$ FILESPEC = F$SEARCH("*.DIR;1")
$ IF FILESPEC .EQS. "" THEN GOTO LOOPEND
$ DELTREE [.'F$PARSE(FILESPEC,,,"NAME")']
$ GOTO LOOP
$LOOPEND:
$ IF F$SEARCH("*.*;*") .NES. "" THEN DELETE *.*;*
$ DIR = (F$DIRECTORY()-"]"-">")-(F$PARSE("[-]",,,-
        "DIRECTORY")-"]"-">")-"."
$ SET PROTECTION=WORLD:RWED [-]'DIR'.DIR;1
$ DELETE [-]'DIR'.DIR;1
$DONE:
$ SET DEFAULT 'DEFAULT'
```

**Figure 2-4: Command Procedure Template for Deleting an Account's Files**

If you never assign multiple users the same UIC, you can use the Backup Utility (see the VAX-11 Utilities Reference Manual) to remove the user's files, even if they are scattered about the directory structure. You would use a command of this form:

```
BACKUP/DELETE PUBLIC:[...]/OWNER=[uic] MTA0:FRED
```

This BACKUP command copies and deletes only those files owned by the specified UIC.

## 2.6 DISABLING A USER ACCOUNT

If it becomes necessary to disable an account from usage, but it is presently undesirable to remove the account, use AUTHORIZE to set the disable user flag (/FLAGS=DISUSER). If the user is logged in, the account is only disabled after the user logs out.

Disabling a powerful yet infrequently used account in this way provides an extra security measure by eliminating the risk of guessed passwords.

## 2.7 ALTERNATE LOGIN PROCEDURES

You can also implement the following alternatives to normal login activities:

- Inaccessible UAF -- If the UAF is locked, disabled, or not present you can log in on the console terminal with any name and password. The system assigns the following values to such a user:

  - Name -- your user name
  - UIC -- [001,004]
  - Command interpreter -- DCL
  - Login flags -- None
  - Priority -- Value of the system parameter DEFPRI
  - Resources -- Values of the PQL system parameters
  - Privileges -- All

  The process name is normally the name of the device on which the user logged in, that is, _OPA0:.

- Alternate UAF -- At bootstrap time, you can select an alternate UAF named SYS$SYSTEM:SYSUAFALT.DAT by using a system parameter file in which the UAFALTERNATE parameter has a value of 1, or by changing this value to 1 during a conversational bootstrap operation. (See Chapter 11 for information on creating system parameter files.) Naturally, a file named SYSUAFALT in UAF format must exist at bootstrap time. You can create or modify such a file with the following DCL commands:

```
$ SET DEFAULT SYS$SYSTEM
$ DEFINE SYSUAF SYSUAFALT
$ RUN AUTHORIZE
```

  Any time after booting the system with an alternate UAF, you can switch to the standard UAF (that is, SYS$SYSTEM:SYSUAF.DAT) with the following DCL command:

```
$ DEASSIGN SYSUAF/SYSTEM
```

  (The system initially changes to the alternate UAF by assigning SYSUAF as a logical name for SYSUAFALT.) If the UAFALTERNATE parameter is set at bootstrap time and no alternate UAF exists, the system behaves as an open system.

  You can also switch to the alternate UAF after bootstrap time with the following DCL command:

```
$ DEFINE/SYSTEM SYSUAF SYSUAFALT
```

  (For this mode of operation, the alternate UAF can have any file name.)

# CHAPTER 3

# UIC-BASED PROTECTION

VAX/VMS provides two structures to control the access that users have to processes and objects on the system (these objects include files, mailboxes, image and data sections, common event flag (CEF) clusters, volumes, and some non-file devices):

- User identification code (UIC) -- A UIC is a two-part identifier initially assigned to a user account in the user authorization file and passed along to processes and objects created by or on behalf of that user. The two parts of the identifier are called the group number and the member number. Every object in the system receives an owner UIC. In most cases this is the UIC of the creating process.

- Protection mask -- In addition to a UIC, an object is also assigned a protection mask. The protection mask defines the types of access permitted various categories of user to the object. For example, a protection mask for a data file might permit all users to read the file but only the owner of the file to write to it.

When a user attempts to access an object, the relationship of the user's UIC to the owner UIC of the object is examined. Depending on the outcome of this check, the user falls into one or more of the following four categories:

- Owner -- A user whose UIC is identical with the UIC of the object. The owner of an object is ordinarily (but not necessarily) the creator of the object.

- Group -- A user whose group number is identical with the group number of the object.

- System -- A user whose group number is between 1 and the value of the MAXSYSGROUP system parameter (10 octal, by default), inclusive. Also, any user with the SYSPRV privilege has the access rights of a system user.

- World -- Any user. This category includes (but does not exclude) users who are not owners of the object, are not members in the same group as the owner, and are not system users. In most cases it also includes users of DECnet-VAX who are performing operations from another network node.

The types of access are:

- Read -- Read files, mailboxes, and sections; read from volumes

- Write -- Write to files, mailboxes, and sections; write to volumes
- Execute -- Execute executable files; look up directory entries without wild card characters; create directories on disk volumes
- Delete -- Delete files
- Create -- Create files on structured volumes
- Allocate -- Allocate non-file, nonshareable devices
- Physical I/O -- Perform physical I/O on foreign volumes and non-file, shareable devices
- Logical I/O -- Perform logical I/O operations on foreign volumes and non-file, shareable devices

The use of four categories of user and these types of access permits a range of options in protecting objects. The system, group, and world categories are also used to apply special access controls to global sections, common event flag clusters, and logical name tables, and to regulate interprocess communications.

NOTE

A process with BYPASS privilege has complete access to all structures and devices regardless of the values of UICs and protection masks.

## 3.1 SPECIFICATION OF UIC

A UIC consists of two numbers, each of which can have a value in the range of 1 through 377 octal (because each number is restricted to three octal digits). The first number is the group number, the second the member number. By convention, the UIC [0,0] is not used.

The UIC consists of one longword divided into two equal parts: the high-order word contains the group number; the low-order word contains the member number.

DCL commands and system utilities require that UICs be specified as follows:

- The entire value must be enclosed in square brackets ([ ]).
- The group number must be stated first, followed by a comma, followed by the member number.
- The group and member numbers must each be specified as from one to three octal digits.

The following examples demonstrate several UICs in UIC notation and their corresponding longword values in hexadecimal:

| UIC | Hexadecimal Value | Meaning |
|---|---|---|
| [1,4] | 00010004 | Group 1, member 4 |
| [22,11] | 00120009 | Group 22, member 11 |

| UIC | Hexadecimal Value | Meaning |
|---|---|---|
| [377,377] | 00FF00FF | Group 377, member 377 |
| [101,67] | 00410037 | Group 101, member 67 |

The Authorize and Disk Quota utilities also permit the asterisk (*) wild card character:

- [*,*] means all UICs (AUTHORIZE handles these UICs in ascending numerical order)
- [n,*] means all UICs in group n (AUTHORIZE handles these UICs in ascending numerical order)
- [*,n] means all UICs with a member number of n

In calls to the system services, the UIC is specified as a longword, where bits 0-15 contain the member number and bits 16-31 contain the group number. For example, a programmer calling a system service might specify the UIC [22,11] as ^X00120009.

The following UICs have special meanings:

| UIC | Meaning |
|---|---|
| [1,3] | File system processes |
| [1,4] | System manager |
| [1,6] | Error logger |
| [1,7] | System test |
| [1,10] | System maintenance |

DIGITAL recommends that group numbers 1 through 10 be reserved for system managers and privileged users; that is, users in the system users category. (The MAXSYSGROUP system parameter sets the upper bound of the group number for system users.)

## 3.2 SPECIFICATION OF PROTECTION

A protection mask consists of four fields, each of which has four indicators. Each field applies to one category of user:

- Field 1 -- System users
- Field 2 -- Owner users
- Field 3 -- Group users
- Field 4 -- World users

Each indicator represents one category of access:

- Indicator 1 -- Read or allocate access
- Indicator 2 -- Write access
- Indicator 3 -- Delete or logical I/O access
- Indicator 4 -- Execute, create, or physical I/O access

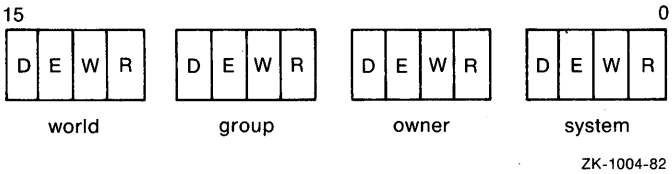
If an indicator is clear, the user is permitted access. If an indicator is set, the user is denied access. Table 3-1 lists the logical structure of the protection mask for the various types of objects. Note that for some objects certain indicators are meaningless.

Table 3-1: Logical Structure of Protection Mask

| Field / Indicator | 1 System 1 2 3 4 | 2 Owner 1 2 3 4 | 3 Group 1 2 3 4 | 4 World 1 2 3 4 |
|---|---|---|---|---|
| Disk Files | R W E D | R W E D | R W E D | R W E D |
| Mailboxes | R W L P | R W L P | R W L P | R W L P |
| Global Sections | R W | R W | R W | R W |
| Structured disk volumes | R W C D | R W C D | R W C D | R W C D |
| Structured tape volumes | R W E D | R W E D | R W | R W |
| Foreign volumes | R W L P | R W L P | R W L P | R W L P |
| Non-file, shareable devices | L P | L P | L P | L P |
| Non-file, non-shareable devices | A | A | A | A |

Legend: R = read; W = write; E = execute; D = delete;
A = allocate; C = create; P = physical I/O;
L = logical I/O

Physically, a protection mask consists of one word divided into four equal parts (four bits each). The low-order bit in each part represents the first indicator, the next bit represents the second indicator, and so on. A value of 0 means the indicator is clear, and 1 means the indicator is set. The protection mask for a file, for example, would have the following physical appearance:




ZK-1004-82

DCL commands require that protection masks be specified as follows:

- The user categories (the fields) are represented by the words SYSTEM, OWNER, GROUP, and WORLD, which can be abbreviated to any length.
- The types of access (the indicators) are represented by the letters R (read), W (write), E (execute and create), D (delete), L (logical I/O), and P (physical I/O).

- The types of access for a user category are written as the user category, followed by a colon, followed by the access indicators allowed that category of user. For example, if world users are allowed only read and execute access to a file, the protection code is written as WORLD:RE. The access indicators can be written in any order, but best form is to write them in the order R, W, E, D, or R, W, L, P. Whenever you omit the colon and the access indicator, the category of user is denied all access.

- User categories and their respective access indicators are separated by commas and enclosed in parentheses. They can be written in any order, but it is best to write them in the order SYSTEM, OWNER, GROUP, WORLD. For example, if system and owner users are allowed all types of access to a file, while group users are restricted to read and execute access, and world users have no access, the protection code is written as (S:RWED,O:RWED,G:RE,W). However, no commas or parentheses are necessary if only one user category is specified.

- Omission of a user category for file protection generally results in no access being granted to that category. However, the DCL command SET PROTECTION retains the existing protection for an existing file and the default protection (discussed below) for a new file.

In calls to system services, the protection mask is specified as a word. For example, the programmer calling a system service might specify the protection code (S:RWED,O:RWED,G:RE,W) as ^XFA00 -- bits 9, 11, 12, 13, 14, and 15 are set.

## 3.3 USER ACCOUNTS AND PROCESSES

Each user's entry in the user authorization file (UAF) contains a UIC. You specify the UIC when adding or modifying a user entry with the Authorize Utility (see Chapter 2).

When a user logs in to the system, VAX/VMS creates a detached process on behalf of the user and assigns to the process the UIC contained in the user's UAF account. Typically, the user process UIC does not change, although it can be changed with the DCL command SET UIC, which is available to users with CMKRNL privilege.

User-created detached processes require that a UIC be specified according to the method of creation, as follows:

- DCL command RUN (PROCESS) -- The /UIC qualifier specifies the UIC

- $CREPRC system service -- The uic argument specifies the UIC

In fact, the presence of either the qualifier or argument determines that the created process is a detached process rather than a subprocess. The following example creates a detached process:

```
$ RUN/UIC=[22,11] DISK$USER:[JONES]MASTUP
```

Creation of a detached process requires the DETACH privilege (see Chapter 4).

A subprocess (created when neither a UIC qualifier nor an argument is specified) takes the same UIC as the creating process. The following example creates a subprocess:

```
$ RUN/PROCESS_NAME=MASTUP DISK$USER:[JONES]MASTUP
```

## 3.4 FILES-11 FILES

By default, a Files-11 file created through any of the DCL commands CREATE, CREATE/DIRECTORY, or COPY takes the UIC of its creator, while an appended file (APPEND command) or renamed file (RENAME command) retains the UIC of the base file. A user with SYSPRV or BYPASS privilege can specify a different UIC in the DCL commands CREATE and CREATE/DIRECTORY, as demonstrated below:

```
$ CREATE CUSTLIST.TXT/OWNER_UIC=[22,11]
```

### 3.4.1 Default Protection of Files

With a few exceptions, newly created Files-11 files receive the user's default file protection. The exceptions are described in the following paragraphs.

Copied files receive the protection of the original file if the user does not give the output file a new name, as in the following example:

```
$ COPY  [FRED]TESTFILE.DAT  [GEORGE]
```

However, the copied file receives the user's default file protection if the user assigns the file a new name, as in this example:

```
$ COPY  [FRED]TESTFILE.DAT  [GEORGE]OUTPUT1.DAT
```

Renamed files preserve their ownership and protection. A file that is appended to another file acquires the ownership and protection of the base file.

The system parameter RMS_FILEPROT determines the initial setting of every user's default file protection. RMS_FILEPROT is described in Chapter 10 and is set by default to (S:RWED,O:RWED,G:RE,W). The user can change the default protection with the DCL command SET PROTECTION/DEFAULT. The following example resets the protection default to deny read and execute access to group users:

```
$ SET PROTECTION=(S:RWED,O:RWED,G,W)/DEFAULT
```

The user-specified default remains in effect until the user logs off or resets the default again.

### 3.4.2 Explicit Protection of Files

The user can specify protection explicitly for new, copied, and appended files with the /PROTECTION qualifer of the DCL commands CREATE, COPY, and APPEND, as illustrated in the following example:

```
$ CREATE MAST12.TXT/PROTECTION=(S:RWED,O:RWED,G,W)
```

In this example, group and world users are denied access to MAST12.TXT.

The user can change the protection of existing files with the DCL command SET PROTECTION, as demonstrated below:

```
$ SET PROTECTION=(S:RWED,O:RWED,G,W) MAST12.TXT
```

The user must own a file, or possess the SYSPRV or BYPASS privilege, to set protection on the file.

### 3.4.3 Directory Protection

By default, a new directory file receives the protection mask of its parent, but with all delete access removed. The final default is the protection of the volume's MFD, which is usually (S:RWE,O:RWE,G:RE,W:E). The user can explicitly set protection for a new directory by specifying the /PROTECTION qualifer with the DCL command CREATE/DIRECTORY. After the directory is created, the user can change its protection with the DCL command SET PROTECTION.

Suppose that the user has been using the subdirectory [.XMAST] with default protection and now wants to delete it. The protection must first be changed to allow the deletion:

```
$ SET PROTECTION=O:RWED XMAST.DIR;1
```

Now the user can delete the subdirectory, provided it contains no files.

### 3.4.4 Mail File Protection

The Personal Mail Utility (MAIL) protects the file MAIL.MAI with a mask of (S:RWD,O:RW,G,W). Group and world users cannot access a user's mail at all, and the owner cannot delete the mail file (but the user can delete messages with the DELETE command of MAIL; MAIL deletes the file if the user deletes all messages). Note that MAIL protects all other files with a file type of MAI with a mask of (S:RWD,O:RWD,G,W).

## 3.5 MAILBOXES

A mailbox is assigned the UIC of its creating process.

A protection mask is specified at mailbox creation time with the promsk argument of the Create Mailbox and Assign Channel ($CREMBX) system service, as illustrated below:

```
$CREMBX_S-                ; CREATE MAILBOX
   PRMFLG=#0,-            ; TEMPORARY MAILBOX
   CHAN=MBXCHAN,-         ; FOR CHANNEL NUMBER
   MAXMSG=MBUFLEN,-       ; MAX MSG SIZE
   BUFQUO=#384,-          ; DYNAMIC MEMORY
   PROMSK=#^XF000,-       ; NO WORLD USERS
   LOGNAM=MBLOGNAM        ; LOGICAL NAME
```

This example grants read and write access to system, owner, and group users, and denies read and write access to world users. If the protection mask is specified as 0 or is not specified, all categories of users are granted read and write access to the mailbox.

## 3.6 SECTIONS

A process can create sections of the following types:

- Private -- Code and data to which only the creator of the section can map
- Global -- Code and data to which members of the creator's group can map
- System global -- Code and data to which all processes can map; requires the SYSGBL privilege (see Chapter 4)

The programmer specifies the type of section in the flags argument to the Create and Map Section ($CRMPSC) system service, as illustrated below:

```
$CRMPSC_S                    ; CREATE AND MAP SECTION
   INADR=MYADR,-             ; REQUESTED ADDRESSES
   RETADR=SYSADR,-           ; ACTUAL ADDRESSES
   GSDNAM=GSD_DESCG,-        ; GLOBAL SYMBOL
   FLAGS=#SEC$M_GBL,-        ; GLOBAL SECTION
   CHAN=SFAB+FAB$L_STV,-     ; CHANNEL NUMBER
   PAGCNT=#4                 ; FOUR PAGES
```

Sections created with the Install Utility are always system global.

A global (or system global) section is assigned the UIC of its creating process.

A protection mask can be specified for a global (or system global) section with the prot argument of the $CRMPSC system service, as illustrated below:

```
$CRMPSC_S-                              ; CREATE AND MAP SECTION
   INADR=MYADR,-                        ; REQUESTED ADDRESSES
   RETADR=SYSADR,-                      ; ACTUAL ADDRESSES
   FLAGS=#<SEC$M_WRT_!SEC$M_GBL>,-      ; READ/WRITE SECTION
   CHAN=SFAB+FAB$L_STV,-                ; CHANNEL NUMBER
   PAGCNT=#4,-                          ; FOUR PAGES
   PROT=#^XFF0F                         ; OWNER ONLY
```

This example grants read and write access to the owner, and denies read and write access to system, group, and world users. (Execute and delete access -- that is, the two high-order bits in each part of the mask -- are meaningless for sections. The mask in the example could be specified as #^X3303.) If the protection mask is specified as 0 or is not specified, all categories of users are granted read and write access to the section.

Specification of a protection mask can limit the scope of a global or system global section. Conversely, specification of a private or global section can limit the allowances of a protection mask. (A protection mask for a private section is meaningless.)

## 3.7 COMMON EVENT FLAG CLUSTERS

A common event flag cluster is assigned the UIC of its creating process. One of the following types of event flag clusters can be permanently associated with the process:

- Local event flag cluster -- Only the creating process can use the cluster

- Common event flag (CEF) cluster -- Members of the creator's group can use the cluster

The programmer specifies a CEF cluster with the Associate Common Event Flag Cluster ($ASCEFC) system service, as illustrated in the following example:

```
$ASCEFC_S-                ; ASSOCIATE WITH CEF
   EFN=#65,-              ; EVENT FLAG NUMBER
   NAME=CLUSTER           ; CLUSTER NAME
```

The programmer can restrict the use of CEF clusters to processes with the creating process's UIC by setting the value of the prot argument at 1 (the prot argument is not a protection mask):

```
$ASCEFC_S                 ; ASSOCIATE WITH CEF
   EFN=#65,-              ; EVENT FLAG NUMBER
   NAME=CLUSTER,-         ; CLUSTER NAME
   PROT=#1                ; PROCESSES WITH SAME UIC
```

## 3.8 STRUCTURED VOLUMES

For information on structured tape volumes, see the VAX/VMS Magnetic Tape User's Guide.

By default, a Files-11 disk volume is assigned the UIC of the user who initializes it. However, the user can specify a different UIC, as illustrated below:

```
$ INITIALIZE DBA1: MASTER/OWNER_UIC=[22,11]
```

In addition, the user who owns the volume or any user with VOLPRO privilege (see Chapter 4) can mount the volume with a different UIC, as follows:

```
$ MOUNT DBA1: MASTER/OWNER_UIC=[1,4]
```

However, a volume that is initialized or mounted with the /SYSTEM qualifier always receives a UIC of [1,1], and a volume initialized or mounted with the /GROUP qualifier receives a UIC of [g,0], where g is the group number of the process initializing or mounting the volume.

By default, no protection is applied to newly initialized structured disk volumes. (That is, all categories of users are permitted all types of access.) The user can specify protection with the /PROTECTION qualifier of the DCL command INITIALIZE, as illustrated below:

```
$ INITIALIZE DBA1: MASTER-
$_/PROTECTION=(S:RWED,O:RWED,G:R,W:R)
```

This example limits group and world users to read access on the volume. (For disk volumes, the designation E means create access -- the right to create files on the volume.)

The user can respecify volume protection at mount time with the /PROTECTION qualifier of the DCL command MOUNT, as illustrated below:

```
$ MOUNT DB1 MASTER-
$_/PROTECTION=(S:RWED,O:RWED,G:RWED,W:R)
```

The user changing protection must be the owner or must have VOLPRO privilege.

A user initializing or mounting a disk volume can also specify the scope of its accessibility with the following qualifiers:

| Qualifier | Meaning |
|---|---|
| /GROUP | System, owner, and group users have RWED access to the volume. World users have no access. (At initialization, specify INITIALIZE/NOSHARE/GROUP to grant RWED access to system, owner, and group users; omitting /NOSHARE grants RWED access to all categories of user.) |
| /SHARE | All users have RWED access to the volume. |
| /NOSHARE | System and owner users have RWED access to the volume. Group and world users have no access (except as noted above under /GROUP). |
| /SYSTEM | All users have RWED access to the volume, but only system users can create first-level directories. |

At initialization time, these qualifiers override any protection mask specified. At mount time, however, the protection mask is the limiting factor. When mounting a volume, you must have the GRPNAM privilege to use the /GROUP qualifier; likewise, you must have the SYSNAM privilege to use the /SYSTEM qualifier.

## 3.9 FOREIGN VOLUMES

By default, a foreign disk or tape volume is assigned the UIC of the user who mounts it. The user can specify a different UIC with the /OWNER_UIC qualifer of the DCL command MOUNT.

By default, system and owner users are granted all access to foreign volumes when they are mounted, while group and world users are denied all access. The user can specify protection with the /PROTECTION qualifier of the MOUNT commmand, as illustrated below:

```
$ MOUNT DMA0: CUSTLIST/FOREIGN-
$_/PROTECTION=(S:RWLP,O:RWLP,G:R,W:R)
```

This example grants group and world users read access to the volume.

A user mounting a foreign disk volume can also specify the scope of its accessibility with the following qualifiers:

| Qualifier | Meaning |
|---|---|
| /GROUP | System, owner, and group users have RWLP access to the volume. World users have no access. |
| /SHARE | All users have RWLP access to the volume. |
| /NOSHARE | System and owner users have RWLP access to the volume. |
| /SYSTEM | All users have RWLP access to the volume. |

The /GROUP qualifier requires GRPNAM privilege. The /SYSTEM qualifier requires SYSNAM privilege.

## 3.10 NON-FILE DEVICES

By default, UIC and protection are associated with non-file devices. You can associate a UIC and protection code with all terminals on the system through the TTY_OWNER and TTY_PROT system parameters, or you can remove the protection associated with these devices. (The values must be specified prior to VAX/VMS initialization with the System Generation Utility (SYSGEN), or during initialization with the SYSBOOT Utility.) The following example restricts the allocation of terminals to system users:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN SYSGEN
SYSGEN>  USE AUTOGEN.PAR
SYSGEN>  SET TTY_OWNER %X10004
SYSGEN>  SET TTY_PROT %X1110
SYSGEN>  WRITE AUTOGEN.PAR
SYSGEN>  EXIT
```

These changes take effect when you specify AUTOGEN.PAR during the next conversational boot.

Only allocate access (the low-order bit of each 4-bit part of the mask) applies to non-file, non-shareable devices.

The example does not prevent users from logging in, since interactive terminals are allocated by the system. When a user logs in, the system associates the user's UIC with the terminal. The example does prevent users without system access rights from allocating terminals that are not in use.

The user with OPER privilege can set protection for terminals and other non-file devices on an individual basis with the DCL command SET PROTECTION/DEVICE, as illustrated below:

```
$ SET PROTECTION=(S,O:R,G,W)-
$_/DEVICE-
$_/OWNER_UIC=[22,11]-
$_TTA1
```

This example permits only a user with UIC [22,11] to allocate the terminal TTA1.

## 3.11 DEVICE ALLOCATION

A user can prevent others from accessing a device by allocating the device. In the following example, a user allocates a disk device before mounting a private volume on the device:

```
$ ALLOCATE DMA1:
$ MOUNT DMA1: XMAST
```

## 3.12 INTERPROCESS CONTROL

The actions of some commands and system services are limited by GROUP and WORLD privilege. GROUP privilege permits a user to perform the activities on all processes in the same group. WORLD privilege permits a user to perform the activities on all processes. The user with neither GROUP nor WORLD privilege can perform the activities only on processes with the same UIC. Tables 3-2 and 3-3 identify the affected commands and system services.

Table 3-2: DCL Commands Affected by GROUP and WORLD Privilege

| Command | Explanation |
|---|---|
| DELETE/ENTRY | Deleting batch or print queue entries queued by another member of the group requires GROUP privilege; by processes outside the group WORLD (or OPER) privilege. |
| SET PROCESS | Setting the priority, suspending, or resuming the process of another member of the group requires GROUP privilege; for processes outside the group, WORLD privilege is necessary |
| SHOW PROCESS/CONTINUOUS | Displaying process information for another process in the group requires the GROUP privilege; for processes outside the group, WORLD privilege is necessary |
| SHOW DEVICES/FILES/NOSYSTEM | Displaying the names of all files opened on a volume by another member of the group requires GROUP privilege; for those opened by processes outside the group, WORLD privilege is necessary |
| STOP/ENTRY | Stopping batch or print queue entries queued by another member of the group requires GROUP privilege; by processes outside the group, WORLD (or OPER) privilege. |

Table 3-3: System Services Affected by GROUP and WORLD Privilege

| System Service | Explanation |
|---|---|
| $FORCEX | Forcing image exit for another member of the group requires GROUP privilege; for a process outside the group, WORLD privilege. |
| $SETPRI | Setting the priority of another member of the group requires GROUP privilege; of a process outside the group, WORLD privilege. |
| $GETJPI | Getting job/process information for another member of the group requires GROUP privilege; for a process outside the group, WORLD privilege. |

(continued on next page)

Table 3-3 (Cont.): System Services Affected by GROUP and WORLD Privilege

| System Service | Explanation |
|---|---|
| $SUSPEND | Suspending another member of the group requires GROUP privilege; a process outside the group, WORLD privilege. |
| $RESUME | Resuming execution of another member of the group requires GROUP privilege; of a process outside the group, WORLD privilege. |
| $DELPRC | Deleting another member of the group requires GROUP privilege; a process outside the group, WORLD privilege. |
| $WAKE | Waking another member of the group requires GROUP privilege; a process outside the group, WORLD privilege. |
| $SCHDWK | Scheduling a wakeup request for another member of the group requires GROUP privilege; for a process outside the group, WORLD privilege. |
| $CANWAK | Cancelling a wakeup request for another member of the group requires GROUP privilege; for a process outside the group, WORLD privilege. |
| $SNDSMB | Affecting queue entries belonging to another member of the group requires GROUP privilege; of a process outside the group, WORLD or OPER privilege. |

## 3.13 LOGICAL NAMES

Logical names can be specified as process-only, group, and system-wide, as demonstrated in the following example:

```
$ DEFINE COMFILES/GROUP DISK$USER:[JONES.COMMON]
```

The group logical name COMFILES is created.

A user requires the GRPNAM privilege to create a group logical name and the SYSNAM privilege to create a system logical name. Once created, a group logical name is available to all members of the creator's group, while a system logical name is available to all users on the system. Process logical names are available only to the creator. Where logical names of different types are identical, group logical names take precedence over system logical names, and process logical names take precedence over group and system logical names.

## 3.14 STATUS INFORMATION

The following DCL commands provide status information concerning UICs and protection:

| Command | Status Information |
|---|---|
| SHOW PROCESS | UIC of user's detached process |
| SHOW SYSTEM | UICs of all active users |
| SHOW PROTECTION | User's default protection |
| SHOW PROCESS | User's default UIC |
| DIRECTORY/OWNER | Owner UIC of file |
| DIRECTORY/PROTECTION | Protection of file |
| SHOW DEVICES/FULL | Owner UIC of device or volume; protection of device or volume |

The Get Device Information ($GETDVI) system service returns the owner UIC and protection of a device or volume.

## 3.15 FORMING GROUPS

In setting up a group, you have two aims:

- Sharing -- Users who typically share data, and/or control one another's processes should be assigned to the same group.
- Protection -- Users who should not have access to each other's data or control over each other's processes should be assigned to separate groups.

In an interactive environment, group members typically consist of persons working on the same project or in the same department. In a turnkey environment, group members typically consist of accounts dedicated to the same user system.

A certain amount of control and protection exists among members of the same group:

- GROUP privilege -- Only a member with GROUP privilege can use the commands and system services listed in Tables 3-2 and 3-3 to affect other members of the group. You should be wary in granting this privilege. Probably the best policy is to grant the privilege only to the account that has supervisory control over the group and other accounts designated by the group supervisor.
- GRPNAM privilege -- Only a member with GRPNAM privilege can assign or deassign group logical names or mount group volumes. You should grant this privilege only to members who perform these operations.
- Protection mask -- The default protection provides read access by group members to one another's objects. However, members can either further protect their objects or allow more extensive access by resetting the default protection or by setting protection on an object-by-object basis.

You should analyze the protection requirements of your site if these requirements normally differ from the default protection provided by the system, reset the default protection using the system parameter RMS_FILEPROT (or direct your users to reset the default protection in their individual login command procedures; see Chapter 2). For example, if you want members of the same group to have write and delete access to one another's files, you could modify the default protection in each user's login command file as follows:

```
$ SET PROTECTION=GROUP:RWED/DEFAULT
```

## 3.16 SECURING SYSTEM DATA

The UIC [1,4] should own all system files. Protection on all system executable and library files should permit only read and execute access to the world. Protection on the following files should permit no access to the world:

```
SYS$SYSTEM:PAGEFILE.SYS
SYS$SYSTEM:SWAPFILE.SYS
SYS$SYSTEM:SYSDUMP.DMP
SYS$SYSTEM:SYSUAF.DAT
SYS$SYSTEM:RMTNODE.DAT
```

Since these files contain sensitive data, a user with even read access to them possesses a degree of control over the system. If you have an alternate system UAF (SYSUAFALT.DAT) or have added additional paging or swapping files, remember to verify that they are properly protected.

The software distribution kit provides the required protection for these files. However, you could easily compromise a sensitive system file by copying the files without applying proper protection. To prevent such a situation, you should always:

- Log in under the system manager account when copying system files.
- Choose the Backup Utility or DCL command COPY for making additional copies of system files. (The VAX-11 utilities DSC1 and DSC2 also preserve the protection originally assigned to files, but are not the preferred means.) Other utilities (RMS-11 Backup and Restore (BCK and RST), for example) should be avoided.
- Use the DCL command DIRECTORY/PROTECTION/OWNER to check the protection and owner of a sensitive file, after copying. You can change the protection, if necessary, with the SET PROTECTION command. Use the DCL command SET FILE/OWNER to change the owner, if necessary.
- If you enable disk quotas on public volumes, make sure you are logged in under the system manager account when you create the quota file. Also verify protection of the quota file ([0,0]QUOTA.SYS). Only system and owner users should be able to write to the quota file. If you want to permit users to look at the quotas and usage of others, permit only read access to world users; otherwise, provide no access to world users.

When you create UAF listing files with the Authorize Utility LIST command, do not leave them in public directories, such as SYS$SYSTEM.

## 3.17 SECURING SYSTEM DEVICES

You should protect interactive terminals and card readers from allocation by processes other than system processes with the following steps:

- System parameters -- Specify the value of TTY_PROT as %X1110 and TTY_OWNER as %X10004
- Variant terminals -- If you want some terminals to be allocated by nonsystem processes (for example, a user system might control some terminals directly), place SET PROTECTION/DEVICE commands for them in the site-specific start-up command procedure (see Chapter 7)
- Card readers -- Place SET PROTECTION/DEVICE commands for card readers in the site-specific start-up command procedure (see Chapter 7)

When a user logs in, the system allocates the user's terminal on behalf of the user. When the user is not logged in, however, the terminal is unallocated. If the terminal is left unprotected, an unprincipled user could allocate it to prevent its normal user from logging in, or worse, could run a login process in place of the system to learn the user's password or otherwise manipulate the user.

## 3.18 PROTECTING USER FILES

You should encourage all users of the system to observe the following guidelines:

- Default protection -- The default protection prevents world users from accessing your files for any purpose, and prevents group users from exercising write and delete access on your files. If you feel this amount of protection is not adequate, put an appropriate SET PROTECTION/DEFAULT command in your login command procedure.
- Sensitive files -- If you create a sensitive file, place additional protection on it with the SET PROTECTION command.
- Directory protection -- You can protect files within a subdirectory by placing additional protection on the subdirectory. In this regard, remember that if you cannot access a file due to a protection violation, and the protection on the file displayed by the DCL command DIRECTORY/PROTECTION does not appear to be the reason, check the protection on the file's directory.

NOTE

Protecting a directory protects the names of the files in that directory. It also protects the files for most practical purposes. However, it is possible for a user to access files without using the directory by writing a program that "guesses" file IDs. This process is time consuming, but effective. Therefore, you should protect sensitive files at both the directory and the file level.

- Directory deletion -- Before you give yourself delete access to a directory make sure (1) the directory is empty and (2) you intend to delete it immediately. (You cannot delete a directory with files in it.) The following example illustrates the proper sequence for the deletion of a subdirectory and its contents after the user has decided they are no longer needed:

```
$ DELETE [.XUPDATE]*.*;*          ! GOODBYE FILES
$ SET PROTECTION=O:D XUPDATE.DIR  ! GOODBYE PROTECTION
$ DELETE XUPDATE.DIR;1            ! GOODBYE DIRECTORY
```

- Copying to another's directory -- If you copy a file into another user's directory (which requires write access to that user's directory), the copied file has your UIC, not the UIC of the user in whose directory it now resides. Such a copy operation could produce a file that the recipient cannot immediately access (or fully access; for example, the user might be able to read the file, but not write to it or delete it). Rather than copy a file into another user's directory, have the other user copy the file. If you do copy a file into another user's directory, be sure to use the DCL command SET FILE/OWNER_UIC to establish the correct new owner.

- Device allocation -- If you are using a device (for example, a disk or tape drive) for a private volume, allocate the device so that other users cannot access it. Specifying protection at initialization time is also a good idea. The /NOSHARE qualifier restricts use of the volume to yourself, while /GROUP allows members of your group to access the volume. A /PROTECTION qualifier is also available.

You should be ever-conscious that a user who has access in any category, has access. For example, if XMAST.DAT has the protection mask (S:RWED,O:RWED,G:RE,W:RE), the following command makes no sense:

```
$ SET PROTECTION=G XMAST.DAT
```

The mask now denies access to group users, but since group users are also world users, they still have read and execute access.

# CHAPTER 4

# RESOURCE CONTROL

This chapter contains detailed descriptions of the resource control attributes that you can assign to a user when creating a record in the UAF:

- Limits on the use of reusable system resources
- Base priority used in scheduling the process that the system creates for the user
- Privileges allowing use of restricted and sensitive system functions

In addition, this chapter discusses the accounting log file.

## 4.1 LIMITS ON REUSABLE SYSTEM RESOURCES

Limits are set on system resources that can be reused, for example, the amount of memory that a process can have in use for I/O requests. Each user of the system is limited in the use of system resources. You set up these limits when you define the user to the system through the creation of a user's account in the UAF. Most limits restrict the use of system memory.

Limits can control the way in which a process shares its allotment of a resource with the subprocesses that it creates. Three methods for sharing resources are used: pooled, deductible, and nondeductible limits. If the limit on the use of a resource is pooled, a process and created subprocesses share the total limit on a first-come, first-served basis. If the limit on the use of a resource is deductible, a subprocess is allotted a portion of the total limit; the portion given to the subprocess is deducted from the total limit. If the limit is nondeductible, the subprocess is allotted the total limit of the creating process; there is no deduction from the allotment of the creating process.

In creating a UAF record, you assign values to the limits shown in Table 4-1. These limits are described in the following sections. Usually, you simply assign the default values for these limits. However, see Chapter 12 for a discussion of WSDEFAULT, WSEXTENT, and WSQUOTA.

Table 4-1 summarizes each of these limits, the suggested minimum value, and the type of limit.

Table 4-1: Summary of System Limits

| Limit | Description | Type[1] | Minimum Value |
|---|---|---|---|
| ASTLM | AST queue limit | N | 2 |
| BIOLM | Buffered I/O count limit | N | 2 |
| BYTLM | I/O byte count limit | P | 1024 |
| CPULM | CPU time limit | D | 10 |
| DIOLM | Direct I/O count limit | N | 2 |
| ENQLM | Enqueue quota | P | 4 |
| FILLM | Open file limit | P | 2 |
| PGFLQUOTA | Paging file limit | P | 256 |
| PRCLM | Subprocess creation limit | P | 0 |
| TQELM | Timer queue entry limit | P | 0 |
| WSDEFAULT | Default working set size | N | 50 |
| WSEXTENT | Working set extent | N | 50 |
| WSQUOTA | Working set quota | N | 50 |

1. N=nondeductible, D=deductible, P=pooled

### 4.1.1 AST Queue Limit (ASTLM)

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

- The number of asynchronous system trap (AST) requests 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 affects not only all system services that accept an AST address as an argument, but also the Schedule Wakeup ($SCHDWK) system service.

ASTLM is a nondeductible limit with a suggested typical value of 6.

### 4.1.2 Buffered I/O Count Limit (BIOLM)

The buffered I/O count limit (BIOLM) limits the number of outstanding buffered I/O operations permitted 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. Buffered operations for a user process include terminal I/O, card reader input, and unspooled printer output. During a buffered I/O operation, the pages containing the process-specified buffer need not be locked in memory.

BIOLM is a nondeductible limit with a suggested typical value of 6.

### 4.1.3 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. It also limits the number of mapping

windows the user can create as segmented (or cathedral) windows. Cathedral windows are primarily useful to reduce the overhead required to read large files.

BYTLM is a pooled limit with a suggested typical value of 8192.

### 4.1.4 CPU Time Limit (CPULM)

The CPU time limit (CPULM) limits the amount of CPU time that a user's process can use per interactive session or batch job.

The time must be specified in abbreviated delta format -- hh:mm:ss.cc.

CPULM is a deductible limit, but only applies to this instance or other instances of this user's processes. CPULM is not cumulative across separate sessions or batch jobs.

### 4.1.5 Direct I/O Count Limit (DIOLM)

The direct I/O count limit (DIOLM) limits the number of outstanding direct I/O operations permitted 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. Direct I/O operations for a user process typically include disk and tape I/O. The pages containing this buffer are locked in memory by the operating system during the direct I/O operation.

DIOLM is a nondeductible limit with a suggested typical value of 6.

### 4.1.6 Enqueue Quota (ENQLM)

The enqueue quota (ENQLM) limits the number of locks a process (and its subprocesses) can own. VAX-11 RMS uses the lock management facility to synchronize shared file access, global buffers, and record locks. Because VAX-11 RMS takes out one lock for every shared file, global buffer, and outstanding record lock, users who expect to perform large amounts of VAX-11 RMS file sharing should have ENQLM set to a large value. If your process performs extensive VAX-11 RMS file sharing without sufficient enqueue quota, you could receive the SS$_EXENQLM error message. Furthermore, if your system performs extensive VAX-11 RMS file sharing and the value of the LOCKIDTBL system parameter is too low, you could receive the SS$_NOLOCKID error message. Note that whenever you increase the value of LOCKIDTBL, you may have to increase the value of the RESHASHTBL system parameter (see Chapter 10).

For shared files the value of ENQLM should represent the number of files open as shared multiplied by the number of locks per process per file. If you use the default multibuffer counts, you can estimate the number of locks as 4 for indexed sequential files and 3 for relative files. If you use other than the default value for the multibuffer counts, you can estimate the number of locks per process per file as one per file plus the multibuffer count for that file plus the number of records locked (which is usually one). Use the DCL command SHOW RMS_DEFAULT to display the default multibuffer counts.

ENQLM is a pooled limit with a suggested typical value of 20.

### 4.1.7 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 pooled limit with a suggested typical value of 20. Note that each open file also requires at least 96 bytes of BYTLM.

### 4.1.8 Paging File Limit (PGFLQUOTA)

The paging file limit (PGFLQUOTA) limits the number of pages that the user's process can use in the system paging file. Effectively, PGFLQUOTA limits the total virtual address space that can be created using the Create Virtual Address Space ($CRETVA) or Expand Program/Control Region ($EXPREG) system services.

The paging file provides temporary disk storage for pages forced out of memory by a memory management operation.

PGFLQUOTA is a pooled limit with a suggested typical value of 2048.

### 4.1.9 Subprocess Creation Limit (PRCLM)

The subprocess creation limit (PRCLM) limits the number of subprocesses a user's process can create.

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 pooled limit with a suggested typical value of 2.

### 4.1.10 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.

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

TQELM is a pooled limit with a suggested typical value of 6.

### 4.1.11 Working Set Default (WSDEFAULT)

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

WSDEFAULT is a nondeductible limit with a typical value of 150 pages. If the value specified exceeds the value of WSQUOTA (see Section 4.1.13), the lesser value is used.

### 4.1.12 Working Set Extent (WSEXTENT)

The working set extent limit (WSEXTENT) specifies the maximum size to which a user's physical memory usage can grow, independent of the system load. This enlargement of the physical memory for a user is accomplished by the Adjust Working Set Limit ($ADJWSL) system service, and is typically done for the user by VAX/VMS in response to heavy page faulting by the user.

WSEXTENT is a nondeductible quota with a typical value of 350 (pages) or more. This value should always be greater than or equal to WSQUOTA. This value is minimized with the system parameter WSMAX.

### 4.1.13 Working Set Quota (WSQUOTA)

The working set quota limit (WSQUOTA) specifies the maximum size to which a user's physical memory usage can grow on a typically loaded system. That is, this parameter guarantees the user that the number of physical pages specified will be available for the user to do with as desired. For example, WSQUOTA limits the number of pages a user can lock in memory.

WSQUOTA is a nondeductible quota with typical values of 200-350 pages. This value should be greater than or equal to WSDEFAULT. This value is minimized with the system parameter WSMAX.

## 4.2 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 timesharing processes is 1 through 15; the range for real-time processes is 16 through 31.

Processes with real-time priorities are scheduled strictly according to base priority; in other words, the executable real-time process with the highest base priority is executed first. Processes with timesharing 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. The priority for timesharing users should remain at the default. Note that attempting to give some users an advantage over other users by varying priorities usually results in ragged performance, as the system reacts sharply to even small priority differences.

You should never specify a value over 31 (system operation will be unpredictable).

## 4.3 PRIVILEGES

Privileges restrict the performance of certain system activities to certain users. These restrictions protect the integrity of the operating system's performance and thus the integrity of service provided to users. You should grant privileges to each user on the basis of two factors: (1) whether the user has the skill and experience to use the privilege without disrupting the system and (2) whether the user has a legitimate need for the privilege.

Privileges fall into seven categories according to the damage that the user possessing them could cause the system:

- None - No privileges
- Normal - Minimum privileges to effectively use the system
- Group - Potential to interfere with members of the same group
- Devour - Potential to consume noncritical system-wide resources
- System - Potential to interfere with normal system operation
- File - Potential to compromise file security
- All - Potential to control the system

A user cannot execute an image that requires a privilege the user does not possess unless the image is installed as a known image with the privilege in question. (See Chapter 6 for instructions on installing known images.) Execution of a known image with privileges temporarily (for the duration of the image's execution) grants those privileges to the user process executing the image. Thus, you should install user images with amplified privileges only after ensuring that the user needs the access and is unlikely to misuse it.

A user's privileges are recorded in the user's UAF record 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. In this way, the user's privileges are passed on to the process created for the user. Users can use the DCL command SET PROCESS/PRIVILEGES to enable and disable privileges for which they are authorized, to further control the privileges available to the images they run. Moreover, any user with the SETPRV privilege can enable any privilege.

Table 4-2 lists the privileges by category and gives brief, general definitions of them. The sections that follow describe each privilege in detail in alphabetical order.

**Table 4-2: VAX/VMS Privileges**

| Category | Privilege | Activity Permitted |
|---|---|---|
| None | None | None requiring privileges |
| Normal | MOUNT | Execute mount volume QIO |
| | NETMBX | Create network connections |

(continued on next page)

Table 4-2 (Cont.): VAX/VMS Privileges

| Category | Privilege | Activity Permitted |
|---|---|---|
| Normal (Cont.) | TMPMBX | Create temporary mailbox |
| Group | GROUP | Control processes in the same group |
| Devour | ACNT | Disable accounting |
| | ALLSPOOL | Allocate spooled devices |
| | BUGCHK | Make bugcheck error log entries |
| | EXQUOTA | Exceed disk quotas |
| | GRPNAM | Insert group logical names in the name table |
| | PRMCEB | Create/delete permanent common event flag clusters |
| | PRMGBL | Create permanent global sections |
| | PRMMBX | Create permanent mailboxes |
| | SHMEM | Create/delete structures in shared memory |
| System | ALTPRI | Set base priority higher than allotment |
| | OPER | Perform operator functions |
| | PSWAPM | Change process swap mode |
| | WORLD | Control any process |
| | SYSLCK | Lock system-wide resources |
| Files | DIAGNOSE | Diagnose devices |
| | SYSGBL | Create system-wide global sections |
| | VOLPRO | Override volume protection |
| All | BYPASS | Disregard UIC protection |
| | CMEXEC | Change to executive mode |
| | CMKRNL | Change to kernel mode |
| | DETACH | Create detached processes |
| | LOG_IO | Issue logical I/O requests |

(continued on next page)

Table 4-2 (Cont.): VAX/VMS Privileges

| Category | Privilege | Activity Permitted |
|---|---|---|
| All (Cont.) | PFNMAP | Map to specific physical pages |
| | PHY_IO | Issue physical I/O requests |
| | SETPRV | Enable any privilege |
| | SYSNAM | Insert system logical names in the name table |
| | SYSPRV | Attain system user status |

### 4.3.1 ACNT Privilege

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 DCL command RUN with the /NOACCOUNTING qualifier or inhibit accounting in the Create Process ($CREPRC) system service.

### 4.3.2 ALLSPOOL Privilege

The ALLSPOOL privilege allows the user's process to allocate a spooled device by executing the Allocate Device ($ALLOC) system service, or the user is allowed to allocate a spooled device by using the DCL command ALLOCATE.

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

You should grant this privilege only to users who need to perform logical or physical I/O operations to a spooled device. Ordinarily, the privilege of allocating a spooled device is granted only to symbionts.

### 4.3.3 ALTPRI Privilege

The ALTPRI privilege allows the user's process 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 ($SETPRI) system service or the DCL command SET PROCESS/PRIORITY. As a rule, this system service lets a process set its own base priority or the base priority of another process. However, one process can only set the priority of a second process if (1) the second process is a subprocess of the first or (2) the first process has process control privilege (GROUP or WORLD) over the second. With the same privilege a process can create a process with a priority higher than its own. Such a process is created by using an optional argument to the Create Process ($CREPRC) system service or to the DCL command RUN.

You should not grant this privilege widely; if unqualified users have the unrestricted ability to set base priorities, the fair and orderly scheduling of processes for execution can easily be disrupted.

### 4.3.4 BUGCHK Privilege

The use of this privilege should be restricted to system software supplied by DIGITAL that uses the VAX/VMS Bugcheck Facility. This privilege allows the process to make bugcheck error log entries.

### 4.3.5 BYPASS Privilege

The BYPASS privilege allows the user's process read, write, execute, and delete access to all files, bypassing UIC protection.

You should grant this privilege with extreme caution, as it overrides all file protection. It should be reserved for use by either well-tested, reliable programs and command procedures or the system back-up operation (see Chapter 5 for a discussion of back-up operations). SYSPRV (see below) is adequate for interactive use, as it ultimately grants access to all files, while still providing access checks.

### 4.3.6 CMEXEC Privilege

The CMEXEC privilege allows the user's process to execute the Change Mode to Executive ($CMEXEC) 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 that was in effect before the system service was called. While in executive mode, the process is allowed to execute the Change Mode to Kernel ($CMKRNL) system service.

You 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 unqualified users have unrestricted access to sensitive data structures and functions, the operating system and service to other users can easily be disrupted. Such disruptions can include failure of the system, destruction of the data base, and exposure of confidential information to unauthorized persons.

### 4.3.7 CMKRNL Privilege

The CMKRNL privilege allows the user's process to execute the Change Mode to Kernel ($CMKRNL) 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 that was in effect before the system service was called.

You 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 unqualified users have unrestricted use of privileged instructions and unrestricted access to sensitive data structures and functions, the operating system and service to other users can easily

be disrupted. Such disruptions can include failure of the system, destruction of the data base, and exposure of confidential information to unauthorized persons.

### 4.3.8 DETACH Privilege

The DETACH privilege allows the user's process to create detached processes by executing the Create Process ($CREPRC) 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 are no restrictions on the files and directories to which a detached process can gain access.

### 4.3.9 DIAGNOSE Privilege

The DIAGNOSE privilege allows the user to run online diagnostic programs and to intercept and copy all messages that are written to the error log file.

### 4.3.10 EXQUOTA Privilege

The EXQUOTA privilege allows the space taken by the user's files on given disk volumes to exceed any usage quotas set for the user (as determined by UIC) on those volumes.

### 4.3.11 GROUP Privilege

The GROUP privilege allows the user's process to affect other processes in its own group by executing the following process control system services: Suspend Process ($SUSPND), Resume Process ($RESUME), Delete Process ($DELPRC), Set Priority ($SETPRI), Wake ($WAKE), Schedule Wakeup ($SCHDWK), Cancel Wakeup ($CANWAK), and Force Exit ($FORCEX). The user's process is also allowed to examine other processes in its own group by executing the Get Job/Process Information ($GETJPI) system service. The user with the GROUP privilege can issue the following DCL commands for other processes in its group: SET QUEUE, DELETE/ENTRY, STOP/ENTRY, and SET PROCESS.

The GROUP privilege is not needed for a process to exercise control over, or to examine, subprocesses that it created. You should, however, grant this privilege to users who need to exercise control over each other's processes and operations.

### 4.3.12 GRPNAM Privilege

The GRPNAM privilege allows the user's process to insert names into the logical name table of the group to which the process belongs and to delete names from that table by the use of the following logical name system services: Create Logical Name ($CRELOG) and Delete Logical Name ($DELLOG).

In addition, the privileged user can use the DCL commands ASSIGN and DEFINE to add names to the group logical name table, the DEASSIGN command to delete names from the table, and the /GROUP qualifier of the MOUNT command to share volumes among group members.

This privilege should not be granted to all users of the system because it allows the user to create an unlimited number of group logical names. When unqualified users have the unrestricted ability to create group logical names, excessive use of system dynamic memory can degrade system performance. In addition, a user with the GRPNAM privilege can interfere with the activities of other users in the same group by creating definitions of commonly used logical names such as SYS$SYSTEM.

### 4.3.13 LOG_IO Privilege

The LOG_IO privilege allows the user's process to execute the Queue I/O Request ($QIO) system service to perform logical-level I/O operations.

Usually, user 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 logical-level I/O operation.

You should grant this privilege only to users who need it because it allows a process to access data anywhere on the selected volume without the benefit of any file structuring. If this privilege is given to unqualified users who have no need for it, the operating system and service to other users can easily be disrupted. Such disruptions can include the destruction of information on the system device, the destruction of user data, and the exposure of confidential information to unauthorized persons.

### 4.3.14 MOUNT Privilege

The MOUNT privilege allows the user's process to execute the mount volume QIO function. The use of this function should be restricted to system software supplied by DIGITAL.

### 4.3.15 NETMBX Privilege

The NETMBX privilege allows the user to perform functions related to a DECnet computer network.

### 4.3.16 OPER Privilege

The OPER privilege allows the user 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.

You 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. (A nonprivileged user can log in on the console terminal to respond to operator requests, for example, to mount a tape.)

### 4.3.17 PFNMAP Privilege

The PFNMAP privilege allows the user's process to map to specific pages of physical memory or I/O device registers, no matter who is using the pages or registers.

You should exercise caution in granting this privilege. If unqualified users have unrestricted access to physical memory, the operating system and service to other users can easily be disrupted. Such disruptions can include failure of the system, destruction of the data base, and exposure of confidential information to unauthorized persons.

### 4.3.18 PHY_IO Privilege

The PHY_IO privilege allows the user's process to execute the Queue I/O Request ($QIO) system service to perform physical-level I/O operations.

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 their applications, 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.

You 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 4.3.13). If this privilege is given to unqualified users who have no need for it, the operating system and service to other users can easily be disrupted. Such disruptions can include the destruction of information on the system device, the destruction of user data, and the exposure of confidential information to unauthorized persons.

### 4.3.19 PRMCEB Privilege

The PRMCEB privilege allows the user's process to create or delete a permanent common event flag cluster by executing the Associate Common Event Flag Cluster ($ASCEFC) or Delete Common Event Flag Cluster ($DLCEFC) 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, because it allows the user to create an unlimited number of permanent common event flag clusters. A permanent cluster remains in the system even after the creating process has been terminated and continues to use up a portion of system dynamic memory. When many users have the unrestricted ability to create permanent common event flag clusters, the excessive use of system dynamic memory can degrade system performance.

### 4.3.20 PRMGBL Privilege

The PRMGBL privilege allows the user's process to create global sections by executing the Create and Map Section ($CRMPSC) system service. In addition, the user with this privilege (plus the CMKRNL and SYSGBL privileges) can use the Install Utility.

Global sections are shared structures that can be mapped simultaneously in the virtual address space of many processes. All processes see the same code or data. Global sections are used for reentrant subroutines or data buffers.

You should grant this privilege with care. If permanent global sections are not explicitly deleted, they tie up space in the global section and global page tables, which are limited resources.

### 4.3.21 PRMMBX Privilege

The PRMMBX privilege allows the user's process to create or delete a permanent mailbox by executing the Create Mailbox and Assign Channel ($CREMBX) system service or the Delete Mailbox ($DELMBX) 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.

The PRMMBX privilege should not be granted to all users of the system. Permanent mailboxes are not automatically deleted when the creating processes are deleted and, thus, continue to use up a portion of system dynamic memory.

### 4.3.22 PSWAPM Privilege

The PSWAPM privilege allows the user's process to control whether it can be swapped out of the balance set by executing the Set Process Swap Mode ($SETSWM) system service. Not only must a process have this privilege to lock itself in the balance set (that is, to disable swapping), but also to unlock itself (that is, to enable swapping).

With this privilege, a process can create a process that is locked in the balance set (process swap mode disabled) by using an optional argument to the Create Process ($CREPRC) system service or, when the DCL command RUN is used to create a process, by using a qualifier of the RUN command.

You should grant this privilege only to users who need to lock a process in memory for performance reasons. Typically, this will be a real-time process. If unqualified users have the unrestricted ability to lock processes in the balance set, physical memory can be held unnecessarily and thereby degrade system performance.

### 4.3.23 SETPRV Privilege

The SETPRV privilege allows the user's process to create processes whose privileges are greater than its own, by executing the Create Process ($CREPRC) system service with an optional argument, or by issuing the DCL command RUN to create a process. A user with this privilege can also execute the DCL command SET PROCESS/PRIVILEGES to obtain any desired privilege.

You should exercise the same caution in granting the SETPRV privilege as in granting any other privilege, since SETPRV allows the user to enable any or all privileges.

### 4.3.24 SHMEM Privilege

The SHMEM privilege allows the user's process to create global sections and mailboxes (permanent and temporary) in multiport memory, if the process also has appropriate PRMGBL, PRMMBX, SYSGBL, and TMPMBX privileges. Just as in local memory, the space required for a multiport memory temporary mailbox counts against the buffered I/O byte count limit (BYTLM) of the process.

### 4.3.25 SYSGBL Privilege

The SYSGBL privilege allows the user's process to create system global sections by executing the Create and Map Section ($CRMPSC) system service. In addition, the user with this privilege (plus the CMKRNL and PRMGBL privileges) can use the Install Utility.

You should exercise caution in granting this privilege. System global sections require space in the global section and global page tables, which are limited resources.

### 4.3.26 SYSLCK Privilege

The SYSLCK privilege allows the user's process to lock system-wide resources with the Enqueue Lock Request ($ENQ) system service. You should grant this privilege to users who need to run programs that lock resources in the system-wide resource name space.

### 4.3.27 SYSNAM Privilege

The SYSNAM privilege allows the user's process to insert names into the system logical name table and to delete names from that table by using the Create Logical Name ($CRELOG)and Delete Logical Name ($DELLOG) system services.

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

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

Note that a user with SYSNAM privilege could redefine such critical system logical names as SYS$SYSTEM and SYSUAF, thus gaining control of the system.

### 4.3.28 SYSPRV Privilege

The SYSPRV privilege allows the user to assume the file access rights of a system user, and to change the owner UIC and protection of a file. Even if a file is protected against system access, the user with the SYSPRV privilege can simply change the file's protection to gain access to it.

You should exercise caution in granting this privilege. If unqualified users have system access rights, the operating system and service to others can easily be disrupted. Such disruptions can include failure of the system, destruction of the data base, and exposure of confidential information to unauthorized persons.

### 4.3.29 TMPMBX Privilege

The TMPMBX privilege allows the user's process to create a temporary mailbox by executing the Create Mailbox and Assign Channel ($CREMBX) 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 it is no longer referenced by any process. Note that this privilege is required to use the DCL commands SUBMIT and PRINT.

You 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 (BYTLM quota).

### 4.3.30 VOLPRO Privilege

The VOLPRO privilege allows the user to (1) initialize a previously used volume with an owner UIC different from the user's own UIC; (2) override the expiration date on a non-owned tape or disk volume; (3) mount a non-owned Files-11 volume with the /FOREIGN qualifier; and (4) override the owner UIC protection of a volume. The VOLPRO privilege only permits control over volumes the user can mount or initialize. Volumes mounted with the /SYSTEM qualifier are safe from the user with the VOLPRO privilege as long as the user does not also have the SYSNAM privilege.

You should exercise extreme caution in granting the VOLPRO privilege. If unqualified users can override volume protection, the operating system and service to others can be disrupted. Such disruptions can include destruction of the data base and exposure of confidential information to unauthorized persons.

### 4.3.31 WORLD Privilege

The WORLD privilege allows the user's process to affect other processes both inside and outside its group by executing the following process control system services: Suspend Process ($SUSPND), Resume Process ($RESUME), Delete Process ($DELPRC), Set Priority ($SETPRI), Wake ($WAKE), Schedule Wakeup ($SCHDWK), Cancel Wakeup ($CANWAK), and Force Exit ($FORCEX). The user's process is also allowed to examine processes outside its own group by executing the Get Job/Process Information ($GETJPI) system service. The user with the WORLD privilege can issue the DCL commands SET QUEUE, DELETE/ENTRY, STOP/ENTRY, and SET PROCESS for all other processes.

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 inside 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.

## 4.4 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 log file SYS$MANAGER:ACCOUNTNG.DAT, which is updated each time the system is initialized, each time an accountable process or image terminates, each time a print job is completed, and each time a login failure occurs. In addition, users can send messages to be inserted into the accounting log file.

Accounting records contain cumulative accounts of the resources used either by processes or images set up for users or by print symbionts that print out files for users. Each accounting record contains three fields -- user name, UIC, 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).

You can use the Accounting Utility to sort, select, and report the accounting records. The reports can provide valuable system management tools. (See the VAX-11 Utilities Reference Manual.) Alternatively, by using the detailed accounting records provided by the system, you or perhaps a system programmer can devise programs for reporting on the use of system resources and for billing for their use.

The accounting log file is created and opened automatically when the operating system is initialized. Accounting records are arranged chronologically in this file. The following list summarizes the characteristics of the accounting log file:

- File name: ACCOUNTNG.DAT (this file is not an ASCII file; hence, it must be formatted before it is printed)
- Residence and Directory: SYS$MANAGER
- File organization: sequential
- Record length: variable length
- Record types: eight

The eight types of records correspond to the conditions that cause records to be written to the file. These record types are shown in the list that follows. Note that their corresponding codes as defined in the macro $ACRDEF in SYS$LIBRARY:STARLET.MLB are shown in parentheses in this list:

1. Records written when processes are deleted (ACR$K_PRCDEL)
2. Records written when an image was exited (ACR$K_IMGDEL)
3. Records written when the system was initialized (ACR$K_SYSINIT)
4. Records written when print jobs are queued (ACR$K_PRINT)
5. Records written when login failures occurred (ACR$K_LOGFAIL)
6. Records written when users' messages are sent to the accounting log file (ACR$K_USER)
7. Record that points to the next accounting file (ACR$K_FILE_FL)
8. Record that point to the previous accounting file, if any (ACR$K_FILE_BL)

For more information about the records of the accounting log file, see Appendix C of the VAX-11 Utilities Reference Manual.

Privilege is required to suppress the accounting function and thus avoid accounting for the use of system resources. Only a user who has the ACNT privilege can create subprocesses or detached processes in which accounting is disabled. The /NOACCOUNTING qualifier of the DCL command RUN disables all accounting in a created process.

A user with OPER privilege can selectively disable various kinds of accounting throughout the system by using the /DISABLE qualifier of the DCL command SET ACCOUNTING. Usually, this is considered a system management task. See the VAX/VMS Command Language User's Guide for a full description of the SET ACCOUNTING command.

As records are entered in the accounting log file, all but image termination 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 log file is closed at the end of a billing period. The current version of the accounting log file is closed and a new version of the file is created and opened. As a rule, you perform this job with the SET ACCOUNTING command.

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

# CHAPTER 5

# MAINTAINING PUBLIC FILES AND VOLUMES

Public volumes, also called system volumes, are file-structured disk volumes that contain public files. Public files are files that must be available to most, if not all, users. Public volumes can 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 volumes and to the files on them.

Public volumes can 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 such as macro libraries, object module libraries, shared run-time libraries, and error message libraries
- Text files such as help files
- Optional software in executable form, plus related libraries and other files

In addition, you 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, you can permit any user to create and store files on a public volume. Depending on their file protection, these files can be of general or restricted accessibility. 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, if the user has write access to a directory on the volume, and if disk quotas permit. As a rule, you create a default disk file directory on a public volume for each user authorized to use the system (see Chapter 2).

Before you can manage a system of public files and volumes, you must know how to initialize and mount public volumes (see Sections 5.4 and 5.5).

## 5.1 FILES-11 DISK STRUCTURE

The VAX/VMS system recognizes two disk file structures: 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, RSX-11M-PLUS, and IAS operating systems.

Nine files control the structure of a Files-11 Structure Level 2 volume. Only five of these files are used for a Files-11 Structure Level 1 volume. All nine files, which are referred to as reserved files, are identified in Table 5-1, with an indication of which Files-11 Structure Level they pertain to.

**Table 5-1: VAX/VMS Reserved Files**

| Reserved File | File Name | Structure Level | |
|---|---|---|---|
| | | 1 | 2 |
| Index file | INDEXF.SYS;1 | X | X |
| Storage bit map file | BITMAP.SYS;1 | X | X |
| Bad block file | BADBLK.SYS;1 | X | X |
| Master file directory | 000000.DIR;1 | X | X |
| Core image file | CORIMG.SYS;1 | X | X |
| Volume set list file | VOLSET.SYS;1 | | X |
| Continuation file | CONTIN.SYS;1 | | X |
| Back-up log file | BACKUP.SYS;1 | | X |
| Pending bad block | BADLOG.SYS;1 | | X |

All the files listed above are cataloged in the master file directory (MFD), [000000].

### 5.1.1 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 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 system volume, this block contains a bootstrap program that loads the operating system into memory. If the volume is not a system volume, this block contains a program that displays a message that the volume is not the system device but a device that contains users' files only.

- 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 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.

- 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, file protection, and location of the file's blocks. The file header contains all the information needed for gaining access to the file.

### 5.1.2 Storage Bit Map File

The storage bit map file controls the available space on a volume; this file is listed in the master file directory 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.

### 5.1.3 Bad Block File

The bad block file, which is listed in the master file directory as BADBLK.SYS;1, contains 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.

### 5.1.4 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.

When the Backup Utility creates sequential disk save sets, it stores the save set file in the MFD. (See Section 5.8.12 for a discussion of sequential disk save sets.)

### 5.1.5 Core Image File

The core image file is listed in the MFD as CORIMG.SYS;1. It is not used by VAX/VMS.

### 5.1.6 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 volume set. The file contains a list of the labels of all the volumes in the set.

### 5.1.7 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 to another volume of a loosely coupled volume set. This file is used for all but the first volume of a sequential disk save set (see Section 5.8.12 for a discussion of sequential disk save sets).

### 5.1.8 Back-up Log File

The back-up log file is listed in the MFD as BACKUP.SYS;1. This file is reserved for future use.

### 5.1.9 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.

### 5.1.10 Files-11 Structure Level 1 Versus Structure Level 2

Files-11 Structure Level 2, a compatible superset of Structure Level 1, is the preferred disk structure on VAX/VMS for reasons of performance and reliability. At volume initialization time (see the INITIALIZE command in the VAX/VMS Command Language User's Guide), Structure Level 2 is the default. Structure Level 1 should be specified only for volumes that must be transportable to RSX-11M, RSX-11D, RSX-11M-PLUS, and IAS systems, as these systems support only that structure level. Additionally, you may be required to handle Structure Level 1 volumes transported to VAX/VMS from one of the above systems.

Where Structure Level 1 volumes are in use on the system, you should bear in mind the following limitations on them:

- Directories -- No hierarchies of directories and subdirectories, and no ordering of directory entries (that is, the file names) in any way (RSX-11M, RSX-11D, RSX-11M-PLUS, and IAS systems do not support subdirectories and alphabetical directory entries)

- Wild cards -- Wild card characters only for complete fields of file specifications (for example, *USER.TXT is illegal, while *.TXT is legal)

- Disk quotas -- Not supported
- Multivolume files and volume sets -- Not supported
- Placement control -- Not supported
- Caches -- No caching of file header blocks, file identification slots, or extent entries
- System disk -- Cannot be a Structure Level 1 volume
- Clustered allocation -- Not supported
- Back-up home block -- Not supported
- Protection code E -- Means extend for RSX-11M, but is ignored by VAX/VMS
- File versions -- Limited to 32767; version limits are not supported

Future enhancements to VAX/VMS will be based primarily on Structure Level 2, so that further restrictions on Structure Level 1 volumes may be incurred.

As explained in the VAX-11 Utilities Reference Manual, if you choose the DSC utilities for back-ups, you must use DSC1 to back up Structure Level 1 volumes and DSC2 to back up Structure Level 2 volumes. The preferred utilities are the Backup Utility and the Verify Utility, which support both Structure Level 1 and Structure Level 2.

## 5.2 SETTING UP PUBLIC FILE STRUCTURES

A major duty in system management is maintaining public file structures. You must strike a balance between your users' needs and the system's available mass storage resources. You must determine how mass storage devices on your system will be configured, which devices will hold public system volumes, which devices will be available for users' private volumes, and how the public volumes will be configured. You are also responsible for creating top level user file directories as needed, and monitoring users' disk space usage.

### 5.2.1 Deciding Where to Put User Files

In most circumstances, you will want to dedicate most of your large disk drives to public disk storage. In relatively large system configurations, you should not put user files on the system volume. The system disk will be kept active with paging and swapping, spooling files, maintaining system logs, and so forth. Furthermore, if you ever find it necessary to build a new system disk from scratch, you will find that having user files on it makes the process much more difficult and time-consuming.

If you have a relatively small mass storage configuration, you will have no other choice than to allocate user files on the system volume. Under these circumstances, you should take adequate precautions with disk quotas and user education to ensure that users' files do not exhaust the free space on the system disk.

### 5.2.2 Should You Use Volume Sets?

A volume set is a collection of disk volumes that have been bound into a single entity, by the MOUNT/BIND command. A volume set appears to be a single, large volume. Files are automatically allocated anywhere on the volume set that space is available, disk quotas are enforced over the set as a whole, and a single directory structure covers the whole volume set. If you want to provide a large homogeneous public file space, use a volume set. You must also use a volume set if you intend to create data base files that are larger than any single disk volume. It is worth noting that the file system attempts to balance the load on the volume, with tactics such as creating new files on the volume that is the least full at the time.

On the other hand, if you want several distinct areas of file storage, with different user bases or different management policies, you must use a separate volume (or volume set) for each area. Each separate volume or volume set must contain a top-level user file directory for each user who will keep files on that volume. For example, you might want one volume for permanent user storage, with carefully limited quotas and careful back-ups, and another volume for "scratch" use, which has liberal or no quotas, is not backed up, and whose files are cleaned out on a periodic basis. Another advantage of using separate volumes is their modularity. Should one of the drives holding a volume set be out of service, the whole volume set will be unavailable because of the interconnected nature of its directory structure. On the other hand, when a drive holding a single volume is unavailable, only a well-defined set of files becomes unavailable.

For planning purposes, remember:

- Any single volume can be turned into a volume set by binding it with a newly initialized volume. Likewise, you can always add another newly initialized volume to an existing volume set.
- You can bind disk volumes of different types into the same volume set.
- You cannot bind two existing separate volumes containing files into a volume set. (The MOUNT command will let you do this, but the result will not be a coherent volume set.)
- You only need to issue the MOUNT/BIND command once to effect binding; thereafter, the volume-set association is recorded on the volumes.
- Once you have bound two or more volumes into a volume set, they cannot be separated. The only way to separate a volume set is to selectively copy sets of directories using BACKUP.

For more information on volume sets, see the VAX/VMS Command Language User's Guide.

### 5.2.3 Should You Make the System Disk Part of a Volume Set?

DIGITAL does not recommend that you make the system disk part of a volume set. While VAX/VMS will continue to boot and run successfully if you do this, optional product installations, maintenance updates, and system upgrades will not install correctly on a system disk that is part of a volume set. Once you have installed an update or software product, system files can be allocated anywhere on the volume set, and VAX/VMS will no longer boot successfully.

## 5.3 FORMATTING DISKS

Disks that you purchase from DIGITAL are preformatted with the EVRAC disk formatter. However, under some circumstances you may need or want to format a disk. Disks must be reformatted if they have been exposed to X rays, degaussing, or certain kinds of power disruptions. Also, you may find formatting desirable if you are experiencing excessive parity errors on a disk. In such cases, you should contact your Field Service representative for assistance.

## 5.4 INITIALIZING PUBLIC VOLUMES

The purpose of initializing a disk volume is to delete all old information from the volume and to impart to the volume a Files-11 structure that the operating system recognizes. This structure prepares a volume to receive data and permits the operating system to locate data stored on the volume.

In initializing a public volume (by using the qualifier /SYSTEM), you may need to use one or more 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.

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

### 5.4.1 /ACCESSED Qualifier

The /ACCESSED 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 directory file control blocks 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. For volumes mounted with the /SYSTEM qualifier, the system parameter ACP_SYSACC overrides this value.

When you create a volume set, specify reasonable values for the /ACCESSED qualifier on each volume because the total number of directory file control blocks retained will be the sum of the values of all the /ACCESSED qualifiers specified for the volume set.

### 5.4.2 /CLUSTER_SIZE Qualifier

The /CLUSTER_SIZE 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).

### 5.4.3 /EXTENSION Qualifier

The /EXTENSION 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, RSX-11M-PLUS, and IAS systems, because VAX-11 Record Management Services use an adaptive algorithm maximized against /EXTENSION. The value of this qualifier should be an even multiple of /CLUSTER_SIZE.

### 5.4.4 /HEADERS Qualifier

The /HEADERS 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 (usually 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 allocated to headers cannot later be made available for file storage.

### 5.4.5 /INDEX Qualifier

The /INDEX 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 only one or a few very large contiguous files.

When the Backup Utility copies a volume as the result of a BACKUP/IMAGE command, it preserves the placement of the index file, if the output device is the same type. Otherwise, it defaults to MIDDLE. (The Disk Save and Compress (DSC) Utilities position the index at BEGINNING.)

### 5.4.6 /MAXIMUM_FILES Qualifier

The /MAXIMUM_FILES qualifier specifies the maximum number of files that a volume can contain. The default value is fairly liberal. A closer estimate of it helps optimize the dynamic allocation of the index file; once set, however, 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.

### 5.4.7 /WINDOW Qualifier

The /WINDOW qualifier specifies the default number of map pointers in a 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.

## 5.5 MOUNTING PUBLIC VOLUMES

The purpose of mounting a volume or volume set is to establish a relationship between the volume or volume set, the device(s) on which the volume is physically mounted, and one or more processes that can gain access to the volume.

In mounting a public volume (by using the qualifier /SYSTEM), you may need to use one or more of the qualifiers /ACCESSED, /EXTENSION, and /WINDOW (described in Section 5.4), or the following additional qualifiers:

- /ASSIST
- /COMMENT=text
- /MOUNT_VERIFICATION
- /PROCESSOR=option

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

### 5.5.1 /ASSIST Qualifier

The /ASSIST qualfier enables or disables the operator-assisted mount feature. Section 5.12 describes operator-assisted mounts. By default, this feature is enabled. You should encourage your users to take advantage of this feature.

There is a situation where operator-assisted mounts generally prove undesirable. During the execution of SYSTARTUP.COM, operator-assisted mounts are disabled by default. The reason for this is that the absence of some system volume normally mounted during SYSTARTUP.COM would prevent the system from booting. If you have one or more volumes that must be present for the correct operation of your system, you can specify the /ASSIST qualifier in the commands to mount them, if you are prepared for the following consequences if any volume proves to be offline or unavailable. The operator assistance software will issue an operator request to have the volume made ready, and will wait for its completion. There is no problem if you can ready the volume. However, if you cannot ready the volume (perhaps the drive is down or the volume is corrupted), you cannot complete SYSTARTUP.COM. To abort the mount request, you would have to issue a REPLY/ABORT operator command. However, the system console is running SYSTARTUP.COM and is unavailable. Furthermore, the other system terminals are usually not yet properly configured or logins are not yet enabled. Under these circumstances, the only way to boot the system is to invoke the command procedure SYS$SYSTEM:STARTUP.MIN.

### 5.5.2 /COMMENT Qualifier

The /COMMENT qualifier includes the quoted text string that you specify as part of the mount request, thus passing it on to the operator if the mount requires operator assistance. This qualifier is primarily useful in situations where operator assistance is expected, such as to inform the operator of the physical location of a particular volume that is required. You should encourage your users to take advantage of this feature.

### 5.5.3 /MOUNT_VERIFICATION Qualifier

The /MOUNT_VERIFICATION qualifier enables or disables the mount verification feature on Files-11 disks. This feature is explained in Section 5.7. By default, the mount verification feature is enabled. However, note that this feature has no effect for foreign disks or magnetic tapes.

### 5.5.4 /PROCESSOR Qualifier

The /PROCESSOR qualifier specifies the number of ancillary control processes (ACPs) to be used in controlling various public volumes. Selecting an appropriate option for the /PROCESSOR qualifier involves making a trade-off. If you specify the option SAME, file system parallelism and performance may be sacrificed for the sake of saving system space. Conversely, if you specify the option UNIQUE, system space is sacrificed for the sake of file system parallelism and performance. Unless you have substantial memory to waste, you should use the system default. (See Chapter 10 for a description of the ACP_MULTIPLE system parameter.)

You can monitor Files-11 ACP performance with the MONITOR FCP command of the Monitor Utility. (See the VAX-11 Utilities Reference Manual.) The information MONITOR provides can help you determine how to configure your file system.

## 5.6 MAINTAINING VOLUME INTEGRITY

To enhance performance, the system caches in memory information concerning a volume's free space, file identifications, quota file entries, and file headers. You determine the degree of caching with the ACP cache system parameters (see Chapter 10). Individual users can alter cache sizes on their volumes with qualifiers to the DCL command MOUNT (see the VAX/VMS Command Language User's Guide). The system writes the information in the caches to the disk when the disk is dismounted or the system is shut down. Naturally, removal of a disk before the caches are written back loses any changes made to the information in the caches. Therefore, you and the individual user should:

1. Not write-lock a volume while it is mounted
2. Not remove a volume from a drive until it has been dismounted
3. Not halt the system without performing the orderly shutdown procedure (see Chapter 7).

If anyone write-locks a volume at mount time, the system additionally applies a software write-lock. If you need to write-enable a volume that was mounted while the write-lock switch was on, you must first dismount the volume, then write-enable the drive, and then remount the volume. If a volume was mounted on a drive with write-lock off and then someone toggles the write-lock switch on (and if mount verification is enabled for the volume, which it is by default), the volume enters mount verification. All I/O operations to the volume are suspended. Section 5.7.2 describes how recovery is effected with write-lock mount verification. (Without the mount verification facility, you would have to dismount the volume, write-enable the drive, and then remount the volume.)

At mount time, if the system detects that the caches were not written back the last time the volume was used, the system automatically

rebuilds the file information by scanning the contents of the volume. However, file headers for files open at the time of the improper dismount may be partially or wholly lost.

## 5.7 MOUNT VERIFICATION

The mount verification feature of Files-11 disk handling generally leaves users unaware that a mounted disk has gone offline and returned online or in some other way has become unreachable and then restored. Mount verification is enabled by default with the /MOUNT_VERIFICATION qualifier when the disk is mounted. To disable mount verification, the user must specify /NOMOUNT_VERIFICATION when mounting the disk. Note that this feature does not apply to foreign disks or to magnetic tapes.

Mount verification sends messages to OPCOM. Because there are cases where mount verification messages are needed at the operator's console and OPCOM might not be able to provide them, mount verification also sends special messages with the prefix %SYSTEM-I-MOUNTVER to the operator's console only, that is, to OPA0. For example, if the system disk undergoes a mount verification or if the OPCOM process is not present on a system, the operator would at least receive the messages with the %SYSTEM-I-MOUNTVER prefix. Under normal circumstances, both messages are received at the operator's terminal, with the %SYSTEM-I-MOUNTVER message arriving first.

### 5.7.1 Device Offline Mount Verificaton

If a mounted disk volume goes offline while mount verification is enabled, you can follow the steps in the procedure below to resume operations.

**Procedure**

When a device is taken offline, the steps in mount verification are as follows:

1. Due to a hardware or user error, a disk volume is taken offline (for example, it might be spun down). Most disk drives generate a special interrupt when the volume comes back online, which causes the software to mark the volume as invalid. However, some disk drives (such as the RX01, RX02, RL02, and TU58) do not generate online attention interrupts. For these devices, an offline condition is only detected when an I/O operation is initiated for the drive.

2. An I/O operation fails with a medium offline or volume invalid status. The software marks the volume to indicate that it is undergoing mount verification, and all I/O operations to the disk are stalled.

3. OPCOM issues a message to the operators enabled for DISKS and DEVICES to announce the disk's unavailability, as follows:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, Device device-name is offline.
        Mount verification in progress.
```

4. You may simply choose to take the disk out of the offline and verification pending state by shutting down mount verification with one of the three techniques described in Section 5.7.3. These techniques cause the pending and future I/Os to the volume to fail.

5. Otherwise, you take corrective action. Perhaps the drive can be brought back online. If the disk drive is faulty, but another functioning drive is available on the same controller, you can move the disk to the functioning drive and swap the unit select plugs.

6. The disk comes back online, which is detected by the mount verification software that polls the disk drive.

7. The system verifies that the disk's home block matches the one in the data base of mounted volumes, thus confirming that this is the same disk as previously mounted.

8. If the drive does not contain the correct volume, OPCOM issues the following message:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, Device device-name contains the wrong volume.
        Mount verification in progress.
```

9. After the mount verification succeeds, the disk is marked as valid. OPCOM issues the following message:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, Mount verification completed for device device-name
```

   At this point I/O operations to the disk are allowed to proceed.

**Example**

```
%OPCOM, 15-JUN-1982 11:54:54.12, Device DMA0 is offline.
        Mount verification in progress.
%OPCOM, 15-JUN-1982 11:57:34.22, Mount verification completed for device DMA0
```

The message from OPCOM alerts the operator that device DMA0 has gone offline and mount verification has been initiated. The operator finds the drive was accidentally spun down and successfully spins it back up. The next message indicates that mount verification is satisfied that the same volume is on the drive (which it has found is online again), and all I/Os to the volume resume.

### 5.7.2 Device Write-Lock Mount Verification

If for some reason a mounted disk volume becomes write-locked while mount verification is enabled, you can follow the steps in the procedure below to resume operations.

**Procedure**

1. A Files-11 disk volume is mounted for writing (the /WRITE qualifier is the default). Through some hardware or user error, the disk becomes write-locked.

2. The software discovers that the disk is write-locked (typically an I/O fails with a write-lock error). The disk is marked that verification is in progress. OPCOM issues a message to the operators enabled for DISKS and DEVICES to announce the disk's unavailability, as follows:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, Device device-name has been write locked.
        Mount verification in progress.
```

3. You may simply choose to take the disk out of the verification pending state by shutting down mount verification with one of the techniques described in Section 5.7.3. These techniques cause the pending and future I/Os to the volume to fail.

4. Otherwise, you take corrective action. Perhaps the drive can simply be write-enabled by toggling the drive's hardware write-lock switch. If the disk drive is faulty, but another functioning drive is available on the same controller, you can move the disk to the functioning drive and swap the unit select plugs. (Note that switching to another drive will cause the volume to undergo offline mount verification (see Section 5.7.1). Once that completes, the write-lock mount verification continues.)

5. The mount verification software that polls the disk drive determines that the volume is in a writeable state. At this point, I/O operations to the disk are allowed to proceed. However, OPCOM does not issue a message indicating that the write-lock mount verification has completed.

**Example**

```
%OPCOM, 29-JUN-1982 15:28:54.07, Device DMA1 has been write locked.
        Mount verification in progress.
```

The OPCOM message alerts the operator that device DMA1 is write-locked. The operator toggles the write-lock switch on the drive to eliminate the write-lock. I/O operations to the disk resume, with no further messages.

### 5.7.3 Cancelling Mount Verification

If you fail to either bring a system disk back online or to write-enable it as appropriate while mount verification is enabled, the whole system can become hung, if the disk uses the same file ACP as the system disk. In that case, you cannot issue a DISMOUNT command to abort the request. This is because the file ACP that must handle the I/O operation is single-threaded. It cannot process any other request until the stalled I/O completes. Thus, as individual users make further requests for ACP services, they each become hung. Under these circumstances, you must cancel the pending mount verification, using one of the first two techniques described below.

You can cancel a mount verification request in one of three ways:

1. Allow the mount verification in progress to continue the number of seconds defined by the system parameter MVTIMEOUT. When the time expires, the system automatically cancels the pending mount verification. Note that a mount verification initiated by a write-lock will not time out.

2. Invoke a special cancelling routine from the console terminal.

3. Dismount the volume with the DCL command DISMOUNT from a process that is not hung.

**5.7.3.1 MVTIMEOUT System Parameter** - The MVTIMEOUT system parameter defines the time (in seconds) that is allowed for a pending mount verification to complete before it is automatically cancelled (see Chapter 10). This dynamic parameter should always be set to a reasonable value for the typical operations at your site. See Chapter 11 for instructions on how to display and modify dynamic system parameters with the System Generation Utility (SYSGEN). Note that resetting the value of the MVTIMEOUT parameter will not affect a mount verification that is currently in progress.

You will probably find that 10 minutes (600 seconds) is a good value for MVTIMEOUT, whether you normally operate with or without an operator.

When a pending mount verification is cancelled by timing out, OPCOM prints the following message:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, Mount verification aborted for device device-name
```

After a mount verification times out, all pending and future I/O requests to the volume will fail. Thus, you must dismount and remount the disk before you can access it again.

Note that a write-lock mount verification will not time out.

### 5.7.4 Cancellation Commands

If you must stop a mount verification before the time specified by MVTIMEOUT can elapse, enter the following sequence of commands from the console terminal of your VAX-11 processor:

```
CTRL/P
>>>HALT
>>>D/I 14 C
>>>CONT
IPC>
```

While you are at the IPC> prompt, all system operation is suspended. (Press CTRL/Z when you are ready to exit.)

Note the following special characteristics of the mount verification cancelling routine:

- Lowercase characters are converted to uppercase.
- Illegal characters (such as most control characters) are not echoed; instead, the terminal bell character is issued as a warning alarm.
- Leading spaces are ignored and are not echoed.
- Multiple spaces are compressed into a single space; that is, all space characters after the first are ignored.

There are two commands you can enter in response to the IPC> prompt:

C device

This command cancels any pending mount verification on the device specified. (A warning is given if no mount verification was in progress for the device specified.)

X

This command transfers control to the debugging tool XDELTA (provided it was loaded with the system by setting the appropriate value in the boot file). If XDELTA has not been loaded, the prompt IPC> is reissued. XDELTA, which is described in the VAX/VMS Guide to Writing a Device Driver, may prove especially useful if you are debugging privileged software on a VAX-11/782 attached processor system.

Press CTRL/Z to exit from the mount verification cancelling routine and resume system operation.

When a pending mount verification is cancelled, OPCOM prints the following message:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, Mount verification aborted for device device-name
```

After you successfully cancel a pending mount verification with this technique, you must dismount and then remount the volume before you can access it again.

**Example**

```
%OPCOM, 15-JUN-1982 10:54:54.12, Device DBA0 is offline.
        Mount verification in progress.

CTRL/P
>>>HALT
>>>D/I 14 C
>>>CONT
IPC>C DBA0
CTRL/Z
%SYSTEM-I-MOUNTVER, _DBA0: has aborted mount verification.
%OPCOM, 15-JUN-1982 10:56:26.13, Mount verification aborted for device DBA0
```

The operator observes that device DBA0 is offline but is unable to spin the disk back up. There is no other available drive on the controller, so it is not possible to switch the unit select plugs of the two drives. (The operator also rejects the possibility of issuing a DISMOUNT command for the disk (see Section 5.7.5), because it was mounted as a private volume.) Rather than wait ten minutes for the mount verification to time out, the operator decides to invoke the cancellation commands at the console terminal. Observe that the %SYSTEM-I-MOUNTVER message also appears here because this is the console terminal.

### 5.7.5 Dismounting the Volume to Abort Mount Verification

In some cases, you can abort mount verification by dismounting the volume in question. This only works if it is possible for you to issue the DCL command DISMOUNT for the volume when both of the following conditions are true:

1. The system file ACP is not hung
2. You are allowed access to the volume; that is, the volume was mounted with the /SYSTEM or /GROUP qualifiers

Follow the steps in the procedure below.

**Procedure**

1. Log in at another terminal or use any logged in terminal that has access to the volume.

2. Enter the DISMOUNT command for the volume

3. When you cancel a pending mount verification by dismounting the volume, OPCOM issues the following message:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, Mount verification aborted for device device-name
```

   If you do not have access to the volume, you did not satisfy the second condition above, and you will receive an error message. You can try again if you can find an appropriate process to use. If your process hangs, it is the system file ACP that is hung (first condition above was not met), and you cannot use this technique to cancel mount verification.

4. Once the cancellation succeeds, remove the volume from the drive.

## 5.8 BACKING UP PUBLIC VOLUMES

Backing up a volume means copying the contents of the volume to another volume or set of volumes (for example, another disk or tape). Backing up volumes is a precautionary measure to allow you to recover from the loss or destruction of valuable information.

Most sites establish a policy and a schedule for regularly backing up files on public volumes. Sections 5.8.3 through 5.8.12 provide the operating procedures for backing up both selected files and entire volumes.

It is just as desirable to back up information on private volumes as it is to back up public volumes. 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) incremental, or partial, back-ups, and (2) full, or all-inclusive, back-ups. The back-up medium, in either case, can be disk or tape. Incremental backups efficiently capture only those files that have been modified recently. Periodic full back-ups are necessary to provide the basis for reconstruction of a lost volume.

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

Generally, you are responsible for setting up a schedule for backing up files and volumes and for maintaining this schedule.

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

- Daily -- An incremental back-up retained for seven days. This schedule requires seven daily tapes that are rotated once a week.

- Weekly -- An incremental back-up retained for four weeks. This schedule requires four weekly sets of tapes that are rotated once every four weeks.

- Monthly -- An all-inclusive back-up retained for a year. This schedule requires twelve monthly sets of tapes that are rotated once a year.

Despite all precautions, there is always the risk of losing a file. Frequent back-ups and longer retention periods reduce this risk.

You can perform full (all-inclusive) back-ups to tape or copy the volume to another disk. Each has its advantages and disadvantages. The advantage of using tape for back-ups is the much lower media cost, which may in turn permit you to retain back-ups longer than keeping full back-ups on disk would.

There are several advantages to keeping copies on disk, which in some cases outweigh their higher cost. Disks exhibit better data reliability than tapes. Furthermore, disks tend to degrade less in storage. Also, if you have to replace a lost volume from its back-up medium, a disk back-up volume is ready for immediate use, whereas you must first restore a tape to disk.

Finally, if you use disks for back-up, you can make use of a "rotating back-up set," in which several disks or sets of disks are used in rotation on the system. At the end of each period of use (for example, once a month), the volume or volume set currently in use is copied to the oldest set of disks, the current volume is retired, and the new copy is put online for use during the next period.

### 5.8.1 Rotating Back-up Sets

Rotating back-up sets have their own advantages and disadvantages, as this section describes.

A rotating back-up set offers two major advantages:

1. Your back-up copy (the volume or volume set just retired) is known to be good, since it has been in use. The integrity of the new copy will be confirmed by its subsequent use; any defects discovered can be repaired using the back-up copy.

2. The free space on the new volume is compressed, and all the files on it are made contiguous or almost contiguous, resulting in better file system performance.

There are three disadvantages to a rotating backup set:

1. Rotating back-up sets are more vulnerable to disk errors than sets created by retiring the copy and continuing to run with the original. A disk error during the copy operation results in corrupted data on your new volume; disk errors in directories or file headers will result in the loss of one or more files. Thus, you must monitor the copy operation very carefully for errors and manually repair any problems that arise.

2. You cannot perform the copy operation while users are updating files. The volume or volume set must be write-locked so that the copy will be consistent. This restriction only applies to rotating back-up sets; it does not apply when you make a back-up copy that will be retired for back-up use. (Even though individual files may be incomplete, they will be covered by the next incremental back-up.)

3. Files created with explicit placement lose their placement when the volume is copied. This means you should not use a rotating back-up set if the volume's primary contents are a set of data base files that were carefully placed for optimized performance.

### 5.8.2 Backing Up Disk Volumes

The preferred method is to use the Backup Utility (BACKUP) for both incremental and full back-ups.

NOTE

DIGITAL does not intend to support the use of the Disk Save and Compress (DSC) utilities on VAX/VMS starting with the next major release. For this reason, users who are new to the system should use the Backup Utility and not the DSC utilities. Current users of DSC should begin planning their transition to BACKUP now.

For more information on the Backup Utility, refer to the VAX-11 Utilities Reference Manual.

The following sections discuss various ways you can back up volumes and files.

### 5.8.3 Backing Up the System Disk (Using Stand-alone BACKUP)

To back up the system disk, you need to use Stand-alone BACKUP. You can back up the system disk onto magnetic tape or another disk. You should follow the procedures given in the software installation guide for your VAX-11 processor.

### 5.8.4 Restoring the System Disk (Using Stand-alone BACKUP)

To restore the system disk, you need to use Stand-alone BACKUP. You should follow the procedures given in the software installation guide for your VAX-11 processor.

### 5.8.5 Backing Up a Public Disk to Disk

The procedure below describes how to copy the contents of a public, or nonsystem, disk to another disk. A public disk is a disk that has been mounted with the /SYSTEM qualifier.

This procedure is performed online. Confirm that your intended target volume has sufficient capacity. You need only write-lock the source volume device to prevent users from changing any data on the disk. However, users still can read data.

Procedure

1. Issue the following command to warn all users that the disk will be dismounted and write-locked so the contents of the disk can be copied to another disk:

   REPLY/ALL/BELL "message-text"

   This message should include the name of the source disk being write-locked and indicate in how many minutes the write-lock will occur.

2. Use the SHOW/DEVICE/FILE command to ensure that all files on the disk have been closed; broadcast messages to users with open files, as necessary.

3. At the time indicated by the message, issue the DISMOUNT/NOUNLOAD command to logically dismount the source disk as follows:

   DISMOUNT/NOUNLOAD device-name

4. Write-lock the source disk by pressing the WRITE-PROTECT switch to the ON position. This switch is located on the front panel of the disk drive.

5. Mount the source disk again using the MOUNT command as follows:

   MOUNT/SYSTEM device-name: volume-label

6. Allocate a drive for the target volume with the following command:

   ALLOCATE device-name:

7. Place the target volume in the allocated drive and ready the device by pressing the RUN/STOP button or the START/STOP switch.

8. Mount the target volume by issuing the following command:

   MOUNT/FOREIGN device-name:

9. Invoke the Backup Utility with the following command:

   BACKUP/IMAGE/VERIFY input-device: output-device:

10. See the VAX/VMS System Messages and Recovery Procedures Manual or the VAX-11 Utilities Reference Manual if the Backup Utility returns any verification error messages.

11. Dismount and deallocate the target volume with the following comands:

    DISMOUNT device-name:
    DEALLOCATE device-name:

12. Remove the target volume from the drive and put a label on the outside of the volume that specifies the volume label and current date.

13. Dismount the source disk by issuing the DISMOUNT/NOUNLOAD command as follows:

    DISMOUNT/NOUNLOAD device-name:

14. Write-enable the source disk by pressing the WRITE-PROTECT switch to the OFF position.

15. Remount the source disk with the MOUNT command as follows:

    MOUNT/SYSTEM device-name: volume label

16. Inform all users that the source disk is no longer write-locked by issuing the following command:

    REPLY/ALL/BELL "message-text"

**Example**

```
$ REPLY/ALL/BELL "DMA2:  WILL BE WRITE-LOCKED IN 5 MINS.  FOR BACK-UP."

_OPA0:,SYSTEM     06:31:29.78
DMA2:  WILL BE WRITE-LOCKED IN 5 MINS. FOR BACK-UP.
$ SHOW DEVICE/FILES DMA2:
      .
      .
      .
$ DISMOUNT/NOUNLOAD DMA2:
$ MOUNT/SYSTEM DMA2: PUBLIC
%MOUNT-I-WRITELOCK, volume is write locked
%MOUNT-I-MOUNTED, PUBLIC      mounted on _DMA2:
$ ALLOCATE DMA1:
  _DMA1: Allocated
$ MOUNT/FOREIGN DMA1:
%MOUNT-I-MOUNTED, PUBLIC      mounted on _DMA1:
$ BACKUP/IMAGE/VERIFY DMA2: DMA1:
$ DISMOUNT DMA1:
$ DEALLOCATE DMA1:
$ DISMOUNT/NOUNLOAD DMA2:
$ MOUNT/SYSTEM DMA2: PUBLIC
%MOUNT-I-MOUNTED, PUBLIC     mounted on _DMA2:
$ REPLY/ALL/BELL "DMA2: IS NO LONGER WRITE-LOCKED."

_OPA0:,SYSTEM     06:46:44,23
DMA2: IS NO LONGER WRITE-LOCKED.

$
```

The operator informs all system users that DMA2 will be dismounted and write-locked for back-up purposes. The operator logically dismounts the source disk, write-locks it, and then remounts it. After remounting the source disk, the operator performs the necessary steps to mount and ready the target disk.

The operator then issues the BACKUP command to perform an image copy from DMA2 to DMA1 with verification. The operator dismounts and deallocates the target volume and removes it for storage. The source volume is dismounted so that it can be write-enabled. Once this is done, the operator notifies the users that DMA2 is available and the dollar sign prompt ($) returns.

The operator then dismounts and deallocates the target disk, dismounts, write-enables, and remounts the disk on DMA2, and finally, informs all users that they can write to DMA2.

### 5.8.6 Selective Back-Up of Files Using BACKUP

The procedure below describes how to copy selected files from one disk to another. Selective back-ups may be necessary for certain groups of files requiring special treatment. Generally, if files must be backed up regularly, you should create a command procedure that contains the required back-up commands.

For more information on creating command procedures, refer to the VAX/VMS Guide to Using Command Procedures.

**Procedure**

1. Allocate a drive for the target volume with the command:

   ```
   ALLOCATE device-name:
   ```

2. Place the target volume in the allocated drive. Ready that device by pressing either the RUN/STOP button or the START/STOP switch.

3. Mount the target volume with the command:

   ```
   MOUNT output-device-name:  volume-label
   ```

4. Issue the following command to allocate a drive for the source disk:

   ```
   ALLOCATE source-device-name:
   ```

5. Place the source disk in the allocated drive and ready that device by pressing the RUN/STOP button or the START/STOP switch.

6. Mount the source disk by typing:

   ```
   MOUNT source-device-name:  volume-label
   ```

7. Copy the files from the source disk to the target volume with the following command:

   ```
   BACKUP input-spec output-spec
   ```

   This command is repeated as necessary for the sets of files that are to be saved. Note that BACKUP creates output directories automatically.

8. Dismount and deallocate the target volume with the commands:

   ```
   DISMOUNT device-name:
   DEALLOCATE device-name:
   ```

9. Remove the target volume from the device and affix a label to the outside of it that indicates the volume label and the current date.

10. Dismount and deallocate the source disk with the commands:

    ```
    DISMOUNT device-name:
    DEALLOCATE device-name:
    ```

11. Remove the source disk from the device.

**Example**

```
$ ALLOCATE DMA0:
  _DMA0:  ALLOCATED
$ MOUNT DMA0:  SPBACKUP
%MOUNT-I-MOUNTED, SPBACKUP           mounted on _DMA0:
$ ALLOCATE DMA1:
  _DMA1:  ALLOCATED
$ MOUNT DMA1:  DATCOM
%MOUNT-I-MOUNTED, DATCOM          mounted on _DMA1:
$ BACKUP DMA0:[GEORGE...]  DMA1:[*...]/OWNER_UIC=ORIGINAL
$ BACKUP DMA0:[SYSEXE]DUNGEON.*;*  DMA1:[SYSEXE]
$ DISMOUNT DMA0:
$ DEALLOCATE DMA0:
$ DISMOUNT DMA1:
$ DEALLOCATE DMA1:
```

The operator copies files from three directories on DMA1 to three newly created directories on DMA0. After performing the necessary steps to mount and ready the target and source volumes, the operator uses BACKUP commands to copy:

- All the files in the [GEORGE] directory and its subdirectories on the source disk to the directories with the same name on the target volume
- All the files with the file name DUNGEON in the [SYSEXE] directory on the source disk to the [SYSEXE] directory on the target volume

When the back-up operation completes, the operator dismounts and deallocates the target and source volumes.

### 5.8.7 Incremental Back-ups

Rather than save all the files on a volume every time a save is performed, it is better to save only those files that were created or modified since the last save operation. This is termed an incremental back-up.

To perform incremental back-ups, use the /SINCE=BACKUP input qualifier and the /RECORD command qualifier. The /SINCE=BACKUP input qualifier directs BACKUP to select only those files that have been created or modified since the last BACKUP/RECORD operation. The /RECORD qualifier directs BACKUP to record the current date in the back-up date field of each file's header.

For example:

```
$ BACKUP/RECORD DB2:[*...]/SINCE=BACKUP MTA0:19JUN.BCK
```

To use the /RECORD command qualifier, you must own the files or have the user privilege SYSPRV.

NOTE

If you use the /RECORD command qualifier to perform incremental back-ups on disk volumes, it is a good idea to discourage its use by other users. Incomplete back-ups may occur if other users perform back-ups using the /RECORD command qualifier.

The drawback to performing incremental back-ups is that you accumulate a large number of tape or disk volumes containing the incremental save sets. You should perform incremental back-ups at regular intervals (daily and weekly, for example) and full back-ups at greater intervals (once a month, for example).

### 5.8.8 Performing Daily Back-up Operations

In performing daily back-ups, you would follow the suggestions in Section 5.8.7 for incremental back-ups, but may want to include the /IGNORE=INTERLOCK qualifier. This qualifier instructs the Backup Utility to copy files even if they are open for writing. This means that you do not need to write-lock the source volume prior to initiating the back-up operation.

**Example**

```
$ MOUNT/FOREIGN MTA0:
%MOUNT-I-MOUNTED, INCD5J mounted on MTA0:
$ BACKUP/IGNORE=INTERLOCK/RECORD/SINCE=BACKUP PUBLIC:[*...] MTA0:INCD12JUN
$ DISMOUNT MTA0:
```

### 5.8.9 Performing Weekly Back-up Operations

You can perform weekly back-ups in much the same manner as daily back-ups. However, the /SINCE qualifier specifies the date of the the last weekly back-up, normally one week earlier.

**Example**

(This example assumes that the current date is 14-JUN-1982.)

```
$ MOUNT/FOREIGN MTA0:
%MOUNT-I-MOUNTED, INCW7J mounted on MTA0:
$ BACKUP/IGNORE=INTERLOCK/RECORD/SINCE=7-JUN-1982 PUBLIC:[*...] MTA0:INCW14JUN
$ DISMOUNT MTA0:
```

### 5.8.10 Performing Monthly Back-up Operations

Monthly back-ups should be all-inclusive; that is, they should be full back-ups. Thus, you specify the /IMAGE qualifier to copy not just the files, but all the information required to initialize the volume or volume set when you need to perform a restore operation.

**Example**

```
$ MOUNT/FOREIGN MTA0:
%MOUNT-I-MOUNTED, FULLMA mounted on MTA0:
$ MOUNT/FOREIGN MTA1:
%MOUNT-I-MOUNTED, FULL02 mounted on MTA1:
$ BACKUP/IMAGE/IGNORE=INTERLOCK/RECORD PUBLIC: MTA0:FULLJUN82,MTA1:
%BACKUP-I-RESUME, resuming operation on volume 2
%BACKUP-I-RESUME, resuming operation on volume 3
%BACKUP-I-RESUME, resuming operation on volume 4
 .
 .
 .
$ DISMOUNT MTA0:
$ DISMOUNT MTA1:
```

### 5.8.11 Backing Up and Restoring the Console Medium

The procedures for backing up and restoring the console medium are processor-dependent. You can find descriptions of the procedures in the software installation guide for your VAX-11 processor.

### 5.8.12 Sequential Disk Save Sets

The Backup Utility can treat a disk volume as it would a tape volume. In such a case the disk volume is called a sequential disk save set. However, there are a few minor differences, as described in this section. A disk volume that contains a sequential disk save set must be mounted with the /FOREIGN qualifier. The Backup Utility manages the file structure on the volume.

To use sequential disk save sets consisting of more than one volume, you must have the user privilege LOG_IO.

An example of a system where sequential disk save sets would be used is a system having a large fixed-media disk and a small removable disk but no tapes. The small disk would be used to contain sequential disk save sets.

**5.8.12.1 Saving Data to Sequential Disk Save sets -** To save data to a sequential disk save set, make certain that the output-specifier contains a device name for a disk device, a save-set-name, and the /SAVE_SET qualifier. The output-specifier cannot contain a directory name. For example:

```
$ BACKUP [ROGERS...] DLA0:29JUN.BCK/SAVE_SET
```

The output disk is initialized by default. The volume label is either derived from the save-set-name or specified with the /LABEL output save set qualifier. If there are files on the output disk that you want to save, use the /NOINITIALIZE qualifier. For example:

```
$ BACKUP/NOINITIALIZE [JONES...] DLA0:30JUN.BCK/SAVE_SET
```

Note, that the following restrictions apply if you use the /NOINITIALIZE qualifier:

- The disk must be Files-11 Structure Level 2
- The disk must not be part of a volume set
- The cluster factor on the disk must be 1
- The free space on the disk cannot be fragmented into more than 100 pieces
- The index file cannot be extended
- The Master File Directory (MFD) cannot be extended

The last two restrictions limit the number of save sets that you can place on a sequential disk to approximately 20.

NOTE

A set of disks written with BACKUP's sequential disk handling is referred to

as a loosely coupled volume set. That is, it is a volume set without some of the informational structures that are present in a normal volume set, such as the volume set list file. Because of the subtle differences in the structure, you should not write files onto a sequential disk volume as if it were a normal Files-11 disk; confusing and somewhat obscure errors may result. Because the sequential disk volumes are part of a volume set, they cannot be processed individually by the Verify Utility.

**5.8.12.2 Restoring Data from Sequential Disk Save Sets** - VAX/VMS can read a sequential disk save set either as a sequential disk (if the volume is mounted using the /FOREIGN qualifier) or as a Files-11 volume. If the volume is read as a sequential disk, the default directory for the save set file-specifier is [000000]. If the volume is read as a Files-11 volume, the default directory is the default directory for your process; to read the save set you must specify the directory [000000].

The following two examples illustrate a user reading data first as a sequential disk and then as a Files-11 volume.

```
$ MOUNT/FOREIGN DLA0:
%MOUNT-I-MOUNTED, 29JUN01 mounted on _DLA0:
$ BACKUP DLA0:29JUN.BCK/SAVE_SET  DB2:[ROGER...]
$

$ MOUNT DLA0: 29JUN01
%MOUNT-I-MOUNTED, 29JUN01 mounted on _DLA0:
$ BACKUP DLA0:[000000]29JUN.BCK/SAVE_SET  DB2:[ROGER...]
$
```

Multivolume save sets are handled differently when you read the disk as a sequential disk than when you read the disk as a Files-11 disk. If you read the disk as a sequential disk, the volumes must be mounted one at a time as you would a multivolume tape save set. If you read the disk as a Files-11 disk, all the volumes must be mounted (as for any Files-11 volume set).

**5.8.12.3 Summary** - The following items are identical when dealing with sequential disk save sets or tape volume save sets:

- Volume must be mounted using the /FOREIGN qualifier
- Volumes can be mounted one by one
- More than one device can be specified to overlap mounting and dismounting the volumes

The following items differ between tape save sets and sequential disk save sets.

- Save operations to tape simply require a tape device to be specified in the output-specifier; Save operations to sequential disks require a disk device to be specified in the output-specifier, and the /SAVE_SET qualifier must be specified with the output-specifier.

- To place more than one save set on a tape, the /NOREWIND output save-set qualifier must be specified; to place more than one save set on a sequential disk, the /NOINITIALIZE command qualifier must be specified.

- LOG_IO privilege is necessary to write multivolume sequential disk save sets; however, it is not necessary to write tapes.

## 5.9 BACKUP JOURNAL FILES

A BACKUP journal file contains records of BACKUP save operations and the file specifications of the files that were saved with each operation. To find a particular file in a multivolume save set, you can review the BACKUP journal file to find the tape volume that contains the file.

Use the /JOURNAL command qualifier to create, or append information to, a BACKUP journal file. If no file-specifier appears with the /JOURNAL command qualifier, the name of the BACKUP journal file defaults to SYS$DISK:BACKUP.BJL. The default file type for BACKUP journal files is BJL.

If the specified BACKUP journal file does not exist, it is created; if it already exists, the new journal information is appended to the existing journal file. You can start a new version of a BACKUP journal file by creating an empty file with an editor such as EDT or the DCL command CREATE.

To list a BACKUP journal file, enter the command:

```
BACKUP/LIST[=file-spec]/JOURNAL[=file-spec]
```

You must not specify an input- or output-specifier with a BACKUP/JOURNAL/LIST command. If the file specification is omitted from the /LIST qualifier, output is directed to SYS$OUTPUT; if the file specification is omitted from the /JOURNAL qualifier, the default BACKUP journal file is used.

You can use file selection qualifiers with the BACKUP/JOURNAL/LIST command. This allows you to locate a set of files in a save set just as the DIRECTORY command allows you to locate a set of files on a disk. The following example shows all files in the directory [SMITH.PROGS] backed up after July 5, 1982, listed in the BACKUP journal file ARCH.BJL:

```
$ BACKUP/LIST/JOURNAL=ARCH.BJL/SELECT=[SMITH.PROGS]/SINCE=5-JUL-1982
```

The following example shows the use of BACKUP journal files.

```
$ BACKUP/JOURNAL/LOG/IMAGE  DRA2: MTA0:3JUL.FUL
                .
                .
                .
        (first set is printed here)
                .
                .
                .
%BACKUP-I-RESUME, resuming operation on volume 2
%BACKUP-I-READYWRITE, mount volume 2 on MTA0: for writing
Press return when ready: RET
                .
                .
                .
        (second set is printed here)
                .
                .
                .
$ BACKUP/JOURNAL/LIST
Listing of BACKUP journal
Journal file DB2:[SYSMGR]BACKUP.BJL;1 on 3-JUL-1981 00:40:56.36
```

```
Save set 3JUL.FUL created on 3-JUL-1982 00:40:56.36
Volume number 1, volume label 3JUL01
    [COLLINS]ALPHA.DAT;4
    [COLLINS]EDTINI.EDT;5
    [COLLINS]LOGIN.COM;46
    [COLLINS]LOGIN.COM;45
    [COLLINS]MAIL.MAI;1
    [COLLINS]MAR.DIR;1
    [COLLINS.MAR]GETJPI.EXE;9
    [COLLINS.MAR]GETJPI.LIS;14
      .
      .
      .
    [LANE]LES.MAI;1

    Save set 3JUL.FUL created on 3-JUL-1982 00:40:56.36
    Volume number 2, volume label 3JUL02
    [LANE]MAIL.MAI;1
    [LANE]MEMO.RNO;5
    [LANE]MEMO.RNO;4
      .
      .
      .
    [WALTERS.VI]KD.RNO;52

    End of BACKUP journal
```

### 5.9.1 Backing Up Volumes and Volume Sets

To back up an entire disk volume or volume set, use the command BACKUP/IMAGE. The /IMAGE qualifier directs BACKUP to create a save set that contains data necessary for reinitializing the disk volume. You cannot use other file selection qualifiers with the /IMAGE command qualifier. All files on the disk are saved; this includes reserved files and lost files (files that have no directory entry).

The following example shows a save operation from a Files-11 disk to tape:

```
$ BACKUP/IMAGE DRA1:   MTA0:1JUN.BCK
```

If you use tape as the back-up medium, you may need to mount additional tapes. The number of tapes depends on the size of the disk being saved.

If you use disks as the back-up medium, you can either use BACKUP to copy files to the new disk or you can create a save set on the new disk. If you create a save set on the new disk, you must create a directory to which the save set will be written and you must use the /SAVE_SET output save set qualifier. The directory must be included in the save-set-name or it must be your default directory.

For example:

```
$ SHOW DEFAULT
  DMA1:[SYSTEM]
$ BACKUP/IMAGE DMA1: DB4:[BACKUPS]21JANDMA1.BCK/SAVE_SET
```

You should specify the /RECORD command qualifier if you are performing the full volume back-up in conjunction with incremental back-ups that use the /RECORD qualifier.

**5.9.1.1 Backing Up An Entire Volume Set** - You can back up an entire volume set by following the procedures outlined in Section 5.9.1. Simply name the device on which the root volume (volume number 1) is mounted.

**5.9.1.2 Backing Up a Disk Volume Set When Drives Are Limited** - BACKUP requires that you mount all volumes of a volume set for back-up. Thus, it may not be possible to copy a large volume set directly to disk if you have a limited number of disk drives. You can copy the volume set one volume at a time with the BACKUP/IMAGE/VOLUME command using one more drive than the number of volumes in the volume set. You must write-lock the volume set during the entire procedure to ensure consistency.

**Example**

```
(suitable warning broadcasts)
      .
      .
      .
$ DISMOUNT/NOUNLOAD DRA0:
$ MOUNT/SYSTEM/NOWRITE DRA0:,DRA1:,DRA2: PUBLIC01,PUBLIC02,PUBLIC03
%MOUNT-I-MOUNTED, PUBLIC01 mounted on DRA0:
%MOUNT-I-MOUNTED, PUBLIC02 mounted on DRA1:
%MOUNT-I-MOUNTED, PUBLIC03 mounted on DRA2:
$ MOUNT/FOREIGN DRA3:
%MOUNT-I-MOUNTED, SCRATCH01 mounted on DRA3:
$ BACKUP/IMAGE/VOLUME=1 DRA0: DRA3:
$ DISMOUNT DRA3:
$ MOUNT/FOREIGN DRA3:
%MOUNT-I-MOUNTED, SCRATCH02 mounted on DRA3:
$ BACKUP/IMAGE/VOLUME=2 DRA0: DRA3:
$ DISMOUNT DRA3:
$ MOUNT/FOREIGN DRA3:
%MOUNT-I-MOUNTED, SCRATCH03 mounted on DRA3:
$ BACKUP/IMAGE/VOLUME=3 DRA0: DRA3:
$ DISMOUNT DRA3:
$ DISMOUNT/NOUNLOAD DRA0:
$ MOUNT/SYSTEM DRA0:,DRA1:,DRA2: PUBLIC01,PUBLIC02,PUBLIC03
%MOUNT-I-MOUNTED, PUBLIC01 mounted on DRA0:
%MOUNT-I-MOUNTED, PUBLIC02 mounted on DRA1:
%MOUNT-I-MOUNTED, PUBLIC03 mounted on DRA2:
      .
      .
      .
(announce public disk is available again)
```

The operator needs to back up a three-volume volume set. Thus, at least four drives are required. The operator warns the users that drives DRA0, DRA1, and DRA2 are about to be write-locked for back-ups. Next the operator issues MOUNT commands for the volumes PUBLIC01, PUBLIC02, and PUBLIC03 on drives DRA0, DRA1, and DRA2, respectively, to specify write-locking with the /NOWRITE qualifier. Note that PUBLIC01 is the root volume. A scratch volume is mounted on drive DRA3 to be used for the target volumes. As the volume on DRA3 is filled, it is dismounted and a new scratch volume is mounted. This procedure repeats until the third volume has been copied. Then the last volume on DRA3 is dismounted. To enable writing on the volume set again, the operator dismounts (but does not unload) DRA0, then mounts the volumes again, omitting the /NOWRITE qualifier. As a final step, the operator announces to the users that the volume set is available again for writing.

### 5.9.2 Restoring Entire Disk Volumes

To restore an entire disk volume from a save set that was created using the /IMAGE command qualifier, you must mount the new volume using the DCL command MOUNT/FOREIGN. You must then restore the volume with the BACKUP/IMAGE command.

When a save set is created using the /IMAGE command qualifier, the data necessary for reinitializing the volume is placed in the save set. When the /IMAGE command qualifier is used to restore the volume, the new volume is initialized using that data in the save set.

For example:

```
$ MOUNT/FOREIGN DRA1
%MOUNT-I-MOUNTED, 24JUND mounted on _DRA1:
$ BACKUP/IMAGE MTA0:24JUNDMA1.BCK  DRA1:
$
```

The volume is initialized using the initialization parameters saved in the save set. All files and directories in the save set are restored to the new volume.

To restore a volume set in image mode, you must mount all the volumes of the set as foreign volumes. You must also specify the list of devices on which the volumes are mounted in the output-specifier of the BACKUP command.

```
$ MOUNT/FOREIGN DRA0:
%MOUNT-I-MOUNTED, PUBLIC01 mounted on DRA0:
$ MOUNT/FOREIGN DRA1:
%MOUNT-I-MOUNTED, PUBLIC02 mounted on DRA1:
$ MOUNT/FOREIGN DRA2:
%MOUNT-I-MOUNTED, PUBLIC03 mounted on DRA2:
$ BACKUP/IMAGE MTA0:31JUNPUB DRA0:,DRA1:,DRA2:
```

The volumes receive volume set numbers in the order in which they are listed in the BACKUP command. In this example, DRA0:, DRA1:, and DRA2: become volumes 1, 2, and 3, respectively.

**5.9.2.1 Changing Volume Initialization Parameters Before Restoring -** If you need to change the volume initialization parameters for a volume, you must:

1. Initialize the new volume using the new initialization parameters.
2. Mount the new volume using the /FOREIGN qualifier.
3. Restore the volume with BACKUP, using the /NOINITIALIZE command qualifier.

In the following example the cluster size of the volume is being changed to 3. The other volume initialization parameters take the default values from the INITIALIZE command.

```
$ ALLOCATE DRA2
  _DRA2: ALLOCATED
$ INITIALIZE/CLUSTER_SIZE=3 DRA2: TEST_PROGS
$ MOUNT/FOREIGN DRA2
%MOUNT-I-MOUNTED, TEST_PROGS mounted on DRA2:
$ BACKUP/IMAGE/NOINITIALIZE/TRUNCATE  MTA0:1JUN.BCK  DRA2:
```

The only initialization parameter that you cannot change is the structure level of the volume. If you change the cluster factor, as in the above example, it is good practice to include the /TRUNCATE qualifier.

**5.9.2.2 Restoring a Volume From Incremental Back-ups** - If you have been performing a combination of full and incremental back-ups on a public volume, you must use the following procedure to recover the volume from its back-ups, should the volume be lost.

First, restore the volume from the last full back-up, using an image restore operation. The /RECORD qualifier is required for the correct operation of this procedure.

```
$ MOUNT/FOREIGN DRA0:
%MOUNT-I-MOUNTED, SCRATCH mounted on DRA0:
$ MOUNT/FOREIGN MTA0:
%MOUNT-I-MOUNTED, FULLMA mounted on MTA0:
$ MOUNT/FOREIGN MTA1:
%MOUNT-I-MOUNTED, FULL02 mounted on MTA1:
$ BACKUP/IMAGE/RECORD MTA0:FULLJUN82,MTA1: DRA0:
%BACKUP-I-RESUME, resuming operation on volume 2
%BACKUP-I-RESUME, resuming operation on volume 3
%BACKUP-I-RESUME, resuming operation on volume 4
 .
 .
 .
$ DISMOUNT MTA0:
$ DISMOUNT MTA1:
$ DISMOUNT/NOUNLOAD DRA0:
```

Now mount the disk as a file-structured volume and apply the incremental back-ups in reverse chronological order. Start with the last daily back-up; then apply the preceding daily back-ups, and finally the weekly back-ups, as follows:

```
$ MOUNT DRA0: PUBLIC
%MOUNT-I-MOUNTED, PUBLIC mounted on DRA0:
$ MOUNT/FOREIGN MTA0: INCD17
%MOUNT-I-MOUNTED, INCD17 mounted on MTA0:
$ BACKUP/INCREMENTAL MTA0:INCD17JUN DRA0:
$ DISMOUNT MTA0:

$ MOUNT/FOREIGN MTA0: INCD16
%MOUNT-I-MOUNTED, INCD16 mounted on MTA0:
$ BACKUP/INCREMENTAL MTA0:INCD16JUN DRA0:
$ DISMOUNT MTA0:

$ MOUNT/FOREIGN MTA0: INCD15
%MOUNT-I-MOUNTED, INCD15 mounted on MTA0:
$ BACKUP/INCREMENTAL MTA0:INCD15JUN DRA0:
$ DISMOUNT MTA0:

$ MOUNT/FOREIGN MTA0: INCW14
%MOUNT-I-MOUNTED, INCW14 mounted on MTA0:
$ BACKUP/INCREMENTAL MTA0:INCW14JUN DRA0:
$ DISMOUNT MTA0:

$ MOUNT/FOREIGN MTA0: INCW7J
%MOUNT-I-MOUNTED, INCW7J mounted on MTA0:
$ BACKUP/INCREMENTAL MTA0:INCW7JUN DRA0:
$ DISMOUNT MTA0:
```

Applying the latest incremental back-up using the /INCREMENTAL qualifier causes the volume's directories to be restored to their state at the time the back-up was taken. In addition, all the files in the incremental save set are restored. Files that are present on the volume from the full restore operation, but are not present in the directories of the incremental back-up, are deleted. (These files were deleted by users during the time period between the full back-up and the last incremental back-up.)

In applying the earlier incremental back-ups, BACKUP restores the remaining files that have directory entries on the volume. These are files that were last modified some time before the last incremental back-up, and were still present when the last incremental back-up was taken. Note that BACKUP will restore the volume correctly regardless of the order in which the incremental back-ups are applied; using reverse chronological order is most efficient. The /RECORD and /INCREMENTAL qualifiers must be used where shown to obtain the correct operation.

If you choose to selectively exclude certain files in your incremental back-ups (for example, listing files or batch logs), these files will not be restored, but will have directory entries in the resulting volume. You can clean up these "null" directory entries by running a repair pass with the Verify Utility (see the VAX-11 Utilities Reference Manual).

### 5.9.3 Restoring Individual Files

To restore individual files or directories, use the BACKUP restore commands documented in the Backup Utility chapter of the VAX-11 Utilities Reference Manual.

To restore individual files in large save sets, use the BACKUP/LIST/JOURNAL command to find the volume that contains the files. Mount the volume, then enter the BACKUP command to select and restore the desired files. If the volume is not the first volume in a multivolume save set, you will receive the warning message:

```
%BACKUP-W-NOT1STVOL, tape 'name' is not the start of a save set
```

### 5.9.4 BACKUP Media Security

Remember that the file system treats a BACKUP save set, whether it is stored on disk or on tape, as a single file. BACKUP does not check protection on individual files within the save set. Therefore, it is crucial to the system's file security to protect save sets adequately. Give save set files that you keep online a restrictive file protection. (Use the /OWNER_UIC and /PROTECTION qualifiers to the BACKUP command, as described in the VAX-11 Utilities Reference Manual.) Provide physical security for save sets that you keep offline (for example, tapes and sequential disks); preferably you should lock them up.

If a user comes to you wanting to restore a particular file, you should not loan out the back-up tape because you would give out access to all the files on the tape. The safest way to restore a particular file is for you to mount the tape and restore the file with a command of the form:

```
$ BACKUP MTA0:SAVESET/SELECT=[JONES.TEXTPROC]LASTMONTH.DAT [*...]/OWNER_UIC=ORIGINAL
```

The file will be restored with its original directory, ownership, and protection, allowing the file system to determine whether the user was ever allowed access to the file.

## 5.10 DISK SPACE MANAGEMENT

Parkinson's Law, as applied to public disk storage, states that user files will expand to exceed the available disk storage space. You have two tools at your disposal to combat this problem:

- File expiration
- Disk quotas

### 5.10.1 File Expiration

File expiration is a file system feature (available on Files-11 Structure Level 2 disks only) that uses the expiration date of each file to track the file's use. The expiration dates aid the disposal of seldom-used files.

To enable the setting of expiration dates, use the command

```
SET VOLUME devname: /RETENTION=(min,max)
```

For min and max you specify the minimum and maximum retention periods for files on the volume, expressed as delta time values. For example, the following command sets the minimum retention period to 15 days and the maximum to 20 days:

```
$ SET VOLUME DRA0: /RETENTION=(15-0:0,20-0:0)
```

The retention periods operate as follows: every time a user accesses a file (for either a read or write operation), and that file's expiration date is earlier than the current date plus the minimum retention period, the file's expiration date is updated to the current date plus the maximum retention period. Thus, the expiration date of a frequently accessed file fluctuates between the minimum and maximum period plus the current date. When you set a suitable interval between minimum and maximum retention periods, you can trade between accuracy and efficiency in maintaining expiration dates. The difference between the two periods is the interval at which the expiration date of a frequently accessed file will be updated.

If you specify only a single value in the SET VOLUME/RETENTION command, it becomes the minimum retention period; the maximum retention period is set to twice the minimum or the minimum plus 7 days, whichever is less.

You can simulate the maintenance of "access dates," available in some other operating systems, by setting the retention periods to very small values (for example, 1 hour). Note, however, that doing so will incur substantial overhead in the file system in updating expiration dates so frequently.

This feature does not automatically remove unused files; it maintains expiration dates to permit you to develop your own policy for handling files with little or no activity. For example, the following BACKUP command will copy to tape and then delete all expired files:

```
$ BACKUP/DELETE PUBLIC:[*...]/BEFORE=TODAY/EXPIRED MTA0:ARCH20JUN
```

(Plan to retain the resulting tape for a substantial period of time unless you are unperturbed by user ire.)

NOTE

If you start maintaining expiration dates on a previously existing volume, you should be aware that the expiration dates on existing files are zero (until the files are accessed). Files with expiration dates of zero are considered expired.

### 5.10.2 Disk Quotas

You limit the amount of space available to individual users on public volumes (or volume sets) by creating and maintaining quota files on those volumes. Individual users can similarly restrict usage on private volumes. Quotas are maintained and enforced on a per-volume basis. Each volume or volume set has its own quota file; a volume on which quotas are not maintained has no quota file; on a volume set, volume 1 contains the quota file. Each entry in a quota file includes the following information:

- UIC -- UIC of a user entitled to maintain files on the volume
- Usage -- Number of blocks on the volume taken up by the user's files
- Quota -- Maximum number of blocks on the volume that the user's files can take up before an error message is issued
- Overdraft -- Number of blocks over the quota that the user's files can take up

The absolute maximum number of blocks permitted a user on a volume is the sum of the quota and the overdraft.

You (or the user maintaining the volume) identify UICs and assign quotas and overdrafts with the Disk Quota Utility (see the VAX-11 Utilities Reference Manual). Usage counts are maintained automatically by the system during normal file activities.

The name of the quota file is [000000]QUOTA.SYS on the applicable volume.

A quota file is initialized with an entry for UIC [0,0]. The usage count for this UIC should not change from 0 -- the UIC should own no files. Its quota and overdraft, however, serve as defaults in certain situations, and so should be set to values most likely to be assigned as quotas and overdrafts to other UICs.

A quota file requires one block of disk storage for each 16 entries.

**5.10.2.1 Disk Quota Operations** - During normal use of a volume with a quota file, the system automatically updates the usage counts as users create, delete, extend, and truncate files. Users without entries in the quota file are not allowed to create files or allocate space on the volume, unless they have the EXQUOTA privilege.

**5.10.2.1.1 Exceeding the Quota** - If an operation to add a new file or expand a current file will put a user's usage count over the quota, the system prohibits the operation and issues the following message:

```
disk quota exceeded
```

If the rejected operation is an extension of a file opened for write, a user with an overdraft can perform the operation by retrying it. Operations to extend the file will succeed until usage exceeds the sum of the quota and the overdraft. At this point, the system reissues the above message and prohibits further extensions to the file.

To create new files, a user's usage must be below quota (not overdraft).

Quota restrictions are not enforced for users with the EXQUOTA privilege. However, their usage counts are maintained.

**5.10.2.1.2 Suspending Quotas** - The DISABLE command of the Disk Quota Utility (see the VAX-11 Utilities Reference Manual) suspends quota operations on a volume. The ENABLE command of the Disk Quota Utility lifts the suspension. In addition, quota operations on a volume can be suspended at mount time by specifying the /NOQUOTA qualifier to the DCL command MOUNT. Note that when you suspend and then resume quota operations on a volume, you will probably find incorrect usage values in the quota file. You can correct the usage values with the REBUILD command of the Disk Quota Utility or with the Verify Utility. (The VAX-11 Utilities Reference Manual describes both utilities.)

To discontinue quota operations on a volume, execute the DISABLE command, exit from DISKQUOTA, and delete the QUOTA.SYS file.

**5.10.2.1.3 Ensuring Quota File Accuracy with REBUILD on Mount** - When a volume is mounted that was not properly dismounted the last time it was used, the system performs an automatic REBUILD operation. If quotas are enforced on the volume, this action ensures that the quota file accurately reflects usage of the disk, in the event that the system failed, the volume was physically removed before being dismounted, or the WRITE PROTECT button was pushed.

**5.10.2.2 Restrictions on Other System Operations** - The following restrictions and limitations apply whether or not disk quotas are being used:

- The SYSPRV privilege is required to change the owner UIC of a file, because a change in file ownership consumes the resources of another user
- Relative volume 1 of the volume set must be online at all times.

## 5.11 ACCESSING TAPE AND DISK VOLUMES

Users may prepare their own volumes for use. Or, depending on the physical arrangement of the installation and the type of volume to be accessed, you may be called upon to assist in the preparation of volumes for use. The following are some of the reasons why your assistance may be required:

- The processor and its peripheral devices are off limits to or remotely located from some or all users
- The magnetic tape file system has requested that a tape volume be mounted
- A system or public disk needs to be mounted

Therefore, at some installations, users must communicate with you to either gain access to or create files. Tape and disk volumes must be physically mounted on devices, and the files contained on these volumes must be backed up regularly.

Physically mounting a volume means placing the volume on a specific drive and starting the drive. For tape drives, you load the tape into the drive and then press the LOAD button to start the tape drive. For disk drives, you place the disk in the disk drive and then press the START or RUN button to start the disk drive.

Before a user can access a tape or disk volume, the following steps must be performed:

1. The device on which the volume is placed should be allocated using the ALLOCATE command.
2. The volume must be physically mounted on the device.
3. New volumes must be initialized using the INITIALIZE command.
4. The user must mount the volume using the MOUNT command.

If you do not need to initialize the disk, the above steps can be accomplished by a single MOUNT command that allocates the device and requests assistance as necessary from the operator.

Allocating devices and initializing and mounting volumes are fully described in the VAX/VMS Command Language User's Guide under the ALLOCATE, INITIALIZE, and MOUNT commands and in the chapter in that manual pertaining to disk and tape volumes.

You should also consult the VAX/VMS Magnetic Tape User's Guide for more information on handling magnetic tape volumes.

The following sections discuss when and how to assist users in gaining access to files on particular volumes.

## 5.12 REQUESTS TO MOUNT VOLUMES

Requests and notifications concerning the mounting and dismounting of disk and tape volumes fall into the following categories:

- MOUNT requests -- Requests issued by the MOUNT command, on behalf of the user, to load disk and tape volumes
- MTAACP requests -- Requests issued by the magnetic tape file system (MTAACP), on behalf of the user, to load additional volumes in a magnetic tape volume set and to recover from errors
- BACKUP requests -- Requests issued by the Backup Utility, on behalf of a batch job, to load additional volumes in a multivolume save set and to recover from errors

- Mount and dismount notifications -- Messages indicating when users mount and dismount disks and tapes

You must be enabled as a disk operator to receive disk requests and enabled as a tape operator to receive tape requests.

### 5.12.1 Requests from the MOUNT Command

You receive requests to load disk and tape volumes when a MOUNT command does not complete successfully (providing the user issuing the MOUNT command has the TMPMBX privilege and does not specify /NOASSIST). See the VAX/VMS Command Language User's Guide for a description of the MOUNT command.

The specific instances in which you receive such requests are as follows:

- Volume not online -- The device is not ready with the proper volume. The request you receive states the date and time, a request number, the user name, the volume name (does not appear if not specified in the MOUNT command), and the device name, as shown in the following example:

  ```
  %OPCOM, 6-JUN-1982 17:02:30.29, request 24, from user TOM
  Please mount volume TOMVOL in device _DMA0:
  ```

  You can also receive a comment from the user. To satisfy the request, load the volume and ready the device, or redirect the mount operation to another device.

- Wrong volume -- The device contains the wrong volume. The request you receive states the date and time, a request number, the user name, the volume name (does not appear if not specified in the MOUNT command), and the device name, as shown in the following example:

  ```
  %OPCOM, 6-JUN-1982 17:02:30.29, message from user TOM
  Device _DMA0: contains the wrong volume
  %OPCOM, 6-JUN-1982 17:02:31.14, request 24, from user TOM
  Please mount volume TOMVOL in device _DMA0:
  ```

  You can also receive a comment from the user. To satisfy the request, unload the current volume in the device, load the proper volume, and ready the device, or redirect the mount operation to another device.

- In use -- The device is allocated by another user. The request and subsequent cancellation message you receive state the date and time, a request number, the user name, and the device name, as shown in the following example:

  ```
  %OPCOM, 6-JUN-1982 17:02:30.17, request 24, from user TOM
  %OPCOM, device _DMA0: is not available for mounting.
  ```

  If the requested device does not appear to be in use, attempt to free it by asking the user to whom it is allocated to deallocate it or by stopping that user's process, then load and ready the device per the request. You can also redirect the mount operation to another device.

You need not respond to the user with the REPLY command in those instances that you load and ready the requested device. The system

detects the event and so informs the user. In addition, the system sends you a message, as shown in the following example:

```
%OPCOM, 6-JUN-1982 17:02:32.28, request 24 was satisfied
```

To redirect the mount operation to another device, load the user's volume on the alternate device, ready the device, and issue a REPLY/TO command stating as the text parameter the word SUBSTITUTE (which you can specify in uppercase or lowercase, and abbreviate down to one character) followed by the name of the device. In the following example, a mount operation is redirected to DMA1:

```
$ REPLY/TO=24 "SUBSTITUTE DMA1"
```

If the user cancels a request by typing CTRL/Y, you receive a message to that effect, as shown in the following example:

```
%OPCOM, 6-JUN-1982 17:02:32.28, request 24 was canceled
```

### 5.12.2 Requests from the Magnetic Tape File System

As a tape operator, you receive requests to switch volumes during multivolume operations and to reload a volume in the event of a hardware error on a tape drive. See Chapter 3 of the VAX/VMS Magnetic Tape User's Guide for procedures on switching volumes.

On a hardware error, the request tells you the date and time, the user name, the volume name and relative volume number, and the device name. You should take one of the following actions:

- REPLY/TO -- If you can fix the problem, load (if necessary) the volume, ready the device, and issue a REPLY/TO command.
- REPLY/ABORT -- If you cannot fix the problem, you must resort to the REPLY/ABORT command.

Assume, for example, that you receive the following messages:

```
%OPCOM, 6-JUN-1982 17:02:30.58, message from user TOM
MFA0:  offline
%OPCOM, 6-JUN-1982 17:02:31.14, request 24 from user TOM
Remount relative volume 1 (TOMVOL) on MFA0:
```

You check the drive and find that it simply lost its vacuum. Remedy the situation by readying the tape drive and issuing the following command:

```
$ REPLY/TO=24

6-JUN-1982 17:02:32.31, request 24 completed by operator OPA0
```

### 5.12.3 Requests from the Backup Utility

When the Backup Utility runs as a batch job, you receive requests to load the next volume of a save set and to reload a volume in the event of an error. (These requests go to the user when the Backup Utility is invoked interactively.)

**5.12.3.1 Writing to a Save Set** - If a save operation (writing from files to a save set) requires the loading of an additional volume, you receive messages stating the date and time, a request number, the user name, and the device name, as shown in the following example:

```
%OPCOM, 6-JUN-1982 17:02:32.31, request 24, from user TOM
%BACKUP-I-READYWRITE, mount volume 2 on _MTA0:  for writing
```

To continue the back-up operation, load a scratch volume, ready the device, and type a REPLY/TO command, as shown in the following example:

```
$ REPLY/TO=24

6-JUN-1982 17:02:34.14, request 24 completed by operator OPA0
```

You can also abort the back-up operation by typing a REPLY/ABORT command, as shown in the following example:

```
$ REPLY/ABORT=24

6-JUN-1982 17:02:34.14, request 24 aborted by operator OPA0
```

**5.12.3.2 Reading from a Save Set** - If a restore operation (reading from a save set to files) requires the loading of an additional volume, you receive messages stating the date and time, a request number, the user name, and the device name, as shown in the following example:

```
%OPCOM, 6-JUN-1982 17:02:32.31, request 24, from user TOM
%BACKUP-I-READYREAD, mount volume 2 on _MTA0:  for reading
```

To continue the restore operation, place the next volume of the save set on the drive, ready the device, and type a REPLY/TO command, as shown in the following example:

```
$ REPLY/TO=24

6-JUN-1982 17:02:34.14, request 24 completed by operator OPA0
```

You can also abort the restore operation by typing a REPLY/ABORT command, as shown in the following example:

```
$ REPLY/ABORT=24

6-JUN-1982 17:02:34.14, request 24 aborted by operator OPA0
```

**5.12.3.3 Recovering from an Error** - On certain errors, you receive messages stating the date and time, a request number, the user name, the device name, and reply options. On read errors from the save set, the options are usually CONTINUE and QUIT. On write errors to the save set, the options are usually RESTART and QUIT. You should take one of the following actions:

- REPLY/TO "CONTINUE" -- If you can fix the problem and are given the CONTINUE option, ready the device, and issue a REPLY/TO command specifying the word CONTINUE as text.

- REPLY/TO "RESTART" -- If you can fix the problem and are given the RESTART option, load (if necessary) the volume, ready the device, and issue a REPLY/TO command specifying the word RESTART as text.

- REPLY/TO "QUIT" -- If you cannot fix the problem, issue a REPLY/TO command specifying the word QUIT as text.

The text in the REPLY/TO command can be uppercase or lowercase, and can be abbreviated down to the first character. See the VAX-11 Utilities Reference Manual for additional information on errors and recovery procedures.

Assume, for example, that you receive the following messages during a restore operation (the save set is being read):

```
%OPCOM, 6-JUN-1982 17:02:30.58, message from user RESJOB
%BACKUP-E-FATALERR, fatal error on MT:[]SAVE.;
%OPCOM, 6-JUN-1982 17:02:30.89, message from user RESJOB
-SYSTEM-F-MEDOFL, medium is offline
%OPCOM, 6-JUN-1982 17:02:31.05, request 24, from user RESJOB
%BACKUP-I-SPECIFY, specify option (QUIT or CONTINUE)
```

You check the drive and find that it simply lost its vacuum. Remedy the situation by readying the tape drive and issuing the following command:

```
$ REPLY/TO=24 "CONTINUE"
CONTINUE
6-JUN-1982 17:02:33.41, request 24 completed by operator OPA0
```

If you find the drive inoperable, you issue the QUIT reply:

```
$ REPLY/TO=24 "QUIT"
QUIT
6-JUN-1982 17:02:33.41, request 24 completed by operator OPA0
```

If the error occurs during a save operation (the save tape is being written), you have a choice of QUIT or RESTART:

```
%OPCOM, 6-JUN-1982 17:02:30.58, message from user RESJOB
%BACKUP-E-FATALERR, fatal error on MT:[]SAVE.;
%OPCOM, 6-JUN-1982 17:02:30.89, message from user RESJOB
-SYSTEM-F-MEDOFL, medium is offline
%OPCOM, 6-JUN-1982 17:02:31.05, request 24, from user RESJOB
%BACKUP-I-SPECIFY, specify option (QUIT or RESTART)
<You ready the tape drive.>
$ REPLY/TO=24 "RESTART"
RESTART
6-JUN-1982 17:02:33.35, request 24 completed by operator OPA0
```

The RESTART option permits you to restart a multivolume back-up operation at the beginning of the current volume. If you specify the QUIT option, you must restart the back-up operation completely.

### 5.12.4 Notification of Volume Mounts and Dismounts

If the system parameter MOUNTMSG has a value of 1, you receive a message whenever a volume is mounted. If the system parameter DISMOUMSG has a value of 1, you receive a message whenever a volume is dismounted. These parameters have an initial value of 0, so you must set them with the AUTOGEN command procedure or the SYSGEN Utility.

An interval of up to 30 seconds can pass between the mount or dismount occurring and the notification message being issued.

The notification message contains the date and time, the name of the volume, and the name of the device. The user name in the message is always SYSTEM (the ERRFMT process actually issues the message). Volume names are padded with blanks to 12 characters. The following example shows a mount notification message:

```
%OPCOM, 6-JUN-1982 17:02:30.29, message from user SYSTEM
Volume "TOMTAPE      " mounted, on physical device _MTA0:
```

The next example shows a dismount notification message:

```
%OPCOM, 6-JUN-1982 17:02:30.29, message from user SYSTEM
Volume "TOMTAPE      " dismounted, on physical device _MTA0:
```

The messages require no reply.

# CHAPTER 6

# INSTALLING IMAGES AS KNOWN IMAGES

You enhance the performance of selected executable and shareable images by installing them as known images with the Install Utility (INSTALL). INSTALL is described in the VAX-11 Utilities Reference Manual. Known images can be assigned the following attributes:

- Permanently open -- Directory information on the image file remains permanently resident, eliminating the usual directory search required to locate a file. The cost of keeping an image file permanently open is a minimum of 192 bytes of nonpaged dynamic memory.

- Header resident -- The header of the image file (native images only) remains permanently resident, saving one disk I/O operation per file access, at a cost of less than one page of paged dynamic memory. The image must also be declared permanently open.

- Privileged -- Amplified privileges are temporarily assigned to any process running the image (executable images only), permitting the process to exceed its UAF privilege restrictions during execution of the image. In this way, "normal" users can run programs that require higher-than-normal privileges.

- Protected -- A shareable image contains protected code, that is, code that runs in kernel or executive mode but that can be called by a user-level image.

- Shared -- More than one user can access the read-only and non-copy-on-reference (non-CRF) read/write sections of the image concurrently, so that only one copy of those sections ever need be in physical memory. (CRF sections always require a separate copy for each process.) The image must also be declared permanently open.

- Writeable -- When a shared non-CRF writeable section is removed from physical memory (for paging reasons or because no processes are referencing it), it is written back to the image file. Any updates made by using processes, therefore, are preserved (while the initial values are lost). The image must also be declared shared.

## 6.1 EXECUTABLE AND SHAREABLE IMAGES

Many images installed as known images are executable images. An executable image is one linked with the /EXECUTABLE qualifier or without the /SHAREABLE qualifier.

You can also install shareable images as known images. A shareable image is one linked with the /SHAREABLE qualifier. A shareable image must subsequently be linked into an executable image to be used.

Do not confuse shareable images with known images installed with the /SHARED qualifier:

- Shareable images -- A shareable image is not copied into the executable images that link with it. Thus, only one copy of the shareable image need be on disk, no matter how many executable images have linked with it.

- Shared images -- The shared attribute can be assigned to, or withheld from, any known image -- shareable or executable. Its assignment results in the creation of permanent, system global sections. Execution of non-CRF global sections requires only one copy per section to be in physical memory, no matter how many processes are running the image to which the sections belong. Global sections are created for CRF sections, but the sections are not shared in memory.

When an image is not installed, or is installed without the shared attribute, each process running the image requires private sections in memory. (A shareable image linked to an executable image need not be installed to be executed. At image execution time, the system will create private sections from the shareable image. The only exception is that a shareable image containing a writeable non-CRF section must be installed as a known image with the shared and writeable attributes.)

The number of images that can be installed with the shared qualifier is restricted by the GBLPAGES and GBLSECTIONS system parameters (see Chapter 10).

## 6.2 KNOWN FILE LISTS

The system defines known images on internal data structures called known file lists. Each known file list contains entries for all known images whose device, directory, and file type are identical. For example, all known images with the file name SYS$SYSDEVICE:[MAIN]file-name.EXE would be on one known file list, while all known images with the file name SYS$SYSDEVICE:[TEST]file-name.EXE would be on another known file list.

The number of known file lists is restricted by the KFILSTCNT system parameter (see Chapter 10). Once a known file list is created, it remains associated with a specific device and directory. If that known file list becomes empty, it cannot be reused for a different device and directory. Take care to anticipate the number of known file lists required or else avoid using up all available list heads by installing images as known files from the same device and directory wherever possible.

## 6.3 OPERATIONAL CONSIDERATIONS

Certain operational considerations come into play when known images are installed and used.

### 6.3.1 Start-up Procedures

Known file lists only last while the system is up. If the system is shut down or fails for any reason, all known images must be reinstalled after the system is booted. For this reason, the site-independent start-up command procedure, SYS$SYSTEM:STARTUP.COM, includes an INSTALL run that installs certain system programs as known images. You are encouraged to include in the site-specific start-up command procedure, SYS$MANAGER:SYSTARTUP.COM, an INSTALL run to install additional images that are run frequently, that are usually run concurrently by several processes, or that require special privileges. (See Chapter 7 for information on the start-up command procedures.)

### 6.3.2 Order of Installation

In local memory, installing less frequently used images first and more frequently used images last (on each known file list) enhances run-time performance. In MA780 multiport memory, installing the more frequently used images first enhances run-time performance.

### 6.3.3 Privileges

VAX/VMS allows images to execute in an enhanced privilege environment through two mechanisms. You can install existing executable images with extra privileges to allow a nonprivileged process to perform the privileged functions of the image. Privileged shareable images are used to implement user-written system services, which allow nonprivileged images to execute select portions of privileged code, without enhancing the privileges of each image that uses the privileged portion of code.

**6.3.3.1 Privileged Executable Images** - You install executable images with enhanced privileges through use of the /PRIVILEGED=(privilege-list) option to the Install Utility. Once executable images are installed with enhanced privileges, users should also link them with the /NODEBUG and /NOTRACE qualifiers to further maintain system integrity and system security.

**6.3.3.2 Privileged Shareable Images** - VAX/VMS supports user-written system services through a mechanism called privileged shareable images. (See the VAX/VMS Real-Time User's Guide for a description of how to create privileged shareable images.) You must link a privileged shareable image with the PROTECT=option or the /PROTECT command qualifier, so that it acquires its particular form of enhanced privileges:

- Use the PROTECT=option when only part of a privileged shareable image requires protection.
- Use the /PROTECT command qualifier when all parts of an image require protection.

You then install the privileged shareable image with the Install Utility, specifying both the /PROTECTED and /SHAREABLE qualifiers. If you fail to follow all these steps, you will prevent successful activation of a privileged shareable image.

### 6.3.4 Deleting Known Images and Dismounting Volumes

System operations are affected by two characteristics of known images:

- Deletion -- A known image is not deleted as soon as the /DELETE qualifier is applied. The deletion occurs only after all processes using the image have released it.

- Dismounting -- A volume cannot be dismounted while any known file lists associated with it contain entries.

To dismount a volume, then, you must not only delete all known images associated with it, but you must wait for all processes using those images to release them and for the system to write writeable images back to their files. You can use the DCL command SHOW DEVICES/FILES to determine the status of the files.

### 6.3.5 Shareable Image Files

At execution time, a shareable image must reside in the directory SYS$SHARE (which is normally equivalent to SYS$LIBRARY), or the file name must be defined as the logical name of the file specification of the shareable image. For example, if the file specification of a shareable image is SYS$SYSDEVICE:[TEST]STATSHR.EXE, the user must define the logical name STATSHR to the file specification before running any executable image that calls STATSHR:

```
$ DEFINE STATSHR SYS$SYSDEVICE:[TEST]STATSHR
```

The file type defaults to EXE. If the file specification for STATSHR were SYS$SHARE:STATSHR.EXE, no DEFINE command would be necessary.

Likewise, one shareable image can be substituted for another without requiring the calling executable image to relink. The user simply defines the file name of the old shareable image as the logical name of the file specification of the new shareable image. The following statement defines the file name STATSHR as the logical name of the shareable image SYS$SYSDEVICE:[MAIN]STATSHR.EXE for executable images calling STATSHR:

```
$ DEFINE STATSHR SYS$SYSDEVICE:[MAIN]STATSHR
```

Again the file type defaults to EXE. (Logical name redirection in the process or group logical name table is ignored when you run a privileged executable image.)

If the new image is installed with the /SHARED qualifier, executable images linked against the old image will be mapped to global sections for the new image. Otherwise, they will be mapped to private sections for the new image.

As demonstrated in the example, the old and new images can have the same name, but must reside in different directories. You should not substitute one version of a file for another in the same directory.

### 6.3.6 MA780 Multiport Memory

To install a shared image so that the global sections will reside in a multiport memory unit, you issue the DCL command DEFINE (an ASSIGN command could also be used) in the format:

```
DEFINE GBL$file-name shmem-name:file-name
```

The following example ensures that any global sections created for an image whose file name is STATSHR reside in the MA780 multiport memory unit whose logical name is SHRMEM1:

```
$ DEFINE GBL$STATSHR SHRMEM1:STATSHR
$ RUN SYS$SYSTEM:INSTALL
INSTALL> STATSHR/OPEN/SHARED
```

# CHAPTER 7

# START-UP AND SHUTDOWN

This chapter describes procedures for restarting the system (assuming that a complete installation has been done) and three procedures for shutting down the system. Finally, the chapter describes how to set up both the site-independent start-up command procedure and the site-specific start-up command procedure.

## 7.1 RESTARTING THE OPERATING SYSTEM

Full details for installing the VAX/VMS operating system are provided in the software installation guide for your VAX-11 processor. These descriptions cover the steps necessary to start up your system initially. Restarting the system means loading the operating system into memory and performing the necessary housekeeping functions for the system to run properly. Generally, when the system fails, it automatically restarts itself. However, sometimes your assistance is required to restart the system. This usually occurs after you have halted the operating system by one of the methods described in Section 7.2.

### 7.1.1 The Start-up Command Procedure

When the operating system is bootstrapped, the command procedure SYS$SYSTEM:STARTUP.COM is automatically executed. This DIGITAL-supplied command procedure contains commands for the site-independent operations that must be performed if the system is to run properly. These operations include assigning system-wide logical names, installing executable images as known images, and creating permanent global sections. The SYS$SYSTEM:STARTUP.COM command procedure also invokes a site-specific command procedure named SYS$MANAGER:SYSTARTUP.COM in which you place site-specific initialization commands. An empty file by this name is furnished in the VAX/VMS distribution kit. See Sections 7.3.1 and 7.3.2 for more information on these command procedures.

### 7.1.2 Bootstrapping to Restart the System

The procedure to restart the VAX/VMS operating system after it has been shut down and consequently needs to be bootstrapped varies from processor to processor. For this reason, you should refer to the software installation guide for your VAX-11 processor. To perform this procedure, you must enter the commands from the system console.

Normally you only boot the standard system (number 0). However, if you have multiple systems on your system disk, you can specify that you want to boot a particular copy of the system. The procedure requires that you load the system number (a hexadecimal value in the range 0 through F) into bits 28 through 31 of register 5 at boot time. (The technique for loading a number into register 5 is described in the software installation guide for your VAX-11 processor.)

### 7.1.3 Boostrapping the System with an Alternate STARTUP Command File

If you need to bootstrap your system and want to avoid autoconfiguring all devices or invoking SYS$MANAGER:SYSTARTUP.COM, you should boot following the standard procedure for your VAX-11 processor, but you should stop in SYSBOOT (see the software installation guide for your VAX-11 processor). Then issue the following command:

```
SYSBOOT>SET /STARTUP SYS$SYSTEM:STARTUP.MIN
```

Note that the start-up file must have an explicit device and directory, such as SYS$SYSTEM.

### 7.1.4 Restarting Problems

Sometimes the operating system does not bootstrap after you have issued the BOOT command. This can be caused by either a hardware or software malfunction.

**7.1.4.1 Hardware Problems** - A read error on a disk drive or console medium, or a machine check error may indicate a hardware malfunction. Whenever a hardware problem occurs, a question mark (?) character usually precedes the error message that is displayed on the system console terminal. When a hardware problem occurs, you should:

- Consult the appropriate hardware manual for your VAX-11 processor
- Contact the appropriate DIGITAL field service representative

**7.1.4.2 Software Problems** - When the operating system is loaded into memory, but the STARTUP.COM command procedure is not executed, a software malfunction has probably occurred. You would probably suspect this if the usual message specifying the number of interactive users does not appear.

You can perform one or more of the following actions to correct the situation:

- Try again, by repeating the bootstrapping procedure to restart (see the software installation guide for your VAX-11 processor)
- Place the system disk in another drive or try a different copy of the disk in the same drive and repeat the restarting procedure as above

## 7.2 SHUTTING DOWN THE OPERATING SYSTEM

There are three procedures you can use to shut down the system:

- An orderly shutdown of the system
- Two emergency shutdowns of the system

The first procedure is a command procedure that is distributed with the VAX/VMS software. This command procedure is named SYS$SYSTEM:SHUTDOWN.COM. Once invoked, SHUTDOWN.COM automatically performs specific housekeeping functions that ensure a smooth shutdown of the system. These housekeeping functions include disabling future logins, stopping the batch and device queues, dismounting mounted volumes, and stopping user processes. This procedure also invokes a site-specific command procedure named SYS$MANAGER:SYSHUTDWN.COM that you tailor to the needs of your specific installation. The SYSHUTDWN.COM file is present in the VAX/VMS distribution kit but contains no commands.

If the operating system cannot be shut down by means of the SHUTDOWN.COM command procedure, you should invoke an emergency shutdown program with the following command:

```
$ RUN SYS$SYSTEM:OPCCRASH
```

This program shuts down the system immediately. The error log buffers are written to the system dump file. Pages on the modified list are written to disk. Then the system disk is dismounted. Data may be lost, since the OPCCRASH program performs only minimal housekeeping functions. Therefore, you should only invoke an emergency shutdown of the system if the orderly shutdown procedure fails.

There is another program you can use for an emergency shutdown of the system if OPCCRASH cannot shut down the system. On the VAX-11/780, VAX-11/782, and VAX-11/730 processors, this procedure is supplied as a console command program named CRASH that is stored on the system console medium. (See Section 7.2.3, which also describes the individual instructions you need to enter to obtain equivalent results on a VAX-11/750.)

### 7.2.1 Orderly Shutdown of the System (With SHUTDOWN.COM)

The procedure below describes how to shut down the system in an orderly fashion. This procedure is contained in a command file that you should not modify. At your discretion commands can be added to the SYS$MANAGER:SYSHUTDWN.COM command procedure to perform additional housekeeping functions. You must have either the SETPRV privilege or all the following privileges to run SHUTDOWN.COM: CMKRNL, SYSNAM, OPER, WORLD, SYSPRV, and EXQUOTA. (If you have the SETPRV privilege, the procedure will automatically assign the required privileges to you.)

**Procedure**

1. Issue the following command from any terminal and any account to begin the shutdown procedure:

   ```
   $ @SYS$SYSTEM:SHUTDOWN
   ```

2. Enter an integer in response to the following question, unless you want an immediate shutdown:

   How many minutes until shutdown [0]?

3. In response to the following prompt, give the reason for shutting down the system:

   Reason?

4. Respond by typing a Y (Yes) or N (No) to the following question:

   Do you want to spin down the disks [No]?

5. Respond to the next question with a time in the format you want printed in the message that will be broadcast to the users. For example, you could specify IMMEDIATELY, or IN 10 MINUTES, or a time such as 2 P.M. or 2:00. If you do not know when the system will be available again, press RETURN.

   Expected uptime (<RET> if not known)?

6. Respond to the next question depending on whether or not you want the system to automatically reboot. By default, the system does not automatically reboot. However, if you respond with Y (Yes), the logical name OPC$REBOOT is defined as true. As a result, when the shutdown is complete, an attempt is made to automatically reboot the system. Note that the system can only be automatically rebooted if the appropriate hardware switch is also set on the processor and the default boot command file is properly set. (See the software installation guide for your VAX-11 processor.)

   The following events occur as the shutdown procedure continues, and the corresponding messages are printed on the terminal:

   a. A message that requests users to log out is broadcast to all users on the system. This message is broadcast at decreasing time intervals.

   b. Batch and device queues are stopped and all future nonoperator logins are disabled when there are four or fewer minutes left until system shutdown.

   c. Next, the site-specific command procedure SYS$MANAGER:SYSHUTDWN.COM is invoked. This command procedure contains commands inserted to tailor the shutdown procedure to the needs of the installation.

   d. All user processes are stopped. However, the following processes continue: NULL, SWAPPER, JOB_CONTROL, OPCOM, ERRFMT, ancillary control processes (ACPs), print symbionts, and the process running the SHUTDOWN.COM command procedure. ACPs may delete themselves when their mounted volumes are finally dismounted.

   e. All mounted volumes are dismounted and, if you request it, the disks are spun down. Note, however, the system disk cannot be spun down.

   f. The operator's log file is closed.

   g. The program SYS$SYSTEM:OPCCRASH is invoked to shut down the system.

7. If you did not request an automatic reboot, the following message appears on the system console:

   SYSTEM SHUTDOWN COMPLETE - USE CONSOLE TO HALT SYSTEM

   Otherwise, the system automatically reboots, provided the switches on the CPU cabinet and the boot command file are correctly set.

8. If you are not automatically rebooting, you must press CTRL/P after the above message is printed at the console terminal. Then follow the standard procedure given in the software installation guide for your VAX-11 processor to halt the system.

**Example**

```
$ @SYS$SYSTEM:SHUTDOWN

        System shutdown command procedure.

How many minutes until shutdown [0]? 5
Reason?  MONTHLY PREVENTIVE MAINTENANCE
Do you want to spin down the disks [No]? YES
Expected uptime (<RET> if not known)? 1:25
Enable automatic reboot [No]? (RET)
%OPCOM, 17-JUN-1982 11:12:20:32, operator status for operator OPA0
CENTRAL, PRINTER, TAPES, DISKS, DEVICES, CARDS, NETWORK, OPER1, OPER2,
OPER3, OPER4, OPER5, OPER6, OPER7, OPER8, OPER9, OPER10, OPER11,
OPER12

__OPA0:,SYSTEM      11:12:20.72
System shutdown in 5 minutes; up 1:25
MONTHLY PREVENTIVE MAINTENANCE

__OPA0:,SYSTEM      11:12:22.73
MONTHLY PREVENTIVE MAINTENANCE
  Login quotas - Interactive limit=0, Current interactive value=32
        Non-operator logins are disabled.
%JBC-E-SYMBDSAB, symbiont manager is disabled

__OPA0:,SYSTEM      11:14:27.30
Batch and device queues have been stopped.

__OPA0:,SYSTEM      11:14:29.86
System shutdown in 2 minutes; up 1:25.  Logins are disabled; please log out.
MONTHLY PREVENTIVE MAINTENANCE

__OPA0:,SYSTEM      11:15:32.62
System shutdown in 1 minute; up 1:25.  Logins are disabled; please log out.
MONTHLY PREVENTIVE MAINTENANCE

__OPA0:,SYSTEM      11:16:35.39
System shutdown in 0 minutes; up 1:25.  Logins are disabled; please log out.
MONTHLY PREVENTIVE MAINTENANCE

        Invoke installation dependent shutdown procedure.
                    .
                    .
                    .
        Stop all user processes.
        Remove installed images.
        Dismount all mounted volumes.
%OPCOM, 17-JUN-1982 11:16:43.62, message from user SYSTEM
__OPA0:, Operator requested shutdown
```

```
%OPCOM, 17-JUN-1982 11:16:45.02, logfile closed by operator OPA0
        logfile was SYS$MANAGER:OPERATOR.LOG
        SYSTEM SHUTDOWN COMPLETE - USE CONSOLE TO HALT SYSTEM
```

In this example, the operator requests that the system be shut down in five minutes to allow monthly preventive maintenance. The operator then indicates that the disks should be spun down, that the system is expected to be available again at 1:25, and that automatic rebooting is not desired. The system then performs housekeeping functions that ensure a clean shutdown. When housekeeping operations are complete, a system message indicates that the site-dependent shutdown procedure will be invoked. When the site-dependent shutdown procedure completes, SHUTDOWN.COM stops all user processes, dismounts all mounted volumes, and spins down the disks (except the system disk). It closes the operator's log file, then invokes OPCCRASH to shut down the system. The operator must press CTRL/P and the halting command to complete the system shutdown procedure.

### 7.2.2 Emergency Shutdown of the System (with OPCCRASH)

The procedure below describes how to halt the system immediately without performing any of the housekeeping functions that ensure a smooth shutdown. Generally, you shut down the system by following the orderly shutdown procedure described in Section 7.2.1. However, if that procedure fails, you can perform the emergency shutdown procedure described in this section.

To perform this procedure, you must possess the CMKRNL privilege. You can enter the commands from any terminal and any account.

**Procedure**

1. Enter the following command to force an immediate shutdown of the system:

   ```
   $ RUN SYS$SYSTEM:OPCCRASH
   ```

2. If the system fails to accept or to respond to the command issued in Step 1, use the appropriate alternate crash procedure for your VAX-11 processor from those that are described in Section 7.2.3.

3. Observe the following message typed on the system console:

   ```
   SYSTEM SHUTDOWN COMPLETE - USE CONSOLE TO HALT SYSTEM
   ```

4. Press CTRL/P after the above message is printed at the console terminal. Then, follow the standard procedure given in the software installation guide for your VAX-11 processor to halt the system.

**Example for a VAX-11/780**

1. ```
   $ RUN SYS$SYSTEM:OPCCRASH

        SYSTEM SHUTDOWN COMPLETE - USE CONSOLE TO HALT SYSTEM

   CTRL/P
   >>>HALT

        HALTED AT 8000708A
   ```
   ```

The VAX-11/780 operator types the command string for an emergency shutdown. The system then instructs the operator to use the console to halt the system. The operator presses CTRL/P to return the console terminal to the control of the console subsystem. The console subsystem responds with the console command language prompt (>>>). The operator then issues the HALT or H command to halt the system.

### 7.2.3 Forcing the System to Fail (with CRASH or its Equivalent)

The CRASH console program command file described below forces VAX-11/780, VAX-11/782, and VAX-11/730 systems to fail, which results in an immediate shutdown of the system. After the CRASH console program command file is invoked, a fatal bugcheck message is printed at the console terminal and the system dump file is written to the disk. Later, the dump file can be used to determine why the system did not respond to command input. However, data may be lost, since no other housekeeping functions are performed. Therefore, you should only use CRASH if the system will not accept command input; that is, if the system fails to accept or respond to OPCCRASH when you follow the instructions in Section 7.2.2.

The console program command file CRASH is not included in the VAX/VMS distribution kit for a VAX-11/750 system. If it becomes necessary to perform an emergency crash of a VAX-11/750 system, you can enter the equivalent instructions manually, as described in the second procedure below.

Note that, if a copy of the system dump file is sent to DIGITAL later for analysis, a copy of the console listing should also be sent. This listing displays fatal bugcheck information that is not contained in the system dump file (for example, program counter (PC) and processor status longword (PSL) data).

All commands that invoke the CRASH console program command file must be typed from the system console terminal.

**Procedure for VAX-11/780, VAX-11/782, and VAX-11/730 Systems**

1. Press CTRL/P to return control to the console command language level.

2. The VAX-11/780 or VAX-11/782 system responds by printing the console command language prompt (>>>). The VAX-11/730 system responds by printing a halt code and then the console command language prompt (>>>).

3. After the prompt, type HALT. You receive another console command language prompt.

   After the prompt, type @CRASH as follows:

   ```
   >>>@CRASH
   ```

4. The console command @CRASH invokes the CRASH console program command file.

   Additional messages and information, such as the fatal bugcheck message, are printed at the console terminal, as shown in the example below. The system dump file is written to the disk.

**Example for a VAX-11/780 System**

```
      CTRL/P
      >>>HALT
      >>>@CRASH

      !

      ! Command file to crash VMS abnormally

      !

      HALT                ! HALT SYSTEM, EXAMINE PC,

           HALTED AT 8000702A
      EXAMINE PSL         ! PSL,

           00000000
EXAMINE/INTERN/NEXT:4 0 ! And all stack pointers

          I   00000000   80001D48
          I   00000001   00000000
          I   00000002   00000000
          I   00000003   00000000
          I   00000004   8009E600
DEPOSIT PC=-1             ! Invalidate PC

DEPOSIT PSL=1F0000        ! Kernel mode, IPL 31

CONTINUE

<@EOF>
<@EXIT>

**** FATAL BUG CHECK, VERSION = 3.0 INVEXCEPTN, Exception while above
ASTDEL or on interrupt stack

      CURRENT PROCESS = NULL

      REGISTER DUMP

         R0 = 0000001F
         R1 = 001F0000
         R2 = 00000000
         R3 = 00000000
         R4 = 00000000
         R5 = 00000000
         R6 = 00000000
         R7 = 00000000
         R8 = 00000000
         R9 = 00000000
         R10= 00000000
         R11= 00000000
         AP = 00000000
         FP = 00000000
         SP = 80001D14
         PC = 800038C1
         PSL= 001F0009
```

```
KERNEL/INTERRUPT STACK

   80001D1C  00000004
   80001D20  00000000
   80001D24  FFFFFFFD
   80001D28  00000000
   .
   .
   .

HALT INST EXECUTED
HALTED AT 800071B3

>>>
```

Typing @CRASH invokes the CRASH console program command file on the VAX-11/780 system. This procedure instructs the system to examine the program counter (PC), the processor status longword (PSL), and the stack pointers. Values are deposited in the PC and PSL to cause an exception condition that forces a system dump. The fatal bugcheck message is printed at the console terminal. Finally, the system halts and prints the contents of the program counter. The console system prompt (>>>) returns.

If enabled, some systems will halt and then automatically rebootstrap. However, you must rebootstrap systems that are not enabled to automatically rebootstrap.

In some cases, on VAX-11/782 attached processor systems only, there may be a delay of up to two minutes before the system responds to the @CRASH command.

**Procedure for VAX-11/750 Systems**

On the VAX-11/750 you must type in the following commands at the console terminal to crash the system:

```
CTRL/P
80007B06  02
>>>E/G F
      G       0000000F       80007B06
>>>E P
      00000000
>>>E/I 0
      I       00000000       800009D0
>>>E/I 1
      I       00000001       00000000
>>>E/I 2
      I       00000002       00000000
>>>E/I 3
      I       00000003       00000000
>>>E/I 4
      I       00000004       8013C000
>>>D/G F FFFFFFFF
>>>D P 1F0000
>>>C

**** FATAL BUG CHECK, VERSION = 3.0 INVEXCEPTN, Exception while above
ASTDEL or on interrrupt stack

    CURRENT PROCESS = NULL

    REGISTER DUMP
       .
       .
       .
```

The commands instruct the system to examine the program counter (PC), the processor status longword (PSL), and the stack pointers. Values are deposited in the PC and PSL to cause an exception condition that forces a system dump. The fatal bugcheck message is printed at the console terminal. Additional messages and information, such as the register dump, are printed at the VAX-11/750 console terminal, as shown in the previous example for the VAX-11/780. The system dump file is written to the disk. Finally, the system halts and prints the contents of the program counter. The console system prompt (>>>) returns.

If enabled, some systems will halt and then automatically rebootstrap. However, you must rebootstrap systems that are not enabled to automatically rebootstrap.

## 7.3 START-UP COMMAND PROCEDURES

The software distribution kit contains two start-up command procedures:

- SYS$SYSTEM:STARTUP.COM -- Commands that, in general, must be executed at initialization time for any VAX/VMS system to run properly
- SYS$MANAGER:SYSTARTUP.COM -- An empty file, called by STARTUP.COM, that you can load with site-specific initialization commands

Although you can tailor the site-independent command procedure (STARTUP.COM) and the site-specific command procedure (SYSTARTUP.COM) with any text editor, DIGITAL discourages you from modifying STARTUP.COM. You should generally put site-specific commands in SYSTARTUP.COM rather than modifying STARTUP.COM, because a new version of the STARTUP.COM command file is provided with each major release of VAX/VMS. Furthermore, SYS$SYSTEM:STARTUP.COM is subject to change in VAX/VMS maintenance updates.

### 7.3.1 Site-independent Start-up Command Procedure

STARTUP.COM is automatically executed immediately after the operating system has been booted. The command procedure includes commands for performing housekeeping chores, assigning system-wide logical names, starting up the three processes that control error logging, the job controller, and the operator's log, installing known images, building the I/O data base and loading the I/O drivers, enabling VAX-11 RMS file sharing, calling the site-specific start-up command procedure, and logging out.

**7.3.1.1 Housekeeping Chores -** The first two commands in STARTUP.COM ensure that execution of the command procedure occurs without the commands being echoed on the terminal and without interruption on an error condition:

```
$ VERIFY = 'F$VERIFY(0)
$ SET NOON
```

The next sequence of commands determines the default directory for the location of the system executable images, ending with the following command:

```
$ SET DEFAULT SYS$SYSTEM
```

**7.3.1.2 Symbolic Debugger** - The VAX-11 Symbolic Debugger requires the following logical name assignments:

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

**7.3.1.3 RSX-11M Programs** - RSX-11M compatibility mode programs (such as BAD, SOS, and PIP) require the following logical name assignments:

```
$ ASSIGN/SYSTEM 'ROOT' LB:
$ ASSIGN/SYSTEM 'ROOT' LB0:
$ ASSIGN/SYSTEM SYS$SCRATCH WK:
$ ASSIGN/SYSTEM SYS$SCRATCH WK0:
$ ASSIGN/SYSTEM 'DISK' SP:
$ ASSIGN/SYSTEM 'DISK' SP0:
```

The definitions of the DCL symbols ROOT and DISK depend on the system number being booted and the boot device. For example, if you boot the default system from device DBA0, then ROOT is __DBA0:[SYS0.] and DISK is _DBA0:.

**7.3.1.4 System Libraries and Help Files** - The language processors, the VAX-11 Linker, the image activator, and the help processor require the following logical name assignments:

```
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSEXE] SYS$SYSTEM:
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSMSG] SYS$MESSAGE:
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSLIB] SYS$SHARE:
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSLIB] SYS$LIBRARY:
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSHLP] SYS$HELP:
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSHLP.EXAMPLES] SYS$EXAMPLES:
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSMGR] SYS$MANAGER:
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSUPD] SYS$UPDATE:
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSTEST] SYS$TEST:
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSMAINT] SYS$MAINTENANCE:
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSERR] SYS$ERRORLOG:
$ ASSIGN /SYSTEM SYS$SYSROOT:[SYSCBI] SYS$INSTRUCTION:
```

**7.3.1.5 Error Logging, Job Controller, and Operator's Log** - The next sequence of commands in STARTUP.COM starts up the error logger, the job controller and the operator's log.

**7.3.1.6 Known Images** - STARTUP.COM installs certain system executable images that are run frequently, that are usually run concurrently by several processes, or that require special privileges using the command procedure SYS$MANAGER:VMSIMAGES.COM.

**7.3.1.7 I/O Devices and Drivers** - STARTUP.COM automatically connects devices physically attached to the system and loads their I/O drivers:

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

**7.3.1.8 VAX-11 RMS File Sharing** - STARTUP.COM enables VAX-11 RMS file sharing with a maximum page count of 20:

```
$ RUN SYS$SYSTEM:RMSSHARE
20
```

**7.3.1.9 Install Deferred Swapping File if Present** - For VAX-11/730 processors with RL02 system disks, installation of the primary swapping file is deferred until after the ERRFMT, JOB_CONTROL, and OPCOM processes have been initiated. The STARTUP.COM procedure checks for the presence of the deferred swapping file, which is named SYS$SYSTEM:SWAPFILE1.SYS, and if it is present, STARTUP.COM installs it at this time with the SYSGEN command:

```
$ RUN SYS$SYSTEM:SYSGEN
INSTALL SYS$SYSTEM:SWAPFILE1.SYS /SWAPFILE
```

**7.3.1.10 Termination of the Procedure** - SYS$SYSTEM:STARTUP.COM calls the site-specific start-up command procedure, sets the number of users who can log in at one time, and logs the site-independent command procedure off:

```
$ @SYS$MANAGER:SYSTARTUP
$ SET LOGIN /INTERACTIVE=64
$ LOGOUT/BRIEF
```

### 7.3.2 Site-specific Start-up Command Procedure

SYS$MANAGER:SYSTARTUP.COM, called from STARTUP.COM, performs any commands you want to place there. Typically, these commands mount public disks, assign logical names, set the characteristics of terminals and other devices, establish and start queues, install known images, run the System Dump Analyzer, purge the operator's log file, submit batch jobs that are run at the time the system is initialized and that are periodically resubmitted, manually connect devices and multiport memory units, install secondary paging and swapping files, and announce that the system is up and running.

NOTE

The commands shown in the following sections are provided as models; do not copy them line for line.

**7.3.2.1 Disable Error Processing** - You can disable error processing with the following command:

```
$ SET NOON
```

**7.3.2.2 Public Disks** - You will probably want to include mount commands to mount your public disks for system-wide access. It is worth considering some of the advantages of using logical volume names to conceal the physical devices. When you mount a disk, MOUNT produces a special logical name called a logical volume name that you can use to reference the volume. When you dismount the volume, the logical name is deleted. For example:

```
$ MOUNT/SYSTEM  DRA1:  USERFILES  USER
```

This command produces the logical volume name USER and equates it to the concealed device name string __DRA1:. However, USER only translates to a physical device while the data disk is actually mounted.

If you mount a disk and do not give an explicit logical volume name, MOUNT assigns a default name of the form: DISK$volume_label. In the example above, if no logical volume name were specified, the default logical volume name would have been DISK$USERFILES. Since the logical volume name is printed on the flag page of listings and by the DCL commands SHOW DEVICE/FILES and SHOW MEMORY/FILES, you may occasionally see such labels.

If you and the users consistently use the logical volume name, it is not necessary to know what physical drive the volume is mounted on. Thus, you can avoid including physical device names in programs and command procedures.

Note that when you run SYSTARTUP.COM (and only then), the default on the MOUNT command is /NOASSIST. This means that operator-assisted mounts are disabled. If you want to enable this feature during SYSTARTUP, you must specify /ASSIST with each MOUNT command. See Chapter 5 for more information on mounting public disks.

**7.3.2.3 Logical Names** - You can assign system-wide logical names in addition to the logical names assigned in the site-independent start-up command procedure:

```
$ ASSIGN/SYSTEM SYS$SYSROOT:  SYSDSK
```

See the chapter, File Specifications and Logical Names, in the VAX/VMS Command Language User's Guide for more detailed information on logical name assignments.

**7.3.2.4 Optional Software Logical Name Requirements** - You may need to add logical name assignments for other components (such as VAX-11 BLISS-32 and VAX-11 BASIC).

The logical name assignments for FOR005, FOR$ACCEPT, and FOR$READ to SYS$INPUT, and FOR006, FOR$PRINT, and FOR$TYPE to SYS$OUTPUT are embedded in VAX-11 FORTRAN, and need not be stated explicitly.

If you run VAX-11 COBOL-74 Version 4.3 or earlier programs at your site, you will need the following logical name assignments:

```
$ ASSIGN/SYSTEM SYS$INPUT:  COB$INPUT
$ ASSIGN/SYSTEM SYS$OUTPUT:  COB$OUTPUT
$ ASSIGN/SYSTEM SYS$ERROR:  COB$CONSOLE
$ ASSIGN/SYSTEM SYS$INPUT:  COB$CARDREADER
$ ASSIGN/SYSTEM SYS$INPUT:  COB$PAPERTAPEREADER
$ ASSIGN/SYSTEM SYS$OUTPUT:  COB$LINEPRINTER
$ ASSIGN/SYSTEM SYS$OUTPUT:  COB$PAPERTAPEPUNCH
```

VAX-11 PASCAL programs require the following logical name assignments:

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

**7.3.2.5 Device Characteristics** - You use a series of SET commands to establish the characteristics of the terminals and other devices on the system, as in the example below. You may want to include comments that give the user names for terminal owners.

```
$ SET TERMINAL TTC2: /SPEED=300/DEVICE_TYPE=LA36/PERMANENT  !JONES
$ SET TERMINAL TTD1: /SPEED=9600/PERMANENT                  !WRENS
$ SET TERMINAL TTD4: /SPEED=1200/PERMANENT                  !JRSMITH
$ SET TERMINAL TTG4: /SPEED=1200/MODEM/PERMANENT            !DIALUP1
$ SET PRINTER  LPA0: /LOWER/NOCR
$ SET DEVICE   LPA0: /SPOOLED
```

Note that the /SPEED qualifier sets both transmission and reception speeds to the same value. The /MODEM qualifier defines a terminal for use on a dial-in line. Printer characteristics (SET PRINTER and SET DEVICE above) must be set prior to establishing queues for the printers.

**7.3.2.6 Queues** - The first time the system is booted, batch and print queues must be initialized and started. Whenever the system is rebooted, the queues must merely be started. The following commands provide for both contingencies by testing $STATUS:

```
$               START/QUEUE LPA0
$               IF $STATUS THEN GOTO ENDLPA0
$               INITIALIZE/QUEUE /FLAG LPA0
$               START/QUEUE LPA0
$ ENDLPA0:      START/QUEUE LPB0
$               IF $STATUS THEN GOTO ENDLPB0
$               INITIALIZE/QUEUE /FLAG LPB0
$               START/QUEUE LPB0
$ ENDLPB0:
```

See Chapter 8 for more information on initializing and starting queues.

**7.3.2.7 Known Images** - You often install user and system programs (in addition to the ones installed in the site-independent start-up command procedure) so that they can be located quickly, shared, or provided privileges:

```
$ RUN SYS$SYSTEM:INSTALL
BLISS32 /OPEN/SHARED/HEADER_RESIDENT
COPY /OPEN
LINK /OPEN
MACRO32 /OPEN/SHARED
DIRECTORY /OPEN
```

The most frequently used images should be installed last in local memory, but first in shared memory. See Chapter 6 for more information on installing images as known images.

**7.3.2.8 System Dump Analyzer** - Each time the system is booted, you should run the System Dump Analyzer (in case the system failed the last time it was running):

```
$ ANALYZE/CRASH_DUMP SYS$SYSTEM:SYSDUMP.DMP
COPY SYS$ERRORLOG:SYSDUMP.DMP
SET OUTPUT LPA0:SYSDUMP.LIS
SHOW CRASH
SHOW STACK/ALL
SHOW SUMMARY
SHOW PROCESS /PCB /PHD /REGISTERS
EXIT
```

If further information is required, you can invoke the System Dump Analyzer for an interactive session upon completion of start-up. (See the VAX/VMS System Dump Analyzer Reference Manual.)

**7.3.2.9 Operator's Log File** - Chapter 9 offers some suggestions for maintaining the operator's log file. If you decide not to put them into practice, you might instead add the following command to purge all but the last two or three versions of the operator's log file:

```
$ PURGE/KEEP=2 SYS$MANAGER:OPERATOR.LOG
```

**7.3.2.10 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 DCL command SUBMIT is used in the start-up file:

```
$ SUBMIT SYS$MANAGER:LOGJOBS
```

**7.3.2.11 Manually Connected Devices and Multiport Memory Units** - You run the SYSGEN Utility to connect devices not automatically connected in STARTUP.COM and to initialize or connect multiport memory units:

```
$ RUN SYS$SYSTEM:SYSGEN
SHARE MPM1 SHR_MEM_1 /INITIALIZE
```

For installations that require permanent residence of the console block storage device, you must explicitly connect the device by including the following command when you run the SYSGEN Utility:

```
$ RUN SYS$SYSTEM:SYSGEN
CONNECT CONSOLE
```

See Chapter 11 and the VAX-11 Utilities Reference Manual for a detailed discussion of the SYSGEN Utility.

The console block storage device should also be mounted with a command of the form:

```
$ MOUNT/FOREIGN/SYSTEM/PROTECTION=(SYSTEM:RWLP) CSA1:  CONSOLE
```

Failure to mount the console block storage device with appropriate protection permits users to mount and access it, because the device is in RT-11 format and has no file protection.

**7.3.2.12 VAX-11 RMS File Sharing** - STARTUP.COM enables VAX-11 RMS file sharing with a maximum page count of 20. If you estimate a larger maximum page count (see the description of the RMS Share Utility in the VAX-11 Utilities Reference Manual) you should run the RMS Share Utility with your value:

```
$ RUN SYS$SYSTEM:RMSSHARE
```

Be sure that this new value does not exceed half the size of paged dynamic memory.

**7.3.2.13 Secondary Paging and Swapping Files** - You run the SYSGEN Utility to install secondary paging and swapping files:

```
$ RUN SYS$SYSTEM:SYSGEN
INSTALL  DISK$SYS2:[SYSTEM]PAGEFILE2.SYS /PAGEFILE
INSTALL  DISK$SYS2:[SYSTEM]SWAPFILE2.SYS /SWAPFILE
```

**7.3.2.14 Announcements** - 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"
```

Before SYSTARTUP.COM exits you may want to provide site-specific definitions for one or both of the following logical names: SYS$ANNOUNCE and SYS$WELCOME. These logical names are checked whenever a user logs in and provide special messages at that time.

**7.3.2.14.1 SYS$ANNOUNCE** - SYS$ANNOUNCE defines text to be printed whenever a user begins to log in; that is, the text is printed immediately after a successful dial in, CTRL/Y, or RETURN is sensed. The text may consist of up to 63 characters. For longer messages, you can precede the name of a text-containing file with an at sign (@) so that the login command procedure prints the entire file as an announcement.

For example, you could include a command of the following form in your SYSTARTUP.COM file:

```
$ DEFINE/SYSTEM SYS$ANNOUNCE "HAVE A NICE DAY"
```

Or, you might prefer to print a file:

```
$ DEFINE/SYSTEM SYS$ANNOUNCE "@SYS$MANAGER:ANNOUNCE.TXT"
```

If you do not define SYS$ANNOUNCE, no announcement is printed.

**7.3.2.14.2 SYS$WELCOME** - SYS$WELCOME defines text to be printed whenever a user succeeds in logging in; that is, the text is printed immediately after the correct password is entered. The text may consist of up to 63 characters. For longer messages, you can precede the name of a text-containing file with an at sign (@) so that the login command procedure prints the entire file as a welcoming announcement.

For example, you could include a command of the following form in your SYSTARTUP.COM file:

```
$ DEFINE/SYSTEM SYS$WELCOME "WELCOME TO THE BEST VAX/VMS SITE IN THE USA"
```

Or, you might prefer to print a file:

```
$ DEFINE/SYSTEM SYS$WELCOME "@SYS$MANAGER:WELCOME.TXT"
```

If you do not specifically define SYS$WELCOME, the standard VAX/VMS welcome message is printed:

```
Welcome to VAX/VMS Version V3.0
```

Note that this message may end by specifying the DECnet-VAX node name if the logical name SYS$NODE is defined.

**7.3.2.15 Redefining the Number of Interactive Users -** If the number of interactive users that your site permits to log on at one time differs from 64, you should consider ending SYSTARTUP.COM with the following commands:

```
$ SET LOGIN /INTERACTIVE=n
$ LOGOUT/BRIEF
```

You specify the appropriate number of users for n. If you include the LOGOUT command, there will be no return to STARTUP.COM. This is necessary, since STARTUP.COM concludes with a SET LOGIN command that would override the value you have just specified for n.

# CHAPTER 8

# BATCH AND PRINT JOBS

System performance can be greatly influenced by how you establish spooled devices, create and control input and output queues, and control batch and print jobs. Typically, you are responsible for performing the following six closely related functions:

- 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, you specify which output devices are to be spooled. Section 8.1 describes spooling and the use of DCL commands to establish spooled devices.

- Controlling batch jobs -- Section 8.2 describes batch processing and the use of DCL commands to control batch jobs.

- Controlling print jobs -- Section 8.3 describes queuing output to line printers and the use of DCL commands to control print jobs.

- Controlling print and batch queues through DCL commands -- See Section 8.4.

- Controlling print and batch queues through procedures -- See Section 8.5.

- Using the card reader -- See Section 8.6.

You need not learn all the inner workings of spooling and queuing. However, you must have a working knowledge of how to establish spooled devices and how to create and control queues in order to keep the system running efficiently. To do so, you must first become familiar with the DCL commands listed in Table 8-1. The use of these commands is restricted to users who have operator privilege (OPER). The VAX/VMS Command Language User's Guide fully describes these commands.

In addition, three other DCL commands play a role in the control of batch and print jobs:

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

- 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 to affect your own files. However, you need the OPER, GROUP, or WORLD privilege to to use the command to modify queued jobs entered by a member of another group. You need the OPER or ALTPRI privilege to increase the queue priority of a job.

- DELETE/ENTRY -- Deletes jobs from queues.

  No privilege is needed to delete entries you have queued; however, you need group privilege (GROUP) to delete a queued job entered by another member of the same group and you need world (WORLD) or operator privilege (OPER) 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 8-1: DCL Commands Used in Regulating Spooling and Queuing**

| Command | Function |
|---|---|
| ASSIGN/MERGE | Removes jobs from queues and places them in other queues |
| ASSIGN/QUEUE | Assigns queues to devices |
| DEASSIGN/QUEUE | Deassigns queues from devices |
| DELETE/QUEUE | Deletes queues |
| INITIALIZE/QUEUE | Creates queues |
| SET DEVICE/NOSPOOLED | Turns off spooling of printers or terminals |
| SET DEVICE/SPOOLED | Establishes spooled printers or terminals, and assigns queues to them |
| START/QUEUE | Starts queues |
| STOP/ABORT | Aborts printing of files currently being printed |
| STOP/QUEUE | Stops queues |
| STOP/REQUEUE | Stops the printing of jobs currently being printed and requeues them |

## 8.1 SPOOLING

Spooling is the technique of using secondary storage to buffer data passing between slow I/O devices (such as line printers and card readers) and physical memory. The slow devices, which can be either the ultimate sources or the ultimate destinations of buffered I/O data, are called spooled devices; the secondary storage devices are called intermediate devices.

As a rule, programs demand input and produce output at irregular intervals during their execution. If programs were allowed to read directly from slow devices and to write directly to slow devices, the execute time of programs would be limited by the speed of the slow devices. If a process output data directly to a printer, the process would be tied up for the time it took to print the listing. Also, other processes needing the printer would have to wait.

To balance these input/output demands and enhance throughput, you can establish spooled devices; to all other users, and their programs, the mechanism of spooling is transparent.

Input spooling makes input from a spooled device (such as a card reader) available for processing by placing it into a file on an intermediate 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 makes output from the processor available for transmission to a spooled device (such as a line printer) by placing it into files on an intermediate 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.

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.

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 is 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: 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.

### 8.1.1 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 used to queue spooled jobs. Thus, no special command is needed to establish card readers as spooled devices.

However, you must use the DCL command SET DEVICE 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. The VAX/VMS Command Language User's Guide describes the SET DEVICE command in detail.

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

At a minimum, you 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. 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), you can define still other devices as spooled devices without having to reboot the system. Normally, all line printers should be spooled.

You can use the SET DEVICE command to specify the intermediate device for each output spooled device. When you select an intermediate device, ensure that it has sufficient free space for the volume of queued output you expect. If you plan to enforce disk quotas on the intermediate device, ensure that all expected users of the spooled device have a quota authorized on the intermediate device.

Finally, and most important, on a system with both spooled input devices and spooled output devices, you 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.

### 8.1.2 Turning Off Spooling

You can, as necessary, turn off spooling to spooled printers and terminals by use of the DCL command SET DEVICE. However, you must stop the corresponding device queues before you can change the spooling status.

## 8.2 BATCH JOBS

Batch jobs can be submitted to the VAX/VMS system and queued for execution in two ways:

- 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. By default, the name of this batch queue is SYS$BATCH.

- 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.

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, SYS$BATCH is the batch queue.

In the VAX/VMS system, many jobs, or streams, can be executed at the same time from each of several batch queues. Thus, you 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.

In a batch queue that has been started, the job with the highest priority is the first candidate for execution. 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. You specify this limit with either of the DCL commands INITIALIZE/QUEUE or START/QUEUE.

- 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.

### 8.2.1 Creating Batch Queues

The DCL command INITIALIZE/QUEUE is used to create or initialize a batch queue. The use of this command is restricted to users with the OPER privilege. The VAX/VMS Command Language User's Guide describes the INITIALIZE/QUEUE command in detail.

Typically, you 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, you create batch queues by making entries in the system start-up command procedure. Note that you can define no more than 64 queues.

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

### 8.2.2 Starting Batch Queues

The execution of batch jobs from a batch queue (dequeuing) can only take place if the batch queue has been started. The DCL command START/QUEUE starts a batch queue. The use of this command is restricted to users with the OPER privilege. The VAX/VMS Command Language User's Guide describes the START/QUEUE command in detail.

Typically, you must see that batch queues created by use of the INITIALIZE/QUEUE command are started. Often, you start 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, you can start queues as operational needs dictate.

### 8.2.3 Stopping Batch Queues

You 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 DCL command STOP/QUEUE is used to stop batch queues. The VAX/VMS Command Language User's Guide describes this command in detail.

### 8.2.4 Deleting Batch Queues

You can delete batch queues, as necessary, by use of the DCL command DELETE/QUEUE. The VAX/VMS Command Language User's Guide describes this command in detail.

### 8.2.5 Emptying the Queue File

If the queue file (SYS$SYSTEM:JBCSYSQUE.EXE) becomes corrupted, it will be necessary to restart the system to create an empty queue file. You should suspect the queue is corrupted whenever you notice irregularities in the display produced by the DCL command SHOW QUEUE, when jobs are not being run, and/or jobs seem to disappear. The system parameter REINITQUE is provided for this purpose. First you would execute the following commands:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN>  USE CURRENT
SYSGEN>  SET REINITQUE 1
SYSGEN>  WRITE CURRENT
%OPCOM, 25-JUN-1982 16:14:09.41, message from user SYSTEM
%SYSGEN-I-WRITECUR, CURRENT system parameters modified by process ID . . .
SYSGEN>  EXIT
```

Next you should shut down the system, using the procedure SHUTDOWN.COM (see Chapter 7). Reboot, following the procedure defined in the software installation guide for your VAX-11 processor. The job controller creates a new empty queue file. If you plan to submit a Software Performance Report regarding your corrupted file (see Chapter 9), you should save the old JBCSYSQUE.EXE file and submit it with your report. (To save space you may choose to purge the file JBCSYSQUE.EXE.)

Next, you need to reset the reinitialize queue system parameter so that your queue file will not be emptied by default the next time you reboot. You can use the following sequence of commands:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN>  USE CURRENT
SYSGEN>  SET REINITQUE 0
SYSGEN>  WRITE CURRENT
%OPCOM, 25-JUN-1982 16:14:09.41, message from user SYSTEM
%SYSGEN-I-WRITECUR, CURRENT system parameters modified by process ID . . .
SYSGEN>  EXIT
```

### 8.2.6 Batch Versus Interactive Jobs

You should normally encourage users to submit large jobs (such as compiling and linking large programs) as batch jobs and reserve interactive use of the system for jobs that do not require extensive resources. A technique toward this end is to (1) restrict the working set size of interactive jobs by providing a small (for example, 200) WSQUOTA value in the UAF records and (2) expand the working set size of batch jobs by providing a large (for example, 500) WSQUOTA value in the START/QUEUE and INITIALIZE/QUEUE commands. You can likewise restrict and expand time limits on jobs by setting the CPU values.

### 8.2.7 Setting Up Batch Queues

The following guidelines 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 -- 1 to 4

   b. Priority -- 3 or 4

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

   d. Working set default -- 120

   e. Working set quota -- 200 to 500

   f. Working set extent -- 400 and up

   g. CPU default -- INFINITE

   h. CPU maximum -- INFINITE

You execute the following command procedure to create and start this queue:

```
$ START/QUEUE SYS$BATCH
$ IF $STATUS THEN GO TO BATCH_DONE
$ INITIALIZE/QUEUE/BATCH/JOB_LIMIT=1/PRIORITY=3/WSDEFAULT=300-
  /WSQUOTA=500/WSEXTENT=2000/CPUDEFAULT=INFINITE-
  /CPUMAXIMUM=INFINITE  SYS$BATCH
$ START/QUEUE SYS$BATCH
$ BATCH_DONE:
```

The first START command and the test on $STATUS ensure that an existing queue is not initialized.

Normally, these commands are contained in the start-up command procedure (see Chapter 7).

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. You can adjust the system workload by stopping and restarting background queues as needed.

2. Give SYS$BATCH the following characteristics:

   a. Job limit -- 6 to 10

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 -- 4 (by default)

c. Swapping mode -- swapping disabled

4. Give SLOW the following characteristics:

a. Job limit -- 1 (by default)

b. Priority -- low (3, for example)

c. Swapping mode -- swapping disabled

NOTE

This configuration should not be attempted on a small system (under 1MB). Additionally, you should add up the pages required for the batch working sets and insure that enough fluid memory[1] remains for interactive jobs. Otherwise, you must reduce the number of batch jobs or make the FAST and SLOW jobs swappable. In particular, you must not let fluid memory drop below the value of the WSMAX system parameter, or a deadlock could result.

Execute the following command procedure to create and start these queues:

```
$ START/QUEUE SYS$BATCH
$ IF $STATUS THEN GOTO BATCH_DONE
$ INITIALIZE/QUEUE/BATCH/JOB_LIMIT=6  SYS$BATCH
$ START/QUEUE SYS$BATCH
$ BATCH_DONE:
$ START/QUEUE FAST
$ IF $STATUS THEN GOTO FAST_DONE
$ INITIALIZE/QUEUE/BATCH/DISABLE_SWAPPING  FAST
$ START/QUEUE FAST
$ FAST_DONE:
$ START/QUEUE SLOW
$ IF $STATUS THEN GOTO SLOW_DONE
$ INITIALIZE/QUEUE/BATCH/PRIORITY=3/DISABLE_SWAPPING  SLOW
$ START/QUEUE SLOW
$ SLOW_DONE:
```

---

1. Fluid memory refers to memory that can be reassigned from one process to another through swapping and paging. You can calculate fluid memory as the space that remains when you subtract the number of pages permanently allocated to VAX/VMS from the total memory. You can obtain these values by issuing the DCL command SHOW MEMORY.

Normally, these commands should be contained in the site-specific start-up command procedure.

## 8.3 PRINT QUEUES

A queue is a list containing jobs that are waiting to be executed. Jobs are executed according to priority.

Print jobs are placed in print queues by means of the PRINT command. Print queues can be any one of the following:

- Physical-device queues -- queues associated with (that is, named for) a specific print device.
- Generic queues -- queues from which jobs can be given to any available print device that has matching characteristics.
- Named, or logical, queues -- queues that are not associated with a print device. To obtain printed output from a logical queue, you must explicitly assign the queue to a print device. The ASSIGN/QUEUE command, described in the VAX/VMS Command Language User's Guide, is used for this purpose.

Terminal queues are print queues destined for terminal devices (which are probably being used as remote printers, never interactively). They are created and controlled by use of the same commands that are used to create and control regular print queues. Print jobs in generic print queues are not dequeued to a terminal device queue unless the generic queue is initialized or started with the /TERMINAL qualifier.

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

Print jobs are queued for processing 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).

An implicitly spooled file is created when a program or DCL command sends its output to a spooled printer. 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. 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 system

logical 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.

The maximum number of physical device queues that can be printing at one time is restricted to the value of the MAXPRINTSYMB system parameter. See Chapter 10 for further information on system parameters.

### 8.3.1 Creating Print Queues

The DCL command INITIALIZE/QUEUE is used to create or initialize a print queue. The use of this command is restricted to users with the OPER privilege. The VAX/VMS Command Language User's Guide describes the INITIALIZE/QUEUE command in detail.

Typically, you must decide on the number of print queues for an installation and on their attributes. Often, you create print queues by making entries in the site-specific start-up command procedure. Note that you can define no more than 64 queues.

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

### 8.3.2 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 DCL command START/QUEUE starts a print queue. The use of this command is restricted to users who have the OPER privilege. The VAX/VMS Command Language User'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, you must see that print queues created with the INITIALIZE/QUEUE command are started. Often, you start print queues by making entries in the site-specific start-up command procedure.

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

### 8.3.3 Stopping Print Queues

You 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, with the START/QUEUE command. The STOP/QUEUE command is used to stop print queues and to suspend printing of jobs. The VAX/VMS Command Language User's Guide describes these commands in detail.

### 8.3.4 Deleting Print Queues

You can delete print queues, as necessary, by use of the DCL command DELETE/QUEUE. The VAX/VMS Command Language User's Guide describes this command in detail.

### 8.3.5 Emptying the Queue File

If the queue file becomes corrupted, you can use the same technique given in Section 8.2.5.

### 8.3.6 Assigning a Named, or Logical, Print Queue to a Printer

The DCL command ASSIGN/QUEUE is used to assign or redirect a named, or logical, print queue to a printer. The use of this command is restricted to authorized users with the OPER privilege. The VAX/VMS Command Language User'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, you can assign the queue to a line printer and start the queue.

### 8.3.7 Deassigning a Named, or Logical, Print Queue from a Printer

The DCL command DEASSIGN/QUEUE is used to deassign a named, or logical, print queue from a printer. The use of this command is restricted to users with the OPER privilege. The VAX/VMS Command Language User's Guide describes the DEASSIGN/QUEUE command in detail.

### 8.3.8 Adjusting Vertical Page Size

By default, the various system utilities (including the EDT editor, the compilers, and the linker) produce listings with a vertical page size of 66 lines. (The page size is defined as the number of lines between perforations.) You may change the vertical page size for listings with one of the following techniques, depending on whether the program was run in native mode or compatibility mode.

### 8.3.9 Page Length of Native Mode Image Listings

You may change the vertical page size for listings produced by the native mode utilities by specifying a numeric value, an integer in the range of 30 to 99, inclusive, for the system logical name SYS$LP_LINES. The following example changes the vertical page size for all users to 60 lines per page:

```
$ DEFINE /SYSTEM SYS$LP_LINES 60
```

Individual users may change the vertical page size on a group or process basis.

A vertical page size of less than that specified by the DCL command SET PRINTER/PAGE (which defaults to 64) causes a skip to the head of the next form each time the specified line count is reached. A greater vertical page size causes the excess lines to overflow to the next form and then skip to the head of the following form when the count is reached.

### 8.3.10 Page Length of Compatibility Mode Image Listings

The default page length of 66 applies to listings created by the following compatibility mode programs:

- MAC.EXE (MACRO/RSX11)
- TKB.EXE (LINK/RSX11)
- CRF.EXE (Compatibility Mode Cross Reference)
- SOS.EXE (EDIT/SOS)
- SLP.EXE (EDIT/SLP)

To modify this default page length, use the command procedure LINEPAGE.COM in SYS$UPDATE. Users with a system UIC or read/write access to system files can execute this command procedure.

Perform the following steps to change the default page length:

1. Change the current default with the following command:

```
$ SET DEFAULT SYS$UPDATE
```

2. Execute the procedure, located in this directory, specifying the count of lines per page as a parameter:

```
@LINEPAGE line-count
```

   For example, to set the default line count per page to 54, execute the procedure as shown below:

```
$ @LINEPAGE 54
```

NOTE

> The command procedure produces a new copy of the images being patched. If the image had been previously installed, remember to reinstall it (with the Install Utility) after patching, or else you must reboot the system. You can delete the old versions of the image files; if you need the standard 66 lines per printer page format, just run the command procedure again, giving the line-count as 66. (The procedure can be run multiple times.)

### 8.3.11 Forms Control

You specify the forms type of a print queue with the /FORMS_TYPE qualifier to the INITIALIZE/QUEUE or START/QUEUE command. If a user enters a print job with a forms value (/FORMS qualifier to the PRINT command) different from that of the queue, the job is placed on a hold status until the forms value of the queue is set equal to the forms value of the job. (You should stop the queue, physically change the forms, and start the queue specifying the new value for the /FORMS_TYPE qualifier.)

The forms type can be specified as a number or an alphabetic code. Alphabetic codes must be defined in the file SYS$MANAGER:FORMSTYPE.DAT, one code per line, in the following format:

```
% code number comments
```

You can include lines of text in the file if desired. Only lines beginning with a percent sign are taken as code-number definitions. The following example defines the code NORMAL as the number 0:

```
% NORMAL 0 NORMAL LINE PRINTER PAPER
```

A specification of /FORMS=NORMAL (or /FORMS=N, /FORMS=NO, and so on) is interpreted to mean /FORMS=0. Note, however, that the first match found in FORMSTYPE.DAT prevails with no ambiguity checks made, so that potentially conflicting names (that is, names starting with the same letter) should be avoided.

### 8.3.12 Printer Characteristics

You specify the printer characteristics of a print queue with the /CHARACTERISTICS qualifier to the INITIALIZE/QUEUE or START/QUEUE command. If a user enters a print job with a characteristic (/CHARACTERISTICS qualifier to the PRINT command) not included in those for the queue, the job is placed on a hold status until the characteristics of the queue are set to include all the characteristics of the job. (You should stop the queue, physically change the characteristics of the printer, and start the queue specifying the new values for the /CHARACTERISTICS qualifier.)

A characteristic can be specified as a number or an alphabetic code. Alphabetic codes must be defined in the file SYS$MANAGER:CHARTYPE.DAT, one code per line, in the following format:

```
% code number comments
```

You can include lines of text in the file if desired. Only lines beginning with a percent sign are taken as code-number definitions. The following example defines the code REDINK as the number 3:

```
% REDINK 3
```

Subsequent INITIALIZE/QUEUE or START/QUEUE and PRINT commands can use the number 3 or the alphabetic code REDINK to describe one printer characteristic. Note, however, that the first match found in CHARTYPE.DAT prevails with no ambiquity checks made, so that potentially conflicting names (that is, names starting with the same letter) should be avoided.

### 8.3.13 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 are in remote locations, that use special forms, or that possess unique printer characteristics should not be enabled for generic printing.

4. To use special forms or apply unique printer characteristics to a general-purpose queue, stop the queue, physically change the forms or apply the printer characteristics, and start the queue with the appropriate /FORMS_TYPE or /CHARACTERISTICS qualifier. After the special jobs are printed, stop the queue, physically reset the forms or printer characteristics, and start the queue with the original /FORMS or /CHARACTERISTICS value.

Figures 8-1 through 8-4 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.

NOTE

The commands shown in the examples assume manual entry at run time. Command procedures -- especially start-up command procedures -- containing INITIALIZE and START commands should contain logic to ensure that an existing queue is not initialized. See Chapter 7 and the examples in Section 8.2.6. If an existing queue is initialized, any jobs in that queue are lost.

Figure 8-1 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.

Figure 8-2 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 with the PRINT command to one of the two printers are placed in the queue associated with the specified printer.

6. Print jobs queued with 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 LPA0
$ ASSIGN/SYSTEM LPA0  SYS$PRINT
$ INITIALIZE/QUEUE/FLAG  LPA0
$ START/QUEUE  LPA0
```

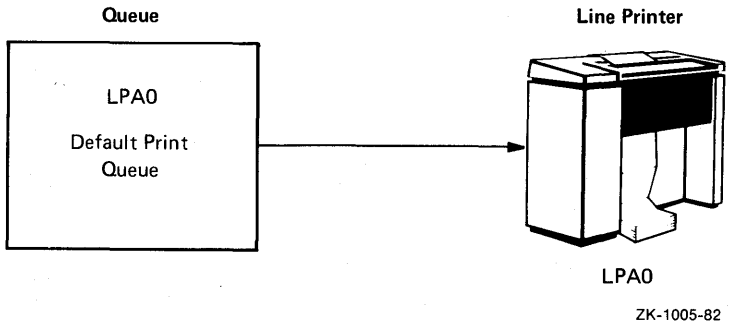



**Figure 8-1: Setting Up a Spooled Printer and a Print Queue on a System with One Line Printer**

Figure 8-3 illustrates a typical spooling and queuing configuration for a system with three line printers: two that have the same characteristics and one that uses special forms, has unique printer characteristics, or is in a remote location. The configuration shown in Figure 8-3 is basically the same as the one in Figure 8-2, with the addition of the spooled printer LPC0 and the creation and starting of the queue LPC0. Because of the special forms, unique characteristics, or the remote location, printer LPC0 is not suited for general printing. Thus, with the following exceptions, the commands listed in Figure 8-3 produce the same results as the commands listed in Figure 8-2:

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 SYS$PRINT.

**COMMANDS**

```
$ SET DEVICE/SPOOLED=SYS$PRINT LPA0
$ SET DEVICE/SPOOLED=SYS$PRINT 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
```

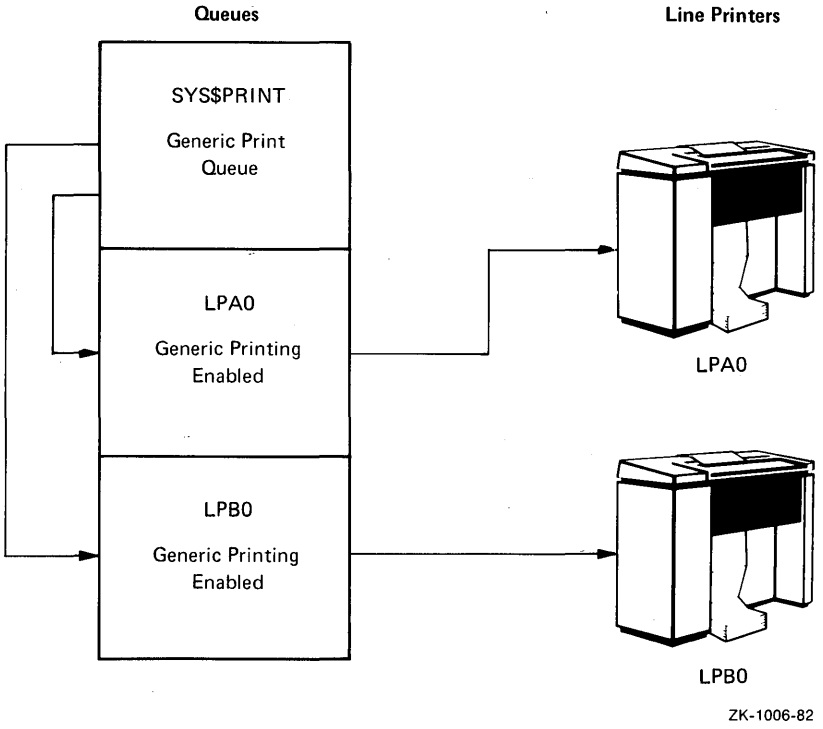



**Figure 8-2: Setting Up Spooled Printers and Print Queues on a System with Two Line Printers with the Same Characteristics**

Figure 8-4 adds still another feature to the configuration illustrated in Figure 8-3. 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).

**COMMANDS**

```
$ SET DEVICE/SPOOLED=SYS$PRINT LPA0
$ SET DEVICE/SPOOLED=SYS$PRINT 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
```

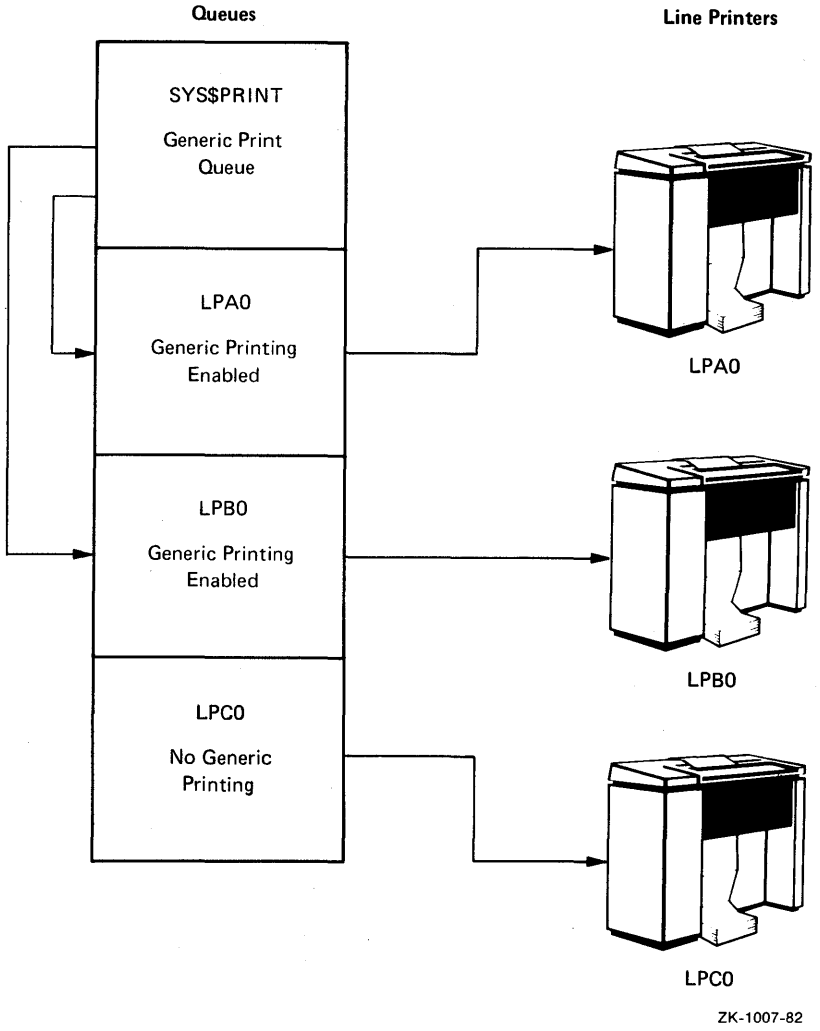



**Figure 8-3: Setting Up Spooled Printers and Print Queues on a System with Three Line Printers; Two with the Same Characteristics and One with Special Characteristics or in a Remote Location**

As shown in Figure 8-4, the DCL command INITIALIZE/QUEUE 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=SYS$PRINT LPA0
$ SET DEVICE/SPOOLED=SYS$PRINT 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
```

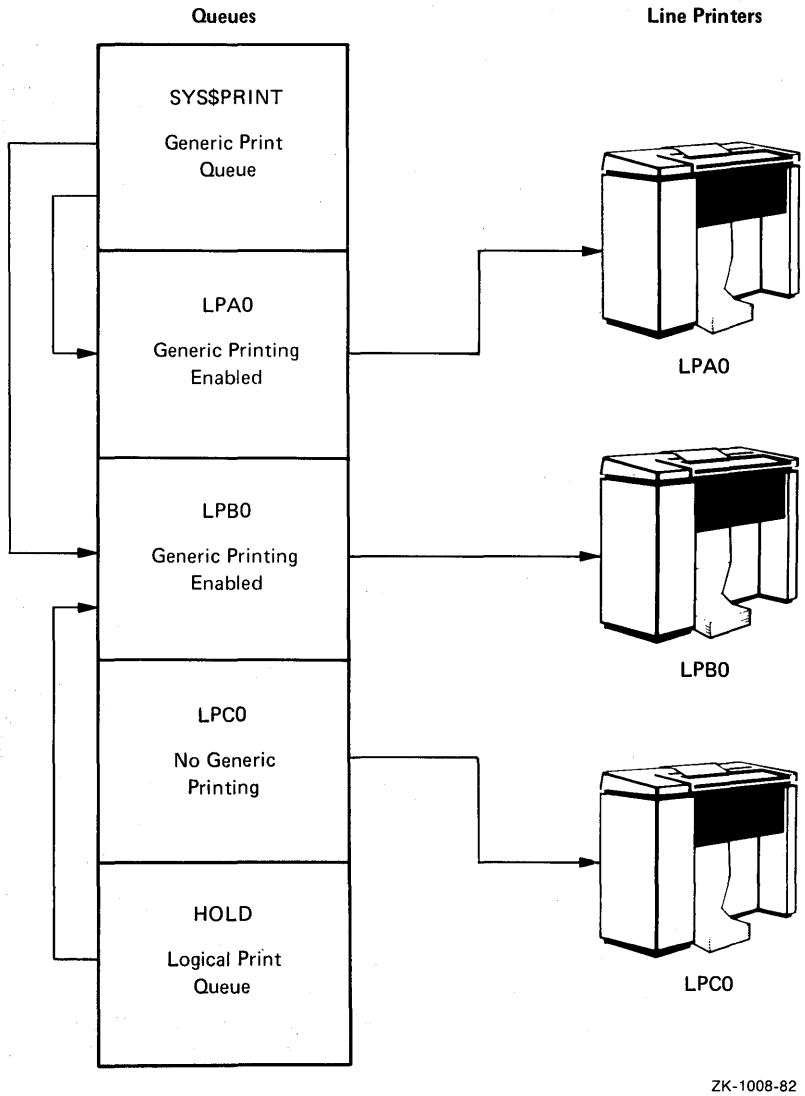



**Figure 8-4: Setting Up Spooled Printers and Print Queues -- Adding a Logical Queue to the System with Three Line Printers**

## 8.4 COMMANDS FOR CONTROLLING PRINT AND BATCH QUEUES

The commands listed in Table 8-2 allow you to manipulate queues and the jobs that they contain. These commands are described in the VAX/VMS Command Language User's Guide.

**Table 8-2: DCL Commands for Controlling Print and Batch Queues**

| Command | Function |
|---|---|
| ASSIGN/MERGE | Removes all jobs from one queue and places them in another queue |
| ASSIGN/QUEUE | Assigns a device to a logical queue |
| DEASSIGN/QUEUE | Deassigns a device from a queue |
| DELETE/ENTRY[1,2] | Deletes an entry from a print or batch job queue or stops printing the current job |
| DELETE/QUEUE | Deletes print and batch queues |
| INITIALIZE/QUEUE | Creates print and batch queues |
| PRINT[1] | Queues one or more files for printing, either on a default system printer or other device |
| SET DEVICE | Establishes the spooling and error-logging status on a device |
| SET QUEUE/ENTRY[1,2] | Changes the status or attributes of jobs in print or batch queues that have not yet been processed by the system |
| SHOW DEVICES[1] | Displays the status of devices in the system |
| SHOW QUEUE[1] | Displays the names, job identification numbers, and status of current and pending jobs in print or batch queues |
| START/QUEUE | Starts print or batch queues |
| STOP[2] | Halts execution of a command procedure, program, subprocess, or detached process |
| STOP/ABORT[1,2] | Stops printing of a job that is currently printing |

1. This command does not require the operator user privilege (OPER).

2. A user with either operator (OPER) or world (WORLD) privilege can use this command to affect any job in the system.

(continued on next page)

Table 8-2 (Cont.): DCL Commands for Controlling Print and Batch Queues

| Command | Function |
|---|---|
| STOP/ENTRY[1,2] | Stops execution of a batch job that is currently running and deletes it |
| STOP/QUEUE | Suspends print queues and stops batch queues |
| STOP/REQUEUE[1,2] | Stops printing of a job that is currently printing and requeues that job, giving it a priority of 1 |
| SUBMIT[1] | Enters a job in a batch queue |

1. This command does not require the operator user privilege (OPER).

2. A user with either operator (OPER) or world (WORLD) privilege can use this command to affect any job in the system.

## 8.5 PROCEDURES FOR CONTROLLING PRINT AND BATCH QUEUES

The following sections contain step-by-step procedures for controlling print and batch queues established on the VAX/VMS operating system.

### 8.5.1 Merging Print Queues

When a problem occurs with a print device, the queue associated with that print device should be rerouted to another print device. The procedure below describes how to merge two print queues without losing jobs in either queue.

**Procedure**

1. Stop the queue associated with the malfunctioning print device by issuing the following command:

   ```
   STOP/QUEUE/NEXT queue-name1
   ```

   This command inhibits further dequeuing but permits the current job to be completed. However, if the print device is inoperable, the current job will not be completed.

2. Requeue the current job by typing the following command:

   ```
   STOP/REQUEUE queue-name1
   ```

   By requeuing the current job, this command ensures this job will be printed in its entirety. Other jobs in the queue will not be dequeued because the queue is stopped.

3. Take the device offline.

4. Reroute the jobs queued to the malfunctioning print device to another print device by entering the following command:

   ASSIGN/MERGE queue-name2 queue-name1

   You should check that the characteristics of the new print device are appropriate for the new jobs.

5. Optionally, delete the queue associated with the malfunctioning print device by typing:

   DELETE/QUEUE queue-name

**Example**

1. 
```
$ STOP/QUEUE/NEXT   LPB0
$ STOP/REQUEUE   LPB0
$ ASSIGN/MERGE LPA0   LPB0
$ DELETE/QUEUE   LPB0
```

   The STOP/QUEUE/NEXT command prevents further dequeuing from the LPB0 queue. The STOP/REQUEUE command terminates the job currently being printed or attempting to print and places it back in the queue with a priority of 1. The next job in the LPB0 queue will not be dequeued because the queue has been stopped. The ASSIGN/MERGE command removes the jobs from the print queue LPB0 and places them in the print queue LPA0. The print queue LPB0 is then deleted by means of the DELETE/QUEUE command.

### 8.5.2 Preventing Loss of Data When the Line Printer Runs Out of Paper

The procedure below describes how to prevent loss of data while paper is loaded in the line printer.

When a line printer runs out of paper, OPCOM prints the following form of message on the operator's terminal to indicate that the device is not ready:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, Device device-name is offline.
```

**Procedure**

1. Suspend the current queue operation by issuing the following DCL command:

   STOP/QUEUE queue-name

   The DCL command SHOW QUEUE queue-name will now show the queue as PAUSED.

2. Take the printer offline.

3. Load a new box of paper in the printer and return the printer online.

4. Resume printing by entering the DCL command:

   START/QUEUE/optional-qualifier queue-name

Note that you can optionally append one of the following qualifiers to the above command to insure the output of the interrupted print job is complete:

| | |
|---|---|
| /BACKSPACE | Backspaces one page before printing resumes. |
| /TOP_OF_FILE | Starts at the beginning of the file that was being printed when the paper ran out. |

**Example**

```
%OPCOM, 19-JUN-1982 22:08:43.40, Device LPA0 is offline.

$ STOP/QUEUE  LPA0
$ START/QUEUE/TOP_OF_FILE  LPA0
```

OPCOM notifies the operator that line printer LPA0 went offline at 22:08:43.40. The operator stops the queue associated with the printer and takes the printer offline. After loading a new box of paper in the printer, the operator returns the printer online. Printing resumes after the operator types the START/QUEUE command. The /TOP_OF_FILE qualifier indicates that the job that was being printed when the operator issued the STOP/QUEUE command will be restarted at the beginning of the file.

### 8.5.3 Terminating the Execution of a Batch Job

The procedure below describes how to terminate the execution of a batch job. You usually perform this task only when careful assessment shows a job must be terminated or the owner of the job requests the termination.

**Procedure**

1. Type the following command to determine the job number of the batch job and the name of the queue in which the job is located:

   ```
   $ SHOW QUEUE/BATCH/ALL
   ```

2. Delete the batch job by issuing one of the following commands:

   ```
   STOP/ENTRY=job-number    queue-name
   DELETE/ENTRY=job-number  queue-name
   ```

**Example**

```
$ SHOW QUEUE/BATCH/ALL

 * Batch queue "SYS$BATCH" Joblim=5, Basepri=4, Swap

  CURR   1662 DEBUG        DBGBUILD   Pri=4   7-JUN-1982 12:25
  AFTER  1710 JEROME       BEGINBLD   Pri=4   7-JUN-1982 18:49
  AFTER  1703 SYSTEM       DELETELO   Pri=4   7-JUN-1982 22:23
  CURR   1708 SYSTEM       LNK32      Pri=4   7-JUN-1982 12:20
  CURR   1706 LANGLEY      FORPROG    Pri=4   7-JUN-1982 12:18
```

```
 * Batch queue "SYS$BUILD" Joblim=7, Basepri=4, Swap

 * Batch queue "SYS$CHECKIN" Joblim=1, Basepri=4, Swap Stopped

 * Batch queue "SYS$LOAD" Joblim=100, Basepri=4, Swap Stopped

$ STOP/ENTRY=1706  SYS$BATCH
```

Assume a user has observed that job 1706 is a runaway job. The user is not the owner of the job, and lacks sufficient privilege to stop it. The user enlists the operator's aid. The operator types the SHOW QUEUE/BATCH/ALL command to determine the queue in which the job has been entered. The display shows that job 1706 is in the SYS$BATCH queue. The operator then deletes the job by typing the STOP/ENTRY command.

### 8.5.4 Terminating the Printing of a Print Job

The procedure below describes how to terminate a print job that is currently being printed on a print device. You usually perform this task only if the system manager or the owner of a job requests that the job be terminated, or if you observe garbled output on the print device.

**Procedure**

Enter the following command:

```
STOP/ABORT print-device
```

This command terminates the current job and begins printing the next job in the queue.

**Example**

```
$ STOP/ABORT   LPA0
```

This command terminates the printing of the current print job on LPA0. The next job in the queue is immediately dequeued for printing.

### 8.5.5 Removing a Batch or Print Job from a Queue

The procedure below describes how to delete a batch or print job that has been entered in a queue but has not yet been processed. You usually perform this task only after careful assessment that it is necessary or at the request of the owner of the job.

**Procedure**

1. Type the following command to determine the job number and the name of the queue in which the job is located:

   ```
   $ SHOW QUEUE/DEVICE/ALL
   ```

2. Delete the job by issuing the following command:

   ```
   DELETE/ENTRY=job-number queue-name
   ```

**Example**

```
$ SHOW QUEUE/DEVICE/ALL

 * Generic Device queue "SYS$PRINT"  Flag

 * Device queue "LPA0" Forms=0, Genprt Lower Flag

  CURR  1327 RAP          ASHLEY    Pri=4  7-JUN-1982 12:36 Size=73

 * Terminal queue "TTF4" Forms=0, Nogen Lower

 * Device queue "LQP" Forms=0, Nogen

          This queue assigned to TTF4

 * Device queue "LPC0" Forms=0, Nogen Lower Flag

  CURR  1328 SMITH        OEUVRE    Pri=4  7-JUN-1982 12:54 Size=312

 * Terminal queue "TTH0" Forms=0, Nogen Lower

 * Device queue "PLOTQ" Forms=0, Nogen

          This queue assigned to TTH0

$ DELETE/ENTRY=1328 LPC0
```

A user has requested that job 1328 be deleted. The operator types the SHOW QUEUE/DEVICES/ALL command to determine in which queue the job has been entered. The display shows that job 1328 is in the LPC0 queue. The operator then aborts the job by typing the DELETE/ENTRY command and by specifying the job-number 1328.

### 8.5.6 Restarting the Job Controller

If it becomes necessary to restart the job controller, you can invoke the STARTUP.COM procedure as follows:

```
$ @SYS$SYSTEM:STARTUP JOBCTL
```

## 8.6 USING THE CARD READER

The card reader is used to read card decks. Users may submit the two following types of card decks for processing:

- Batch job card decks
- Data card decks

You must understand the two types of card decks and how to tend the card reader in order to use the card reader and ensure card decks are processed efficiently. This chapter describes which cards you should check before processing a card deck through the card reader and how to determine which cards are damaged. Section 8.6.3.2 outlines a procedure for processing a card deck through the card reader.

### 8.6.1 Types Of Card Decks

Before loading a card deck into the card reader, you should determine:

- Whether the deck is a batch job or a data deck, because their processing requirements differ.
- Whether the card reader is set to the correct translation mode.

**8.6.1.1 Batch Job Card Deck** - A batch job card deck can be divided into three segments: the initial cards, the program cards or the cards containing the batch job, and the last card. The initial two cards in a batch job card deck are the $JOB and the $PASSWORD cards. These cards log in the user and the batch job to the system. Following the initial two cards are program cards. Program cards contain instructions that direct the system to libraries, routines, and data needed to complete the batch job. The last card must be either an End-Of-Job command card ($EOJ) or an End-Of-File (EOF) card. Both of these cards tell the system this is the end of the job.

**8.6.1.1.1 Checking Batch Job Card Deck Input** - When a batch job is inserted into the card reader input hopper for processing, the first two cards in the card deck must be:

- A $JOB card
- A $PASSWORD card

The system cannot execute the job without these cards. If you are given a card deck with these cards omitted, you should return the deck so the user can insert these cards.

Since the card deck contains the user's password, you must ensure it is always handled with care to preserve the security of the user's account.

The last card in the deck must be one of the following:

- A $EOJ card
- An EOF card (12-11-0-1-6-7-8-9 overpunch)

If the last card is not one of these end cards, you can type one on the card punch and insert it at the end of the deck.

**8.6.1.1.2 Checking Batch Job Output** - The log file produced by a card reader batch job is queued for printing to the default system print queue, SYS$PRINT. To have the log file queued to a different queue, the user can include an $ASSIGN or $DEFINE card in the job to redefine SYS$PRINT. The VAX/VMS Command Language User's Guide explains how to use the ASSIGN and DEFINE commands.

If an error occurs while the system is attempting to validate the $JOB and $PASSWORD cards, a listing containing the error message is queued to SYS$PRINT. The user's name (if any) on the listing flag page is the user's name from the $JOB card. The job name is INPBATCH.ERR. When the user's name cannot be determined from the $JOB card, the deck is simply flushed through the card reader and no listing is queued.

**8.6.1.2 A Data Card Deck** - A data deck is a deck of cards containing data that either will be read by a program or copied to a file for later use. The process that will read the data deck usually is associated with an interactive user at a terminal or a batch job that is submitted by an interactive user. Since the user and process already are logged in to the system, the first card can contain any data the user wants to specify. However, the program must read the exact number of cards supplied, or the last card should be an EOF card (12-11-0-1-6-7-8-9 overpunch) to inform the program that this is the end of the data deck.

When a user wants a data deck to be read, you should ensure that the user has allocated the card reader. If the card reader is not allocated, the system tries to submit the deck as a batch job and subsequently just flushes the deck through the reader, rejecting the job.

If the program does not read the exact number of cards, as with the COPY command, the EOF card (12-11-0-1-6-7-8-9 overpunch) must be the last card in the deck, to inform the program that this is the end of the deck. Without this card, the program waits indefinitely for more cards and the system prints "card reader offline" messages on the operator's terminal. If the card deck lacks an EOF card, you can type an EOF card on a card punch and insert it at the end of the deck.

### 8.6.2 Card Reader Translation Modes

For the system to read input properly, the card reader must be set to the correct translation mode. The translation mode used must be the same as the translation mode of the card punch on which the cards were punched. VAX/VMS supports 026 and 029 card punches. (These translation modes are discussed in detail in the VAX/VMS I/O User's Guide and the VAX/VMS Guide to Using Command Procedures.)

One of the following two conditions must exist for you to be able to set the card reader to the correct translation mode:

- The first card in the deck must be the translation mode card
- You must know the mode in which the cards were punched

Without the above information, you cannot set the card reader to the correct translation mode.

To set the translation mode of the card reader for many decks of the same type, use the SET CARD_READER command. This command is fully described in the VAX/VMS Command Language User's Guide. By default, when the system is bootstrapped, the translation mode is set to 029.

### 8.6.3 Tending The Card Reader

The job of tending the card reader includes:

- Ensuring that the cards in batch jobs and data decks are properly ordered as discussed in the preceding pages
- Replacing physically defective cards
- Operating the card reader

NOTE

For more information on card reader batch jobs from a system user's viewpoint, refer to the VAX/VMS Guide to Using Command Procedures.

**8.6.3.1 Replacing Physically Defective Cards** - Even when the card deck contains all the required cards, the card reader may not be able to read the deck. This usually occurs because one or more cards are physically defective.

If the deck contains a faulty card, one of the error indicators located on the front panel of the card reader lights up when the card is read. The card reader goes offline, and operator intervention is required to put it back online. Table 8-3 at the end of this chapter describes the error indicators, reasons why they may light up, and the operator action required to correct the situation.

**8.6.3.2 Operating the Card Reader** - This section contains a step-by-step procedure for processing card decks through the card reader.

**Procedure**

This procedure describes how to load and process a card deck through a card reader.

1. Remove the card weight from the input hopper. Place the cards, face down and with column 1 on the left, in the hopper. Ensure that the first card to be read is at the bottom of the hopper.

   Do not pack the input hopper so full that the air from the blower cannot riffle the cards. If the cards are packed too tightly, the vacuum picker cannot operate properly.

2. Press the RESET button. The HOPPER CHECK error indicator and the STOP light will go out and the cards will be read.

   If the card deck is too large to fit in the input hopper, the excess cards can be loaded while the reader is operating if tension is maintained on the front portion of the deck.

3. Remove the cards from the output stacker when the HOPPER CHECK error indicator and the STOP light are lit.

   If the STOP button is accidentally pressed while the card deck is being read, return the last card in the output hopper to the bottom of the input hopper and press the RESET button.

4. If the cards are not read properly after the RESET button has been pressed, refer to Table 8-3 below for recovery procedures.

Table 8-3: Card Reader Errors: Causes and Corrective Actions

| Error Indicator | Causes | Corrective Action |
|---|---|---|
| READ CHECK | Card edges torn<br>Punch in column 0 or 81 | Remove the faulty card from the output stacker, duplicate the card, place it in the input hopper, and press the RESET button<br>If READ CHECK occurs for all cards, the read logic of the card reader is malfunctioning |
| PICK CHECK | Damage to leading edge<br>Torn webs<br>Cards stapled together | Remove the card from the input hopper, duplicate the faulty card, place the card back in the input hopper, and press the RESET button<br>If there is no evidence of card damage, check for excessive warping of the card deck and/or a buildup of ink glaze on the picker face |
| STACK CHECK | Jam in the card track<br>Badly mutilated card | Correct the jam and/or remove the mutilated card from the output stacker, duplicate the card, place it in the input hopper, and press the RESET button |
| HOPPER CHECK | Input hopper empty<br>Output stacker full | Load the input hopper<br>Unload the output stacker |

# CHAPTER 9

# ERRORS AND OTHER SYSTEM EVENTS

The system provides several tools for recording and reporting errors and other system events. These tools include facilities for logging and reporting system events, logging operator messages, and reporting problems to DIGITAL. In the event of a severe system failure, VAX/VMS automatically shuts down the system and produces a crash dump of the state of the system at the time the error was detected. You can analyze the dump to help you determine the cause of the system failure. See the VAX/VMS System Dump Analyzer Reference Manual. In a few cases, as described in Section 9.3, you may want to forward the dump to DIGITAL with a Software Performance Report.

## 9.1 ERROR LOG

The system automatically writes messages to the latest version of a file named SYS$ERRORLOG:ERRLOG.SYS as the following events occur:

- Errors -- Device errors, machine checks, bus errors, soft error correcting code (ECC) errors, asynchronous write errors, hard ECC errors
- Configuration changes -- Volume mounts and dismounts
- System events -- Cold start-ups, warm start-ups, crash start-ups, messages from Send Message to Error Logger ($SNDERR) system service, time stamps

You can display and report the information in the error log by running the SYE Utility, which is decribed in the VAX-11 Utilities Reference Manual. Since the system continues to log messages to ERRLOG.SYS and creates a new version of the file if the current one is locked, you should either rename the file, copy it, or append it to another file, before running SYE against the new file. ERRLOG.OLD is recommended as the new name since SYE uses this name as a default for the input file.

### 9.1.1 Operations

The error logging facility consists of three 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
- The SYE Utility

The executive routines and the ERRFMT process 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 writes the buffers to ERRLOG.SYS. 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, but does not save the data. You can detect this condition by noting a skip in the sequence numbers of the records reported by SYE. As soon as ERRFMT frees the buffer space, the executive routines resume preserving error information in the buffers.

The ERRFMT process displays an error message on the system console and deletes itself if it encounters excessive errors while writing the error log file. You can restart ERRFMT by invoking the STARTUP command procedure in the [SYSEXE] directory and by specifying one parameter, as in the following example:

```
$ @SYS$SYSTEM:STARTUP  ERRFMT
```

The command procedure should be invoked from the system manager's account (UIC [1,4]).

### 9.1.2 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 the failure.

The detailed contents of the reports are most meaningful to DIGITAL field service personnel. However, you can 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, you 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 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, you 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 you 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 system downtime.

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, you can rename and copy to a disk file a version of the error log file that covers a crucial period of system activity. Upon arriving on site, your field service representative can generate one or more reports from the copied file as well as from the current version of ERRLOG.SYS.

### 9.1.3 Maintaining the Error Log Files

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. You can take steps to minimize or control the number of versions created.

For example, when generating several reports from the current error log file, you should first rename the error log file to ERRLOG.OLD 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, you ensure that SYE is accessing the same error log file for each report.

All versions of the ERRLOG.SYS file remain on the system disk (SYS$SYSROOT) until they are explicitly deleted. Therefore, you must devise a plan for regular maintenance of the error log file.

One way to do this is to rename the highest version of ERRLOG.SYS on a daily basis. This action 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, you could rename the current copy of ERRLOG.SYS every morning at 9:00 o'clock to ERRLOG.OLD. To free space on the system disk, you 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. You may also want to consider adopting a naming convention for your files that incorporates a beginning or ending date for the data in the file name. A detailed example of this maintenance procedure is provided in the next section.

### 9.1.4 Printing The Error Log File

The procedure below describes how to rename a formatted error log file and obtain a copy of it. Note that these instructions are for renaming and printing one version of the error log file at a time. For a complete description of the SYE Utility, refer to the VAX-11 Utilities Reference Manual.

**Procedure**

1. Set the default disk and the default directory by typing the following command:

   ```
   $ SET DEFAULT SYS$ERRORLOG
   ```

2. Examine the directory to see which versions of the ERRLOG.SYS file are on disk by typing:

   ```
   $ DIRECTORY ERRLOG.SYS
   ```

3. Rename all versions of the ERRLOG.SYS file, one at a time, to ERRLOG.OLD by issuing the command:

   ```
   RENAME/LOG ERRLOG.SYS;n    ERRLOG.OLD
   ```

   To preserve the chronological order of the files after renaming, first rename the oldest version of ERRLOG.SYS (the version with the lowest version number, n), then rename versions n+1, n+2 and so on.

   Do not use a wild card character in the version field of the file specification to rename more than one version of ERRLOG.SYS at a time. Such a RENAME command will attempt to preserve version numbers and thus chronological order, but the presence of previously renamed ERRLOG.OLD files will interfere with that algorithm.

4. Invoke the SYE Utility to format the error log file into a readable error log report with the command:

   ```
   $ RUN SYS$SYSTEM:SYE
   ```

   After the above command string is processed, the SYE Utility prompts for input and output file specifications, options, a device name, and entry dates.

5. Respond to the SYE prompt by specifying the name of the error log file and the type of output desired. Note that if you respond to all the SYE Utility prompts by pressing the RETURN key, the SYE Utility defaults to the specifications contained in the square brackets (as shown in the example below).

6. Obtain a printed copy of the error log report by entering an output file name in response to the SYE prompt for an output file as shown in the example below. Then enter the following command:

   ```
   $ PRINT file-name
   ```

   The file name indicates the name of the file containing the error log report.

**Example**

```
$ SET DEFAULT SYS$ERRORLOG
$ DIRECTORY ERRLOG.SYS

Directory SYS$SYSROOT:[SYSERR]

ERRLOG.SYS;1

Total of 1 file.
```

```
$ RENAME ERRLOG.SYS ERRLOG.OLD
$ RUN SYS$SYSTEM:SYE
SYE VERSION 3.3
_INPUT FILE:     [ [SYSERR]ERRLOG.OLD ]  ?RET
_OUTPUT FILE:    [SYS$OUTPUT]            ?ERRLOG.LISRET
_OPTIONS:        [ROLL-UP]               ?RET
_DEVICE NAME:    [<CR>]                  ?RET
_AFTER DATE:     [FIRST ENTRY]           ?RET
_BEFORE DATE:    [LAST ENTRY]            ?RET

%SYE-I-SUCCOM,  SUCCESSFUL COMPLETION
$ PRINT ERRLOG.LIS
```

The SET DEFAULT command sets the operator's default disk and directory to SYS$SYSROOT:[SYSERR]. The DIRECTORY command lists all the ERRLOG.SYS files contained in the [SYSERR] directory. In this example, [SYSERR] contains only one version of ERRLOG.SYS. The RENAME command renames ERRLOG.SYS to ERRLOG.OLD; a new version number is assigned if a file of this name already exists.

The operator then invokes the SYE Utility by typing RUN SYS$SYSTEM:SYE. The SYE Utility lists the defaults enclosed in square brackets for each of the following parameters and prompts for any changes in these:

- The name of the file to be manipulated (here, ERRLOG.OLD, the default input file).
- The name of the file that is to contain the error log report. The error log file in the preceding example is written to ERROLOG.LIS, which facilitates obtaining a printed copy of the error log file later. The default output file, SYS$OUTPUT, is printed at the operator's terminal.
- The type of report that SYE should generate. The type of report here is a summary ROLL-UP report. For a description of other types, see the VAX-11 Utilities Reference Manual.
- The devices for which SYE should report errors. By pressing the RETURN key, you request error reports for all devices.
- The time from which SYE should report errors. By pressing the RETURN key, you request that SYE report all errors that occurred since the error log file was created.
- The time until which SYE should report errors. By pressing the RETURN key, you request that SYE report the occurrence of errors up to and including the last error in the error log file.

The SYE Utility creates a readable error log report. The operator obtains a hardcopy of this report by pressing the RETURN key in response to the final SYE prompt, provided the operator previously had not specified an output file name. If the operator had chosen an output file name other than the default SYS$OUTPUT (as was done in the preceding example by responding ERRLOG.LIS to the OUTPUT FILE question), an additional step (use of the DCL command PRINT) would have been required to produce a printed copy of the report.

## 9.2 OPERATOR'S LOG FILE

The operator's log file (SYS$MANAGER:OPERATOR.LOG) is another system management tool that is useful in anticipating and preventing hardware and software failures. By regularly examining the operator's log file, you can often detect tendencies, or trends, toward failures and can thereby ensure that corrective action is taken before these failures occur. You should, therefore, print out copies of the operator's log file regularly, and retain these copies for reference. Figure 9-1 illustrates some typical messages found in the operator's log file.

```
%OPCOM, 29-JUN-1982 22:28:28.72, message from user NETACP
DECnet shutting down
%OPCOM, 29-JUN-1982 22:33:54.07, operator disabled, operator OPA0
%OPCOM, 29-JUN-1982 22:34:15.47, operator enabled, operator OPA0
%OPCOM, 29-JUN-1982 22:34:15.57, operator status for operator OPA0
PRINTER, TAPES, DISKS, DEVICES
%OPCOM, 29-JUN-1982 22:38:53.21, request 1, from user PUBLIC
Please mount volume KLATU in device _MTA0:
Gort, the tape is in cabinet A
%OPCOM, 29-JUN-1982 22:39:54.37, request 1 was satisfied.
%OPCOM, 29-JUN-1982 22:40:23.54, message from user SYSTEM
Volume "KLATU       " mounted, on physical device MTA0:
%OPCOM, 29-JUN-1982 22:40:38.02, request 2, from user PUBLIC
MOUNT new relative volume 2 () on MTA0:
%OPCOM, 29-JUN-1982 22:41:07.54, message from user SYSTEM
Volume "KLATU       " dismounted, on physical device MTA0:
%OPCOM, 29-JUN-1982 22:41:14.95, device LPA0 is offline
%OPCOM, 29-JUN-1982 22:41:50.98, message from user SYSTEM
CURRENT system parameters modified by process ID 001F003C into file SYS$SYSROOT:[SYSEXE]SYS.EXE;1
BERADA
29-JUN-1982 22:42:14.81, request 2 completed by operator OPA0
%OPCOM, 29-JUN-1982 22:42:15.83, request 3, from user PUBLIC
MOUNT new relative volume 3 () on MTA0:
%OPCOM, 29-JUN-1982 22:42:16.95, device LPA0 is offline
%OPCOM, 29-JUN-1982 22:42:44.54, message from user SYSTEM
Volume "BERADA      " mounted, on physical device MTA0:
%OPCOM, 29-JUN-1982 22:42:44.73, message from user SYSTEM
Volume "BERADA      " dismounted, on physical device MTA0:
I'm sorry, but we are out of tapes.
29-JUN-1982 22:45:11.45, request 3 aborted by operator OPA0
%OPCOM, 29-JUN-1982 22:46:47.96, request 4, from user PUBLIC
_TTB5:, This is a sample user request w/ reply expected.
%OPCOM, 29-JUN-1982 22:47:38.50, request 4 was canceled
%OPCOM, 29-JUN-1982 22:48:21.15, message from user PUBLIC
_TTB5:, This is a sample user request w/o a reply expected.
%OPCOM, 29-JUN-1982 22:49:07.90, Device DMA0 is offline.
        Mount verification in progress.
%OPCOM, 29-JUN-1982 22:49:20.22, Mount verification completed for device DMA0
%OPCOM, 29-JUN-1982 22:49:37.64, Device DMA0 has been write locked.
        Mount verification in progress.
%OPCOM, 29-JUN-1982 22:53:54.52, Device DMA1 is offline.
        Mount verification in progress.
%OPCOM, 29-JUN-1982 22:54:16.56, Mount verification aborted for device DMA1
```

**Figure 9-1: Sample Operator's Log File (SYS$MANAGER:OPERATOR.LOG)**

The operator's log file records the occurrence of system events. These messages are produced by the operator's communication process (OPCOM) and are preceded by the label %OPCOM. Section 9.2.4 explains the messages in detail.

### 9.2.1 Maintaining The Operator's Log File

The operator's log file (SYS$MANAGER:OPERATOR.LOG) resides on the system disk in the [SYSMGR] directory. This file is in ASCII format and can be printed as readable text. You should print copies of the operator's log file regularly, and retain these copies for reference. Section 9.2.2 describes how to print copies of the operator's log file.

A new version of the operator's log file is created each time the system is rebootstrapped. The highest version of this file is always the one in use and is inaccessible. You should devise a plan for regular maintenance of these files.

One way to maintain these files is to rename the second highest version of the operator's log file on a daily basis. For example, you might rename the current version of OPERATOR.LOG to OPERATOR.OLD every morning at 9:00. To free space on the system disk, you then could back up the renamed version of the file on a different volume and delete the renamed file from the system disk. You should not delete versions of the operator's log file that have not been backed up.

The procedure for renaming the operator's log file is the same as that described in Section 9.1.4 for renaming the error log file.

### 9.2.2 Printing The Operator's Log File

The procedure below describes how to produce a printed copy of the current version of the operator's log file (OPERATOR.LOG). You should periodically print a copy of this file for review.

**Procedure**

1. Close the current log file and open a new one by entering the following command:

   ```
   $ REPLY/LOG
   ```

2. Set the default to the system disk by typing:

   ```
   $ SET DEFAULT SYS$MANAGER
   ```

3. Rename the second highest version of the operator's log to OPERATOR.OLD with the following command:

   ```
   $ RENAME OPERATOR.LOG;-1 OPERATOR.OLD
   ```

   The version number, -1 specifies that the second highest version of this file is to be printed. The highest version number is the current operator's log file.

4. Obtain a printed copy of the operator's log file by issuing the following command:

   ```
   $ PRINT OPERATOR.OLD
   ```

**Example**

```
$ REPLY/LOG
%OPCOM, 16-JUN-1982 12:29:24.52, logfile initialized by operator TTA2
        logfile is SYS$MANAGER:OPERATOR.LOG

$ SET DEFAULT SYS$MANAGER
$ DIRECTORY OPERATOR.LOG

Directory SYS$SYSROOT:[SYSMGR]

OPERATOR.LOG;582          OPERATOR.LOG;581

Total of 2 files.

$ RENAME OPERATOR.LOG;-1 OPERATOR.OLD
$ PRINT OPERATOR.OLD
```

The REPLY/LOG command closes the current log file and opens a new one; the response from OPCOM verifies that a new log file has been opened. The SET DEFAULT command sets the operator's default disk to the system disk, thus enabling the operator to examine the files contained in the directory [SYSMGR]. The operator renames the second highest version of the operator's log file to OPERATOR.OLD and then issues the PRINT command to request that this version of the operator's log file (OPERATOR.OLD) be printed.

### 9.2.3 Restarting OPCOM

You can restart OPCOM if for some reason it is deleted or suspended. Simply invoke the STARTUP command procedure in the [SYSEXE] directory and specify one parameter, as in the following example:

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

You should invoke the STARTUP command procedure from the system manager's account (UIC [1,4]).

### 9.2.4 Messages in the Operator's Log File

This section describes six of the seven types of message that appear in the operator's log file:

- Initialization of the operator's log file
- Status reports for devices attached to the system
- Terminals enabled and disabled
- Volume mounts and dismounts
- User requests and operator replies
- Changes to system parameters through the SYSGEN Utility
- DECnet-VAX status messages

See the DECnet-VAX System Manager's Guide for information about the seventh type, the DECnet-VAX status messages.

**9.2.4.1 Initialization Messages** - When you issue the REPLY/LOG command, the current operator's log file is closed and a new version of that file is created and opened. All subsequent OPCOM messages are recorded in this new log file.

When a new log file is created, the first message recorded in it is an initialization message that tells when and by whom the log file was initialized. This message appears in the following format:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, logfile initialized by operator operator-name
        logfile is SYS$MANAGER:OPERATOR.LOG
```

**9.2.4.2 Device Status Messages** - Some VAX/VMS I/O drivers send messages to OPCOM concerning changes in the status of the devices they control. For example, when a line printer goes offline, an OPCOM message is written into the operator's log file at periodic intervals until the device is explicitly returned to online status.

The device status message appears in the operator's log file in the following format:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, device device-name is offline
```

The devices for which this message can appear are card readers, line printers, and magnetic tapes.

**9.2.4.3 Terminal Enable and Disable Messages** - You designate a terminal as an operator's terminal by issuing the REPLY/ENABLE command from the desired terminal. OPCOM confirms the request by displaying the following message at the operator's terminal and in the operator's log file:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, operator enabled, operator terminal-name
```

This message tells you which terminal has been established as an operator's terminal and when it was so established.

If a terminal has been designated as an operator's terminal for a particular function, OPCOM displays the name of that function. For example, if you issue the command REPLY/ENABLE=TAPES, OPCOM displays the following message:

```
%OPCOM, 14-JUN-1982 10:25:35.74, operator enabled, operator TTE1
$
%OPCOM, 14-JUN-1982 10:25:38.82, operator status for operator TTE1
TAPES
```

OPCOM confirms that the terminal is established as an operator's terminal and indicates that the terminal can only receive and respond to requests concerning magnetic tape-oriented events, such as the mounting and dismounting of tapes.

A terminal that has been designated as an operator's terminal is automatically returned to nonoperator status when the operator logs out. To return the terminal to normal (nonoperator) status without logging off, you must issue the REPLY/DISABLE command from the terminal. OPCOM confirms that the terminal is no longer an operator's terminal by displaying a message in the following format both at the operator's terminal and in the operator's log file:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, operator disabled, operator terminal-name
```

This message tells you which terminal has been restored to nonoperator status and when the transition occurred.

If a terminal is designated as an operator's terminal and only partial operator status is disabled, OPCOM displays a status message. This message lists which requests the terminal can still receive and respond to. This message is displayed at the operator's terminal and in the operator's log file in the following format:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, operator status for operator terminal-name
status-report
```

For example, suppose you designate a terminal as an operator's terminal that receives messages concerning magnetic tapes and disks, as well as messages intended for the special site-specific operator class known as OPER10. Later, you relinquish the terminal's ability to receive messages concerning tapes. When you issue the REPLY/DISABLE=TAPES command, OPCOM returns the following message:

```
%Opcom, 14-JUN-1982 09:23:45.32, operator status for operator TTA3
 DISKS, OPER10
```

This message tells you that terminal TTA3 still receives and can respond to messages about disks and messages directed to OPER10.

**9.2.4.4 Volume Mount and Dismount Messages** - Perhaps the widest range of operator messages occur with volume mounts and dismounts. Chapter 5 discusses mount verification and operator-assisted mounts. An appropriate sample of the messages appears in those discussions and examples.

**9.2.4.5 User Request and Operator Reply Messages** - To communicate with you, the user issues the REQUEST command, specifying either the /REPLY or /TO qualifier.

If the user issues a REQUEST/REPLY command, the request is recorded in the operator's log file in the following format:

```
%OPCOM,dd-mmm-yyyy hh:mm:ss.cc, request request-id from user user-name
__terminal-name:, "message-text"
```

This message tells you which user sent the message, the time the message was sent, the request identification number assigned to the message, the originating terminal, and the message itself.

If the user issues a REQUEST/TO command, the request is recorded in the operator's log file in the format described above, but without a request identification number, as follows:

```
%OPCOM,dd-mmm-yyyy hh:mm:ss.cc, request from user user-name
__terminal-name:, "message-text"
```

For examples of the OPCOM messages that result from requests to mount magnetic tapes through the magtape ACP using the REQUEST/BLANK_TAPE and REQUEST/INITIALIZE_TAPE commands, see Chapter 5.

When you respond to a user's request and specify the /TO qualifier, the response is recorded in the operator's log file in the following format:

```
response message
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, request request-id completed by operator operator-name
```

This message indicates how the operator responded to the user's request, as well as when the response was issued and which operator responded.

When you respond to a user's request and specify the /ABORT qualifier, the response is recorded in the operator's log file in the following format:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, request request-id was cancelled.
```

When you respond to a user's request using the /PENDING qualifier, the response is not recorded in the operator's log file because the request has not yet been completed (that is, the request has not been fulfilled or aborted).

When a user issues a REQUEST/REPLY command and you have disabled all terminals as operator's terminals, OPCOM records all subsequent user's requests in the log file in the format shown above, but returns a message to the user indicating that no operator coverage is available.

All other OPCOM responses to REPLY commands, except responses involving the REPLY/ENABLE, REPLY/DISABLE, and REPLY/LOG commands, are not logged in the operator's log file.

**9.2.4.6 Changes to System Parameters Through the SYSGEN Utility -** Users with the CMKRNL privilege can use the SYSGEN Utility to change system parameters in the running (active) system. Users with the SYSPRV privilege can use the SYSGEN Utility to change system parameters in the current system. OPCOM logs all changes made to system parameters with messages in the following format:

```
%OPCOM, dd-mmm-yyyy hh:mm:ss.cc, message from user user-name
%SYSGEN-I-WRITExxx, system-mode system parameters modified by process ID n into file y
```

**9.2.4.7 DECnet-VAX Messages -** For information on DECnet-VAX status messages, see the DECnet-VAX System Manager's Guide.

## 9.3 REPORTING SOFTWARE PROBLEMS

To inform DIGITAL about problems with the VAX/VMS operating system or about errors in VAX/VMS software documents, you should use the Software Performance Report (SPR), which is illustrated in Figure 9-2. Complete directions for completing the SPR form accompany the form itself.

A supply of SPR forms is included in the VAX/VMS software distribution kit; more forms can be obtained from an SPR center. The addresses of these centers are listed on the backs of the forms.

This section offers advice on how to use this service most effectively, by describing the information that should be provided with all SPRs. Depending on the problem, this information will vary in quantity and content. Remember that the more information you include, the easier it will be for DIGITAL to resolve the problem.

digital

SOFTWARE PERFORMANCE REPORT

FIELD NO.: | CORPORATE SPR NO.: 395878

PAGE ____ OF____

TO SET UP FOR PROPER ALIGNMENT, START AT MARK BELOW.

| OPERATING SYSTEM | VERSION | SYSTEM PROGRAM OR DOCUMENT TITLE | VERSION OR DOCUMENT PART NO. | DATE |
|---|---|---|---|---|
| | | | | |

NAME:
FIRM:
ADDRESS:
CUST. NO.:

DEC OFFICE

DO YOU HAVE SOURCES? YES ☐ NO ☐

REPORT TYPE/PRIORITY

☐ PROBLEM/ERROR
☐ SUGGESTED ENHANCEMENT
☐ OTHER

1. ☐ HEAVY SYSTEM IMPACT
2. ☐ MODERATE SYSTEM IMPACT
3. ☐ MINOR SYSTEM IMPACT
4. ☐ NO SIGNIFICANT IMPACT
5. ☐ DOCUMENTATION/SUGGESTION

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 |
|---|---|---|---|---|---|
| | | | | | ☐ |

ALL SUBMISSIONS BECOME THE PROPERTY OF DIGITAL EQUIPMENT CORPORATION.

| SHORT NAME | MNT. CAT. | MNT. GRP. | XFER GRP. | PL | PRB. TYPE |
|---|---|---|---|---|---|
| | | | | | |

| DATE RECEIVED (MAIL) | DATE TO MAINTAINER | XFER DATE | LOGGED ON |
|---|---|---|---|
| | | | |

| DATE RECEIVED (ASG) | DATE RECEIVED FROM MAINTAINER | DATE ANSWERED | LOGGED OFF |
|---|---|---|---|
| | | | |

EN-01044-07-REV. H (35C)

ADMINISTRATIVE SERVICES GROUP, SWS

Figure 9-2: Software Performance Report (SPR)

### 9.3.1 The Problem Environment

You should supply a complete description, usually in the form of a batch log or console listing, that shows exactly how the problem was produced. Merely supplying the output produced by the problem is not enough. You must provide a complete scenario depicting what you did. The problem may be caused by an interaction between various system events, software packages, devices, SYSGEN parameters, DCL symbols, or logical names. Consider including some or all of the displays these commands produce, depending on your problem:

```
$ SHOW LOGICAL/ALL
$ SHOW SYMBOL/ALL/GLOBAL
$ RUN SYS$SYTEM:SYSGEN
$ SYSGEN>  USE ACTIVE
$ SYSGEN>  SHOW /ALL
$ SYSGEN>  SHOW /SPECIAL
```

### 9.3.2 Limiting the Problem Scope

As much as possible, eliminate all extraneous elements from the scenario you provide. For example, if the execution of a very complex program causes a problem, strip the program to just the code that causes the problem or write a small program that reproduces the problem. This action may help you locate logic errors, and will enable DIGITAL to isolate the problem more quickly.

### 9.3.3 Machine-readable Files

Supply any software needed to reproduce the problem, if possible. This may include source programs, image files, sample data, command procedures, and other items. When you submit source programs, be sure to include any libraries or require files that you reference. These files should all be provided in machine-readable format. Console media or magnetic tape are the best media to include with an SPR.

If the problem involves a system failure, then include the system dump file. Write the data onto a Files-11 Structure Level 2 disk or an ANSII magnetic tape. The following commands will copy the system dump file to an ANSII magnetic tape:

```
$ MOUNT/FOREIGN MTA0:  DUMPS
$ BACKUP SYS$SYSTEM:SYSDUMP.DMP  MTA0:DUMP/SAVE_SET
```

You can copy files to the console medium using the following DCL commands:

```
$ RUN SYS$SYSTEM:SYSGEN
CONNECT CONSOLE
```

(Now remove the console medium and place a scratch medium in the console device.)

```
$ INITIALIZE CSA1:  SPRDATA
$ MOUNT CSA1:  SPRDATA
$ CREATE/DIRECTORY CSA1:[SPR]
$ BACKUP MYDATA.DAT,MYIMAGE.EXE CSA1:[SPR] DATA/SAVE_SET
$ DISMOUNT CSA1:
```

When you provide machine-readable data, always include the exact instructions used to write the medium and instructions for reading the medium. (Avoid using other media formats, since doing so can cause unnecessary problems. For example, using FLX without /IM creates an unusable dump file since FLX eliminates all bytes of zero.) All machine-readable media you submit with SPRs will be returned to you.

### 9.3.4 System Environment

Every site runs a different type of workload. Some problems only appear under certain conditions. For example, some sites give different classes of users different base priorities. These sites may encounter problems that other sites do not. Such background information can be extremely important in resolving the problem, especially for system hangs or system failures.

You should describe any special software packages that are running, and mention any unusual hardware devices or user-written drivers on your system.

Also include the various version numbers of different pieces of software. For example, if a problem occurs while accessing local symbols during a DEBUG session, specify the version numbers of DEBUG and all compilers or assemblers used.

Sometimes DIGITAL forwards special patches or prereleases of patches for problems that are seriously affecting the system. If you are using any such patches, mention them in the SPR.

### 9.3.5 User Analysis (Optional)

Optionally you can include an analysis of what you believe is causing the problem. Include miscellaneous information such as, "the problem could only be reproduced when xyz happened," or "the problem does not occur on Version Vx.y".

### 9.3.6 Problem-specific Information to Include

DIGITAL requires different kinds of information to resolve different kinds of problems. Use Table 9-1 as an initial checklist to begin collecting the proper information to forward with your SPR.

Table 9-1: Typical Information Requirements for SPRs

| Problem | Information to Collect |
|---|---|
| System Bugcheck/Failure | A machine-readable copy of the system dump file.[1] (Do NOT send output from the SDA Utility since it usually does not provide enough information to resolve the problem. If you send the dump file, DIGITAL can always run SDA to obtain as much information and possibly more.)<br><br>A copy of the error log at the time of the error to help DIGITAL determine if the system problem was triggered by hardware errors.[2] |
| Machine Check | A copy of the error log at the time of the error.[2]<br><br>A machine-readable copy of the system dump file.[1] |
| System Hang | A copy of the system dump file, obtained after you crash the system in the manner described in the software installation guide for your VAX-11 processor.<br><br>This causes the system to bugcheck in a manner that is recognizable as a forced crash. Include the console listing and a description of the currently running work load. |
| Executive | A listing of the active values of the system parameters. (Invoke the SYSGEN Utility and specify the USE ACTIVE, SHOW/ALL, and SHOW/SPECIAL commmands.)<br><br>A machine-readable copy of the source program showing the problem.<br><br>A copy of the error log at the time of the problem.[2] |
| Devices | A copy of the error log at the time of the problem.[2] |

1. The raw data file (SYS$SYSTEM:SYSDUMP.DMP), not the formatted output from the SDA Utility. This provides DIGITAL a chance to run SDA in enough ways to obtain sufficient information to analyze the problem.

2. The raw data file (SYS$ERRORLOG:ERRLOG.SYS), not the formatted output from the SYE Utility. This provides DIGITAL a chance to run SYE in enough ways to obtain sufficient information to analyze the problem.

(continued on next page)

Table 9-1 (Cont.): Typical Information Requirements for SPRs

| Problem | Information to Collect |
|---|---|
| Files | A listing of all the information available on the file, obtained with the DCL command DIRECTORY/FULL or a dump header of the file's directory obtained with the DCL command DUMP/HEADER.<br><br>A machine-readable copy of the file itself. |
| Intermittent | A copy of the error log at the time of the problem.[2] |
| Command Language Interpreters | A listing obtained from the DCL commands SHOW SYMBOL/ALL/GLOBAL and SHOW LOGICAL/ALL. |
| Job Controller | A copy of the console printout and a machine-readable copy of the file SYS$SYSTEM:SNAPSHOT.DAT that is produced when the job controller aborts. |
| Librarian | A machine-readable copy of the library itself.<br><br>Machine-readable copies of all input files to the library.<br><br>A listing of all the information available on the library file, obtained from the DCL command DIRECTORY/FULL.<br><br>A listing of the library contents, obtained from the DCL command LIBRARY/LIST/FULL.<br><br>(If the problem cannot be reproduced, describe the scenario and include any command files used.) |
| Linker | Machine-readable copies of the object files and libraries used in the link.<br><br>A full map obtained by the DCL command LINK/MAP/FULL. |

2. The raw data file (SYS$ERRORLOG:ERRLOG.SYS), not the formatted output from the SYE Utility. This provides DIGITAL a chance to run SYE in enough ways to obtain sufficient information to analyze the problem.

(continued on next page)

Table 9-1 (Cont.): Typical Information Requirements for SPRs

| Problem | Information to Collect |
|---|---|
| Network | A description of the configurations of the systems involved in the problem, including the versions of the operating systems and DECnet-VAX, the hardware on both systems, and the patch level of the DECnet software on the non-VAX/VMS system, if applicable. |
| Terminals | A list of terminal characteristics produced by the DCL command SHOW TERMINAL.<br><br>A description of the type of terminal, the type of modem (if any), any special front-end equipment, and any unusual terminal configuration. |
| Compiler or Assembler | A machine-readable copy of the source program that caused the problem, including all require files and libraries that are referenced. (Try to limit the scope of the problem.)<br><br>A description of the compiler (or assembler) and operating system, including the version of each. |

# CHAPTER 10

# SYSTEM PARAMETERS

This chapter describes the VAX/VMS system parameters and the importance of each, as well as suggested values for each. Chapter 11 discusses the System Generation Utility (SYSGEN), which you can use to change the values of system parameters. Chapter 12 describes the AUTOGEN command procedure, which is the preferred tool for changing system parameters. (AUTOGEN uses SYSGEN.) Chapter 12 also describes how some of the system parameters have significant impact on system performance and provides some further suggestions on how to modify the parameters, if necessary.

During system operation you can run SYSGEN (described in the VAX-11 Utilities Reference Manual) to modify the current values or the values of any parameter file or create a new parameter file, for use in subsequent bootstrap operations. SYSGEN also enables you to dynamically alter the running system configuration by modifying a subset of the active values.

It is important to understand the terms current and active when they are used to refer to system parameters. The term active parameters refers to the parameter values the system is actively running with; that is, the values that are active at the present time. The only active parameters that can be changed on a running system are those that are categorized as dynamic parameters (see Section 10.1). The term current parameters refers to those values stored on disk (SYS$SYSTEM:SYS.EXE) that are used to boot the system. The current parameters become the active parameters at each bootstrap operation. When you modify certain active parameters with SYSGEN, you have no effect on the values of the current parameters; you merely change the values of these parameters while the system is running. The next time you bootstrap the system, the old values of the current parameters are established as the active parameters. (When you want to change the values of the current parameters on disk, you must use the SYSGEN command WRITE CURRENT. Furthermore, when you want to change the value of any active parameter that is not in the dynamic category, you must not only issue the WRITE CURRENT command, but you must also reboot the system to make it active. Chapter 11 describes how to do this.)

## 10.1 PARAMETER CATEGORIES

The system parameters fall into eleven general categories:

- MAJOR -- Parameters most likely to require modification
- SYS -- Parameters that affect overall system operation
- JOB -- Job control parameters

- ACP -- Parameters associated with Files-11 ancillary control processes (ACPs)
- TTY -- Parameters associated with terminal behavior
- SCS -- Parameters that control System Communication Services (SCS) and port driver operation. The SCS parameters that affect SCS operation have the prefix SCS. Those SCS parameters that affect the CI780 port driver have the prefix PA
- RMS -- Parameters associated with VAX-11 RMS
- PQL -- Parameters associated with process creation limits and quotas
- GEN -- Parameters that affect the creation and initialization of data structures at bootstrap time
- SPECIAL -- Special parameters used by DIGITAL
- DYNAMIC -- Parameters whose active values can be modified

There is also a group of parameters that can be user-defined: USERD1, USERD2, USER3, and USER4. (USERD1 and USERD2 are in the dynamic category.)

Each parameter has associated with it default, minimum, and maximum values that define the scope of allowable values. Tables 10-1 through 10-8 briefly describe the parameters in each category. To determine the default, minimum, and maximum values you should invoke the System Generation Utility (see Chapter 11) and issue the appropriate SHOW command. For example, to display the values for the MAJOR parameters, you can specify SHOW/MAJOR. (The SPECIAL parameters are not documented; they should be used only by DIGITAL personnel.)

**Table 10-1: MAJOR Parameters**

| Parameter Name | Description | Dynamic |
|---|---|---|
| PFCDEFAULT | Default page fault cluster size (in pages) | D |
| GBLSECTIONS | Number of global section descriptors | |
| GBLPAGES | Number of global page table entries | |
| MAXPROCESSCNT | Maximum number of processes | |
| SYSMWCNT | Maximum size of system working set (in pages) | |
| BALSETCNT | Maximum number of resident working sets | |
| IRPCOUNT | Number of preallocated intermediate request packets | |

(continued on next page)

Table 10-1 (Cont.): MAJOR Parameters

| Parameter Name | Description | Dynamic |
|---|---|---|
| WSMAX | Maximum size of any working set (in pages) | |
| NPAGEDYN | Size of nonpaged dynamic pool (in bytes, but rounded down to an integral number of pages by the system) | |
| PAGEDYN | Size of paged dynamic pool (in bytes, but rounded down to an integral number of pages by the system) | |
| VIRTUALPAGECNT | Maximum virtual space per process (in pages) | |
| LRPCOUNT | Number of preallocated large request packets | |
| SRPCOUNT | Number of preallocated small request packets | |
| QUANTUM | Maximum time a process can use at once and minimum service a process must receive before being outswapped, (in 10ms units) | D |
| PFRATL | Page fault rate low limit (in faults per 10 seconds of processor time) | D |
| PFRATH | Page fault rate high limit (in faults per 10 seconds of processor time) | D |
| WSINC | Working set increment (in pages) | D |
| WSDEC | Working set decrement (in pages) | D |
| FREELIM | Minimum size of the free page list | |
| FREEGOAL | Number of pages required on the free page list after a memory shortage | |
| GROWLIM | Number of pages required on the free page list to allow process to exceed working set quota | D |
| BORROWLIM | Minimum size of the free page list before process can grow beyond process working set quota (in pages) | D |
| LOCKIDTBL | Number of entries in the system lock id table | |
| RESHASHTBL | Number of entries in the lock management resource name hash table | |

Table 10-2: SYS Parameters

| Parameter Name | Description | Dynamic | MAJOR |
|---|---|---|---|
| PFCDEFAULT | Default page fault cluster size (in pages) | D | M |
| KFILSTCNT | Number of known file list heads | | |
| GBLSECTIONS | Number of global section descriptors | | M |
| GBLPAGES | Number of global page table entries | | M |
| GBLPAGFIL | Maximum number of system-wide pages allowed for global page-file sections | | |
| MAXPROCESSCNT | Maximum number of processes | | M |
| PROCSECTCNT | Number of process sections | | |
| MINWSCNT | Minimum number of fluid pages in any working set | | |
| PAGFILCNT | Maximum number of paging files that can be installed | | |
| SWPFILCNT | Maximum number of swapping files that can be installed | | |
| SYSMWCNT | Maximum size of system working set (in pages) | | M |
| INTSTKPAGES | Size of interrupt stack (in pages) | | |
| BALSETCNT | Maximum number of resident working sets | | M |
| IRPCOUNT | Number of preallocated intermediate request packets | | M |
| IRPCOUNTV | Maximum size to which IRPCOUNT can be increased | | |
| WSMAX | Maximum size of any working set (in pages) | | M |
| NPAGEDYN | Size of nonpaged dynamic pool (in bytes, but rounded down to an integral number of pages by the system) | | M |
| NPAGEVIR | Maximum size to which NPAGEDYN can be increased | | |
| PAGEDYN | Size of paged dynamic pool (in bytes, but rounded down to an integral number of pages by the system) | | M |
| VIRTUALPAGECNT | Maximum virtual space per process (in pages) | | M |

(continued on next page)

Table 10-2 (Cont.): SYS Parameters

| Parameter Name | Description | Dynamic | MAJOR |
|---|---|---|---|
| SPTREQ | Number of additional system page table entries | | |
| LRPCOUNT | Number of preallocated large request packets | | M |
| LRPCOUNTV | Maximum value to which LRPCOUNT can be increased | | |
| LRPSIZE | Size of the large request packets (in bytes) | | |
| SRPCOUNT | Number of preallocated small request packets | | M |
| SRPCOUNTV | Maximum value to which SRPCOUNT can be increased | | |
| QUANTUM | Maximum time a process can use at once and minimum service a process must receive before being outswapped, (in 10ms units) | D | M |
| MPW_WRTCLUSTER | Number of pages written per I/O from the modified page list | | |
| MPW_HILIMIT | Maximum size of modified page list (in pages) | | |
| MPW_LOLIMIT | Minimum size of modified page list (in pages) | | |
| MPW_THRESH | Minimum size of the modified page list requiring swapper action (in pages) | D | |
| MPW_WAITLIMIT | Number of pages on the modified page list that forces a process to wait until the next time the modified page writer writes the modified page list | D | |
| PFRATL | Page fault rate low limit (in faults per 10 seconds of processor time) | D | M |
| PFRATH | Page fault rate high limit (in faults per 10 seconds of processor time) | D | M |
| WSINC | Working set increment (in pages) | D | M |
| WSDEC | Working set decrement (in pages) | D | M |
| AWSMIN | Automatic working set minimum (in pages) | D | |
| AWSTIME | Automatic working set time for collecting sample (in 10ms units) | D | |

(continued on next page)

Table 10-2 (Cont.): SYS Parameters

| Parameter Name | Description | Dynamic | MAJOR |
|---|---|---|---|
| SWPOUTPGCNT | Process size before outswapping occurs (in pages) | D | |
| LONGWAIT | Time to elapse before a process is judged idle by the swapper (in 6.6ms units) | D | |
| EXTRACPU | Extra CPU time added to process after CPU time is expired (in 10ms units) | D | |
| MAXSYSGROUP | Highest system UIC | D | |
| MVTIMEOUT | Amount of time allowed for a mount verification attempt to succeed before it is cancelled | D | |
| MAXBUF | Maximum number of bytes that can be transferred in one buffered I/O | D | |
| DEFMBXBUFQUO | Default mailbox buffer quota (in bytes) | D | |
| DEFMBXMXMSG | Default mailbox maximum message size (in bytes) | D | |
| DEFMBXNUMMSG | Not implemented | D | |
| FREELIM | Lower limit of free page list (in bytes) | | M |
| FREEGOAL | Number of pages required on the free page list after a memory shortage | | M |
| GROWLIM | Number of pages required on the free page list to allow process to exceed working set quota | D | M |
| BORROWLIM | Minimum size of the free page list before process can grow beyond process working set quota (in pages) | D | M |
| XFMAXRATE | Maximum rate of transfer for DR32 devices | D | |
| LAMAPREGS | Number of map registers allocated to an LPA11 device driver | | |
| REALTIME_SPTS | Number of system page table entries reserved for connect-to-interrupt processes | | |
| CLISYMTBL | Size of command interpreter symbol table (in pages) | D | |
| LOCKIDTBL | Number of entries in the system lock id table | | M |

(continued on next page)

Table 10-2 (Cont.): SYS Parameters

| Parameter Name | Description | Dynamic | MAJOR |
|---|---|---|---|
| RESHASHTBL | Number of entries in the lock management resource name hash table | | M |
| DEADLOCK_WAIT | Time that a lock request waits before deadlock search is initiated (in seconds) | D | |
| TIMEPROMPTWAIT | Time allowed for entry of the system time during a boot (in micro-fortnights) | | |
| LOGSHASHTBL | Number of entries in the system logical name hash table | | |
| LOGGHASHTBL | Number of entries in the group logical name hash table | | |
| LOGPHASHTBL | Number of entries in the process logical name hash table | | |
| BUGREBOOT | Automatic reboot on fatal bugcheck; switch | D | |
| CRDENABLE | Detection and logging of memory-corrected read errors; switch | | |
| DUMPBUG | Writing of dump file on fatal bugcheck; switch | | |
| BUGCHECKFATAL | All bugchecks fatal; switch | D | |
| SETTIME | Time-of-day prompt at boot time; switch | | |
| UAFALTERNATE | Use of alternate UAF; switch | | |
| MOUNTMSG | Controls OPCOM handling of volume mount messages; switch | D | |
| DISMOUMSG | Controls OPCOM handling of volume dismount messages; switch | D | |
| DEFPRI | Sets the base default priority for processes | D | |

Table 10-3: TTY Parameters

| Parameter Name | Description | Dynamic |
|---|---|---|
| TTY_SCANDELTA | Terminal dial-up/hang-up scan interval (in increments of 100ns) | |
| TTY_DIALTYPE | Dialup flag bits | |
| TTY_SPEED | Default speed for terminals; code | |

(continued on next page)

Table 10-3 (Cont.): TTY Parameters

| Parameter Name | Description | Dynamic |
|---|---|---|
| TTY_RSPEED | Receive speed for terminals | |
| TTY_PARITY | Not implemented | |
| TTY_BUF | Default line width for terminal | |
| TTY_DEFCHAR | Default terminal characteristics, longword 1 | |
| TTY_DEFCHAR2 | Default terminal characteristics, longword 2 | |
| TTY_TYPAHDSZ | Size of terminal type-ahead buffer (in bytes) | |
| TTY_ALTYPAHD | Size of alternate type-ahead buffer (in bytes) | |
| TTY_ALTALARM | Size of alternate type-ahead buffer alarm (in bytes) | |
| TTY_DMASIZE | Minimum number of output buffer characters to invoke DMA transfers | D |
| TTY_PROT | Terminal protection against allocation by another process; mask | |
| TTY_OWNER | Owner UIC for TTY_PROT specification | |
| TTY_CLASSNAME | Terminal class driver name prefix for booting | |
| TTY_SILOTIME | Input silo polling interval for DMF-32 hardware (in milliseconds) | |

Table 10-4: JOB Parameters

| Parameter Name | Description | Dynamic |
|---|---|---|
| JOBQUEUES | Print or batch queue utilization; switch | D |
| REINITQUE | Reinitialization of queue file; switch | D |
| MAXPRINTSYMB | Maximum number of print symbionts | D |
| DEFPRI | Default priority | D |
| IJOBLIM | Not implemented | D |
| BJOBLIM | Not implemented | D |
| NJOBLIM | Not implemented | D |
| RJOBLIM | Maximum number of remote terminals | D |

Table 10-5: ACP Parameters

| Parameter Name | Description | Dynamic |
|---|---|---|
| ACP_MULTIPLE | One ACP per disk volume mounted on different device types; switch | D |
| ACP_SHARE | Sharing of ACP code; switch | |
| ACP_MAPCACHE | Size of bit map cache (in pages) | D |
| ACP_HDRCACHE | Size of file header cache (in pages) | D |
| ACP_DIRCACHE | Size of directory cache (in pages) | D |
| ACP_WORKSET | Working set default for ACP | D |
| ACP_FIDCACHE | Size of file identification cache (in pages) | D |
| ACP_EXTCACHE | Size of extent cache (in pages) | D |
| ACP_EXTLIMIT | Maximum amount of free space in extent cache (in tenths of a percent of available free space) | D |
| ACP_QUOCACHE | Number of entries in quota cache | D |
| ACP_SYSACC | Size of directory access cache (in pages) | D |
| ACP_MAXREAD | Maximum directory blocks to read (in blocks) | D |
| ACP_WINDOW | Default number of window pointers | D |
| ACP_WRITEBACK | Caching of file headers; switch | D |
| ACP_DATACHECK | Data verification on ACP I/O | D |
| ACP_BASEPRIO | Base priority for ACP processes | D |
| ACP_SWAPFLGS | Swapping of ACP working sets; code | D |

Table 10-6: RMS Parameters

| Parameter Name | Description | Dynamic |
|---|---|---|
| RMS_DFMBC | Default multiblock count | D |
| RMS_DFMBFSDK | Default multibuffer count for sequential disk operations | D |
| RMS_DFMBFSMT | Default multibuffer count for magnetic tape operations | D |

(continued on next page)

Table 10-6 (Cont.): RMS Parameters

| Parameter Name | Description | Dynamic |
|---|---|---|
| RMS_DFMBFSUR | Not implemented | D |
| RMS_DFMBFREL | Default multibuffer count for relative disk operations | D |
| RMS_DFMBFIDX | Default multibuffer count for indexed sequential disk operations | D |
| RMS_DFMBFHSH | Not implemented | D |
| RMS_PROLOGUE | Default file structure level for VAX-11 RMS files; code | D |
| RMS_EXTEND_SIZE | Default file extend size (in blocks) | D |
| RMS_FILEPROT | Default file protection; mask | |

Table 10-7: PQL Parameters

| Parameter Name | Description | Dynamic |
|---|---|---|
| PQL_DASTLM | Default number of pending ASTs | D |
| PQL_MASTLM | Minimum number of pending ASTs | D |
| PQL_DBIOLM | Default buffered I/O limit | D |
| PQL_MBIOLM | Minimum buffered I/O limit | D |
| PQL_DBYTLM | Default buffered I/O byte limit | D |
| PQL_MBYTLM | Minimum buffered I/O byte limit | D |
| PQL_DCPULM | Default CPU time limit (in increments of 10ms) | D |
| PQL_MCPULM | Minimum CPU time limit (in increments of 10ms) | D |
| PQL_DDIOLM | Default direct I/O limit | D |
| PQL_MDIOLM | Minimum direct I/O limit | D |
| PQL_DFILLM | Default open file limit | D |
| PQL_MFILLM | Minimum open file limit | D |
| PQL_DPGFLQUOTA | Default paging file quota | D |
| PQL_MPGFLQUOTA | Minimum paging file quota | D |
| PQL_DPRCLM | Default subprocess limit | D |

(continued on next page)

Table 10-7 (Cont.): PQL Parameters

| Parameter Name | Description | Dynamic |
|---|---|---|
| PQL_MPRCLM | Minimum subprocess limit | D |
| PQL_DTQELM | Default timer queue entries | D |
| PQL_MTQELM | Minimum timer queue entries | D |
| PQL_DWSDEFAULT | Default working set sizes | |
| PQL_MWSDEFAULT | Minimum default working set size | |
| PQL_DWSQUOTA | Default working set quota | D |
| PQL_MWSQUOTA | Minimum working set quota | D |
| PQL_DWSEXTENT | Default working set extent | D |
| PQL_MWSEXTENT | Minimum working set extent | D |
| PQL_DENQLM | Default number of locks queued at one time | D |
| PQL_MENQLM | Minimum number of locks queued at one time | D |

Table 10-8: SCS Parameters

| Parameter Name | Description | Dynamic |
|---|---|---|
| SCSBUFFCNT | Not implemented | |
| SCSCONNCNT | Maximum number of SCS connections for System Applications | |
| SCSRESPCNT | Maximum number of response descriptor table entries for System Applications | |
| SCSMAXDG | Maximum amount of application data in one datagram (in bytes) | |
| SCSMAXMSG | Maximum amount of application data in one SCS message (in bytes) | |
| SCSFLOWCUSH | Threshold value for notifying the remote SCS of new receive buffers | D |
| SCSSYSTEMID | Identifier of the system within a cluster | |
| PASTRETRY | Maximum number of times CI port driver attempts start datagram exchange | D |

(continued on next page)

Table 10-8 (Cont.): SCS Parameters

| Parameter Name | Description | Dynamic |
|---|---|---|
| PASTIMOUT | Time interval between CI port driver wakeups for time-based operations (in seconds) | D |
| PASTDGBUF | Maximum number of CI port driver start handshakes in progress simultaneously | |
| PAPOLLINTERVAL | Time between CI port driver polling activations (in seconds) | D |
| PAPOOLINTERVAL | Time between wakeups for CI port driver message buffer allocation requests (in seconds) | D |
| UDABURSTRATE | Not implemented | |

## 10.2 PARAMETERS

The remaining sections of the chapter describe the parameters in more detail and provide guidelines to help you decide whether or not to consider modifying them. The parameters are presented in alphabetical order for your convenience in referring to them.

Where the descriptions refer to the default value, they mean the value of the parameter that is contained internally in the SYSGEN Utility as the default value. The default values allow booting on any supported VAX/VMS configuration. (SYSGEN displays these default values under the heading "default" when you issue the SYSGEN command SHOW for one of the parameter categories. Also, these are the parameter values you can establish with the SYSGEN command USE DEFAULT.)

Where the descriptions refer to the computed, installed value, they mean the value derived by the AUTOGEN commmand procedure. These values are appropriate for booting on the specific configuration that initiated the AUTOGEN procedure.

### 10.2.1 ACP_BASEPRIO Parameter

ACP_BASEPRIO sets the base priority for all ACPs. The DCL command SET PROCESS/PRIORITY can be used to reset the base priorities of individual ACPs.

Normally the default value is adequate.

### 10.2.2 ACP_DATACHECK Parameter

ACP_DATACHECK enables verification of reading and/or writing of file structure data (for example, directories and file headers): a specification of 3 means read and write checks; 2 means write check only; 1 means read check only; 0 means no checks. On a read check, the ACP information is read twice and compared. On a write check, the ACP information is written, then read and compared.

### 10.2.3 ACP_DIRCACHE Parameter

ACP_DIRCACHE sets the number of pages for caching directory blocks.

Too small a value causes excessive ACP I/O operations, while too large a value causes excessive physical memory to be consumed by the ACP.

Normally the computed, installed value is adequate.

### 10.2.4 ACP_EXTCACHE Parameter

ACP_EXTCACHE sets the number of entries in the extent cache. Each entry points to one contiguous area of free space on disk. A specification of 0 means no cache.

Too small a value causes excessive ACP I/O operations, while too large a value causes excessive physical memory to be consumed by the ACP.

Normally the default value is adequate.

### 10.2.5 ACP_EXTLIMIT Parameter

ACP_EXTLIMIT specifies the maximum amount of free space to which the extent cache can point, expressed in thousandths of the currently available free blocks on the disk. For example, if available free space on the disk is 20,000 blocks, a specification of 10 limits the extent cache to 200 blocks. This parameter's purpose is to limit the amount of free space that might be lost in the event of a system failure. However, lost free space on a volume is normally recovered at mount time.

Normally the default value is adequate.

### 10.2.6 ACP_FIDCACHE Parameter

ACP_FIDCACHE sets the number of file identification slots cached. A specification of 1 means no cache.

Too small a value causes excessive ACP I/O operations, while too large a value causes excessive physical memory to be consumed by the ACP.

Normally the default value is adequate.

### 10.2.7 ACP_HDRCACHE Parameter

ACP_HDRCACHE sets the number of pages for caching file header blocks.

Too small a value causes excessive ACP I/O operations, while too large a value causes excessive physical memory to be consumed by the ACP.

Normally the computed, installed value is adequate.

### 10.2.8 ACP_MAPCACHE Parameter

ACP_MAPCACHE sets the number of pages for caching bit map blocks.

Too small a value causes excessive ACP I/O operations, while too large a value causes excessive physical memory to be consumed by the ACP.

Normally the computed, installed value is adequate.

### 10.2.9 ACP_MAXREAD Parameter

ACP_MAXREAD sets the maximum number of directory blocks read in one I/O operation. This parameter does not affect user file I/O.

Normally the default value is adequate.

### 10.2.10 ACP_MULTIPLE Parameter

ACP_MULTIPLE enables or disables the default creation of a separate disk ACP process for each volume mounted on a different device type. Performance on larger disks is better enhanced by increasing cache sizes than by adding another ACP. The parameter can be overridden on an individual-volume basis with the DCL command MOUNT.

### 10.2.11 ACP_QUOCACHE Parameter

ACP_QUOCACHE sets the number of quota file entries cached. A specification of 0 means no cache.

Too small a value causes excessive ACP I/O operations, while too large a value causes excessive physical memory to be consumed by the ACP.

Normally the computed, installed value is adequate.

### 10.2.12 ACP_SHARE Parameter

ACP_SHARE enables or disables the creation of a global section for the first ACP used, so that succeeding ACPs may share its code. This parameter should be set off (0) when ACP_MULTIPLE is off.

### 10.2.13 ACP_SWAPFLGS Parameter

ACP_SWAPFLGS enables or disables swapping for four classes of ACPs through the value of a four-bit number:

| Bit | Class of ACP |
|---|---|
| 0 | Disks mounted by MOUNT/SYSTEM |
| 1 | Disks mounted by MOUNT/GROUP |
| 2 | Private disks |
| 3 | Magnetic tape ACP |

If the value of the bit is 1, the corresponding class of ACPs can be swapped. The value of decimal 15 (hexadecimal F -- all bits on)

enables swapping for all classes of ACP. A value of decimal 14 disables swapping for ACPs for volumes mounted with the /SYSTEM qualifier, but leaves swapping enabled for all other ACPs.

### 10.2.14 ACP_SYSACC Parameter

ACP_SYSACC sets the number of directory file control blocks (FCBs) that will be cached for disks mounted with the /SYSTEM qualifier. Each directory FCB contains a 16-byte array containing the first letter of the last entry in each block of the directory (or group of blocks if the directory exceeds 16 blocks). Since entries in a directory are alphabetical, the cached FCB provides quick access to a required directory block. This parameter value should be roughly equivalent to the number of directories that will be in use concurrently on each system volume. It may be overridden on a per-volume basis with the /ACCESSED qualifier to the DCL command MOUNT. The value should be kept low in systems with small physical memory, as the FCBs require a significant amount of space in the nonpaged dynamic pool.

Too small a value causes excessive ACP I/O operations, while too large a value causes excessive physical memory to be consumed by the ACP.

Normally the computed, installed value is adequate.

### 10.2.15 ACP_WINDOW Parameter

ACP_WINDOW sets the default number of window pointers to be allocated in a window for a default file access, for disks mounted with the /SYSTEM qualifier.

Normally the default value is adequate.

### 10.2.16 ACP_WORKSET Parameter

ACP_WORKSET sets the default size of a working set for an ACP. A value of 0 permits the ACP to calculate the size.

Too small a value causes excessive ACP paging, while too large a value causes excessive physical memory to be consumed by the ACP. This value should be nonzero only on small systems where memory is tight.

### 10.2.17 ACP_WRITEBACK Parameter

ACP_WRITEBACK enables the deferred writing of file headers. A specification of 0 causes all modifications of file headers to be written to disk immediately.

### 10.2.18 AWSMIN Parameter

AWSMIN establishes the lowest number of pages to which a working set limit can be decreased by automatic working set adjustment.

Normally the default value is adequate.

### 10.2.19 AWSTIME Parameter

AWSTIME specifies the minimum amount of processor time that must elapse for the system to collect a significant sample of a working set's page fault rate. The time is expressed in units of 10 milliseconds. The default value of 20, for example, is 200 milliseconds.

### 10.2.20 BALSETCNT Parameter

BALSETCNT sets the number of balance set slots in the system page table. Each memory-resident working set requires one balance set slot. Each balance set slot requires 4 bytes of permanently resident memory per 128 virtual pages (as specified in the VIRTUALPAGECNT parameter).

You can monitor the active system with the DCL command SHOW MEMORY or the MONITOR PROCESSES command of the Monitor Utility (see the VAX-11 Utilities Reference Manual) to determine the actual maximum number of working sets in memory. If this number is significantly lower than the value of BALSETCNT, this parameter value may be lowered. If all balance set slots are being used, the value of BALSETCNT should be raised.

BALSETCNT should never be set to a value higher than two less than MAXPROCESSCNT. If physical memory is a significant system constraint, you should consider lowering this value even further. However, if your system runs with a number of processes nearly equal to MAXPROCESSCNT, lowering BALSETCNT will force swapping to occur, which can affect system performance.

### 10.2.21 BJOBLIM Parameter

BJOBLIM is not currently implemented.

### 10.2.22 BORROWLIM Parameter

BORROWLIM defines the minimum number of pages required on the free page list before the system will permit process growth beyond the working set quota (WSQUOTA) for the process. This parameter should always be greater than FREELIM.

With the system's use of automatic working set adjustment, via the parameters WSINC, PFRATH, and AWSTIME, this parameter allows a process to grow beyond the value set by the working set quota (WSQUOTA) to the working set quota extent (WSEXTENT) on a system that has a substantial memory on the free page list. Such growth attempts to alleviate heavy page faulting. To make use of this growth, you must also set the user's WSEXTENT authorization quota to a larger number than the WSQUOTA value.

### 10.2.23 BUGCHECKFATAL Parameter

BUGCHECKFATAL enables or disables the conversion of nonfatal bugchecks into fatal bugchecks. The system must be rebooted on a fatal bugcheck. A nonfatal bugcheck only places an entry in the error log and deletes the corresponding process.

This parameter should normally be off (0); you should only set it on (1) when the executive is being debugged.

Normally the default value is adequate.

### 10.2.24 BUGREBOOT Parameter

BUGREBOOT enables or disables automatic rebooting of the system if a fatal bugcheck occurs. This parameter should normally be on (1) and only off (0) when the executive is being debugged.

### 10.2.25 CLISYMTBL Parameter

CLISYMTBL sets the size of the command interpreter symbol table, which controls the number of DCL or MCR symbols that can be created.

Normally the default value is adequate.

### 10.2.26 CRDENABLE Parameter

CRDENABLE enables or disables detection and logging of memory-corrected read data (ECC) errors. This parameter should normally be on (1).

### 10.2.27 DEADLOCK_WAIT Parameter

DEADLOCK_WAIT defines the number of seconds that a lock request must wait before the system initiates a deadlock search on behalf of that lock. Setting DEADLOCK_WAIT to 0 disables deadlock checking. Setting DEADLOCK_WAIT to a value greater than 0 but still less than the default setting provides faster detection of deadlocks, but requires more CPU usage.

Normally the default value is adequate.

### 10.2.28 DEFMBXBUFQUO Parameter

DEFMBXBUFQUO sets the default for the mailbox buffer quota size in bytes when this value is not specified in a Create Mailbox ($CREMBX) system service call.

Normally the default value is adequate.

### 10.2.29 DEFMBXMXMSG Parameter

DEFMBXMXMSG sets the default for the mailbox maximum message size in bytes when this value is not specified in a Create Mailbox ($CREMBX) system service call.

Normally the default value is adequate.

### 10.2.30 DEFMBXNUMMSG Parameter

DEFMBXNUMMSG is not currently implemented.

### 10.2.31 DEFPRI Parameter

DEFPRI sets the base default priority for processes.

Normally the default value is adequate.

### 10.2.32 DISMOUMSG Parameter

DISMOUMSG controls whether or not the messages that log volume dismounts appear on the operator's terminal and in the operator's log. The default value of 0 disables the reporting of these messages.

### 10.2.33 DUMPBUG Parameter

DUMPBUG enables or disables the writing of error log buffers and memory contents to SYS$SYSTEM:SYSDUMP.DMP when a fatal bugcheck occurs. This parameter should be off (0) only when the executive is being debugged.

### 10.2.34 EXTRACPU Parameter

EXTRACPU sets the time, in units of 10 milliseconds, allotted to each of a process's exit handlers (for each access mode) after the process times out (that is, reaches its CPU time limit).

Normally the default value is adequate.

### 10.2.35 FREEGOAL Parameter

FREEGOAL establishes the number of pages desired on the free page list as the result of a system memory shortage. Memory shortages occur when the system drops below the minimum number of pages required on the free page list (FREELIM). The value of FREEGOAL must always be greater than or equal to the value of FREELIM.

Normally the computed, installed value is adequate.

### 10.2.36 FREELIM Parameter

FREELIM sets the minimum number of pages that must be on the free page list. The system will write pages from the modified page list, swap out working sets, or reduce the size of the working sets to maintain the minimum count.

While the larger free page list generally means less paging I/O, it also means less space for the balance set, which tends to result in more swapping I/O. You can monitor the size of the free page list,

the amount of paging, and the amount of swapping with the MONITOR IO command of the Monitor Utility (see the VAX-11 Utilities Reference Manual).

Normally the computed, installed value is adequate.

### 10.2.37 GBLPAGES Parameter

GBLPAGES sets the number of global page table entries allocated at bootstrap time. Each global section requires one global page table entry per section page, plus two entries, with the total rounded up to an even number. Every 128 global page table entries add 4 bytes to permanently resident memory in the form of a system page table entry.

The default value is more than adequate for the images normally installed as shared in the system start-up command procedures. Once the system is running and all global sections are created, you can examine the actual requirements with the /GLOBAL option of the Install Utility (see the VAX-11 Utilities Reference Manual) and reduce the value of GBLPAGES accordingly. However, the value of this parameter should not be set too small, since the page table entries are not expensive in terms of permanently resident memory. If you plan to install many user images as shared, or if user programs are likely to create many global sections, you must increase the value of this parameter.

Note that as you increase GBLPAGES beyond its default setting, you must make an adjustment to SYSMWCNT. For every 128 pages you add to GBLPAGES, increase SYSMWCNT by 1.

### 10.2.38 GBLPAGFIL Parameter

GBLPAGFIL defines the maximum number of system-wide pages allowed for global page-file sections, that is, scratch global sections that can be used without being mapped to a file. These global page-file sections can be temporary, permanent, system, or group and are allocated from the paging file specified in the system process header (the paging file specified at boot time). Thus, when you allow pages for global page-file sections, be sure to increase the size of the paging file accordingly.

Global page-file sections are created with the Create and Map Section ($CRMPSC) system service without an explicit disk file. These sections are used for the VAX-11 RMS global buffers required for shared files. Users of shared files are warned that global page-file sections cause both global page table and the default system paging file (PAGEFILE.SYS) to be used. If the value of GBLPAGFIL is too small, the $CRMPSC system service issues an error message when users attempt to create global page-file sections.

You need scratch global sections if you use VAX-11 RMS global buffers. Each file using global buffers requires approximately the following number of pages in the system paging file: the file's bucket size multiplied by the number of global buffers for that file.

Normally the default value for this parameter is adequate for most systems. However, if your site uses VAX-11 RMS global buffering to a significant extent, you will probably need to raise the value of GBLPAGFIL. Global buffers are normally enabled with the DCL command SET FILE/GLOBAL_BUFFERS, which is described in the VAX/VMS Command Language User's Guide.

You can list the global sections with the /GLOBAL qualifier of the Install Utility. The sections used by VAX-11 RMS for global buffers all begin with the prefix RMS$ followed by eight hexadecimal digits representing the longword address.

### 10.2.39 GBLSECTIONS Parameter

GBLSECTIONS sets the number of global section descriptors allocated in the system header at bootstrap time. Each global section requires one descriptor. Each descriptor takes 32 bytes of permanently resident memory.

The default value is more than adequate for the images normally installed as shared in the system start-up command procedures. You can, once the system is running and all global sections are created, examine the actual requirements with the /GLOBAL option of the Install Utility (see the VAX-11 Utilities Reference Manual) and reduce the value of GBLSECTIONS accordingly. However, the value of this parameter should not be cut too closely -- each descriptor requires only 32 bytes. If you plan to install many user images as shared or if user programs are likely to create many global sections, you must increase the value of this parameter.

If the value of GBLSECTIONS is too small, you receive a message from the Install Utility at system start-up time or whenever you install images manually. On the other hand, too large a value for GBLSECTIONS wastes physical memory.

### 10.2.40 GROWLIM Parameter

GROWLIM sets the number of pages that the system must have on the free page list so that a process can add a page to its working set when it is above quota. GROWLIM has no effect if the process is below its working set quota. The effect of GROWLIM is to act as a fast shutoff to the working set extent mechanism based on the system's free memory.

### 10.2.41 IJOBLIM Parameter

IJOBLIM is not currently implemented. You can control the maximum number of concurrent interactive users on the system with the DCL command SET LOGINS/INTERACTIVE.

### 10.2.42 INTSTKPAGES Parameter

INTSTKPAGES sets the size of the interrupt stack in pages. Each page on the interrupt stack requires a page of permanently resident memory.

The default value of 2 should be used unless interrupt-stack-not-valid exceptions occur. These may be caused by either an unusually large number of devices or a driver that requires a very large amount of stack space.

### 10.2.43 IRPCOUNT Parameter

IRPCOUNT sets the number of preallocated intermediate request packets. Each packet requires 160 bytes of permanently resident memory. If IRPCOUNT is too large, physical memory is wasted. If IRPCOUNT is too small, the system increases its value automatically, as needed, to permit the system to perform properly. However, the system cannot increase IRPCOUNT beyond the value of IRPCOUNTV. Furthermore, there is a minor physical memory penalty for allowing this growth. If IRPCOUNT is underconfigured, the penalty is 4 percent of physical memory from the configured value to the actual value on the running system.

You can use the DCL command SHOW MEMORY/POOL/FULL to determine IRPCOUNT usage.

### 10.2.44 IRPCOUNTV Parameter

IRPCOUNTV establishes the upper limit to which IRPCOUNT can be automatically increased by the system.

If this parameter is set too low, system performance can be adversely affected by preventing the system from using this memory allocation mechanism for nonpaged pool requests. There is a penalty of 1 percent of physical memory for any unused growth space (one longword for every three unused intermediate request packets).

Normally the computed, installed value is appropriate.

### 10.2.45 JOBQUEUES Parameter

JOBQUEUES determines whether or not print or batch queues can be defined or used. When JOBQUEUES is set to 0, no batch or print queues can be defined or used. Thus, users with a system UIC would be able to check for the existence of the file SYS$SYSTEM:JBCSYSQUE.EXE and delete it, if desired, to save space.

### 10.2.46 KFILSTCNT Parameter

KFILSCNT sets the number of known file list heads. Each active list head requires 68 bytes of permanently resident memory. Each extra, never-used list head requires 4 bytes. The actual known file entries are allocated from the paged dynamic pool.

One known file head is required for each set of installed images with a different combination of device name, directory name, and file type. (See Chapter 6.)

Normally the default value is adequate.

### 10.2.47 LAMAPREGS Parameter

LAMAPREGS sets the number of UNIBUS map registers allocated to an LPA11 driver when the driver is loaded, and limits the registers for the driver to that number. A value of 0 permits dynamic allocation of an unlimited number of registers.

### 10.2.48 LOCKIDTBL Parameter

LOCKIDTBL establishes the number of entries in the system lock id table, which limits the number of locks in the system. There must be one entry for each lock in the system; each entry requires four bytes. For simple timesharing systems, the default value is adequate. However, if your application uses many locks, as in the case of heavy VAX-11 RMS file sharing or a data base management application, you should increase this parameter. (Whenever you change the value of LOCKIDTBL, you should examine the value of RESHASHTBL and change it, if necessary.)

The VAX/VMS Lock Management facility is described in the chapter, Lock Management Services, in the VAX/VMS System Services Reference Manual. You can monitor locks with the MONITOR LOCK command of the Monitor Utility, as described in the VAX-11 Utilities Reference Manual.

If you set this parameter value too low, programs can receive the following error message:

```
%SYSTEM-E-NOLOCKID, no lock id. available
```

### 10.2.49 LOGGHASHTBL Parameter

LOGGHASHTBL sets the size of the group logical name hash table. Logical names are hashed using a function of the name length and contents. The LOGGHASHTBL parameter determines the number of entries in the group logical name hash table. The recommended setting is the average number of group logical names that will be in the table.

Normally the default value is adequate.

### 10.2.50 LOGPHASHTBL Parameter

LOGPHASHTBL sets the size of the process logical name hash table. Logical names are hashed using a function of the name length and contents. The LOGPHASHTBL parameter determines the number of entries in the process logical name hash table. The recommended setting is the average number of process logical names that will be in the table.

Normally the default value is adequate.

### 10.2.51 LOGSHASHTBL Parameter

LOGSHASHTBL sets the size of the system logical name hash table. Logical names are hashed using a function of the name length and contents. The LOGSHASHTBL parameter determines the number of entries in the system logical name hash table. The recommended setting is the average number of system logical names that will be in the table.

Normally the default value is adequate.

### 10.2.52 LONGWAIT Parameter

LONGWAIT defines how much real time must elapse before the swapper considers a process to be temporarily idle. This parameter is applied

to local event flag (LEF) and hibernate (HIB) waits to detect such conditions as an inactive terminal or ACP.

Normally the default value is adequate.

### 10.2.53 LRPCOUNT Parameter

LRPCOUNT sets the number of preallocated large request packets. Each packet requires an amount of permanently resident memory that is equal to the number of bytes specified by the LRPSIZE parameter. (Normally LRPSIZE is 576 bytes.) If LRPCOUNT is too large, physical memory is wasted. If LRPCOUNT is too small, the system increases its value automatically, as needed, to permit the system to perform properly. However, the system cannot increase LRPCOUNT beyond the value of LRPCOUNTV. If LRPCOUNT is underconfigured, the penalty is 4 percent of physical memory from the configured value to the actual value on the running system.

You can use the DCL command SHOW MEMORY/POOL/FULL to determine LRPCOUNT usage.

### 10.2.54 LRPCOUNTV Parameter

LRPCOUNTV establishes the upper limit to which LRPCOUNT can be automatically increased by the system.

If this parameter is set too low, system performance can be adversely affected by preventing the system from using this memory allocation mechanism for nonpaged pool requests. There is a penalty of 1 percent of physical memory for any unused growth space (approximately one longword for every unused large request packet).

Normally the computed, installed value is appropriate.

### 10.2.55 LRPSIZE Parameter

LRPSIZE is the size (in bytes) of the large request packets. The actual physical memory consumed by a large request packet is LRPSIZE + 64 bytes.

Normally the default value is adequate. However, for VAX-11 DECnet use, this parameter should be the same as the DECnet buffer size.

### 10.2.56 MAXBUF Parameter

MAXBUF sets the maximum size of a buffered I/O transfer (card readers, console floppy diskettes, line printers, mailboxes, and terminals). The space for a buffered I/O operation is allocated from the permanently resident nonpaged dynamic pool. Note that the system adds from 16 to 64 bytes (depending on the device driver and the nature of the I/O) to a buffer at allocation time for header information, so that the largest size transfer possible is reduced by this amount.

### 10.2.57 MAXPRINTSYMB Parameter

MAXPRINTSYMB sets the maximum number of print symbionts that can be created. (However, the maximum number of print symbionts is further restricted by the value of the PQL_DPRCLM parameter -- the default process creation limit.)

### 10.2.58 MAXPROCESSCNT Parameter

MAXPROCESSCNT sets the number of process entry slots allocated at bootstrap time. One slot is required for each concurrent process on the system. Each slot requires six bytes of permanently resident memory.

The default value is normally generously configured to allow you to create the desired number of processes. If the following message appears, you may need to adjust MAXPROCESSCNT:

```
%SYSTEM-F-NOSLOT, No PCB or swap slot to create process
```

This message can also be produced if the swapping file is full. Use the DCL command SHOW MEMORY to determine which limit is being reached.

### 10.2.59 MAXSYSGROUP Parameter

MAXSYSGROUP sets the highest value that a group number can have and still be classified as a system UIC group number. Note that the specification is not in octal unless preceded by the %O radix indicator. This parameter is normally left at 8 (10 octal).

### 10.2.60 MINWSCNT Parameter

MINWSCNT sets the minimum number of fluid pages -- that is, pages not locked in the working set -- required for the execution of a process. This value plus the size of the process header establishes the minimum working set size.

The value of MINWSCNT must provide sufficient space to execute any VAX-11 instruction. Theoretically, the worst-case instruction requires 52 pages; however, all VAX/VMS code can run with 20 fluid pages. An insufficient value may inhibit system performance or even put a process into an infinite loop on some instructions.

### 10.2.61 MOUNTMSG Parameter

MOUNTMSG controls whether or not the messages that log volume mounts appear on the operator's terminal and in the operator's log. The default value of 0 disables the reporting of these messages. (This parameter does not control the messages generated by mount assistance requests.)

### 10.2.62 MPW_HILIMIT Parameter

MPW_HILIMIT sets an upper limit for the modified page list. When the list accumulates the number of pages specified by this limit, writing

of the list begins. (The pages that are written are then transferred to the free page list.)

If MPW_HILIMIT is too low, excessive page faulting can occur from the paging file. If it is too high, too many physical pages can be consumed by the modified page list.

If you increase MPW_HILIMIT, you may also need to increase MPW_WAITLIMIT. Note that if MPW_WAITLIMIT is less than MPW_HILIMIT, a system deadlock will occur. The value for MPW_HILIMIT is normally equal to the value of MPW_WAITLIMIT.

Normally the default value is adequate.

### 10.2.63 MPW_LOLIMIT Parameter

MPW_LOLIMIT sets a lower limit for the modified page list. When writing of the list causes the number of pages on the list to drop to or below this limit, writing stops.

MPW_LOLIMIT ensures that a certain number of pages will be available on the modified page list for page faults. If it is too small, the caching effectiveness of the modified page list is reduced. If it is too high, less memory is available for processes, so that swapping (and paging) may increase.

Normally the default value is adequate.

### 10.2.64 MPW_THRESH Parameter

MPW_THRESH sets a lower bound of pages that must exist on the modified page list before the swapper writes this list to acquire free pages. If this requirement is met, the swapper will try to write the modified page list rather than take pages away from or swap out a process.

Normally the default value is adequate.

### 10.2.65 MPW_WAITLIMIT Parameter

MPW_WAITLIMIT sets the number of pages on the modified page list that will cause a process to wait until the next time the modified page writer writes the modified list. This acts to limit the rate at which any single process can produce modified pages. If this value is less than MPW_HILIMIT, a system deadlock will occur. The value for this parameter is normally equal to MPW_HILIMIT.

### 10.2.66 MPW_WRTCLUSTER Parameter

MPW_WRTCLUSTER sets the number of pages to be written from the modified page list during one I/O operation to the paging file or a section file. (The actual size of the cluster may be limited by the number of pages available for the I/O operation.) This parameter can range in value from 16 to 120, in multiples of eight. Each page in the cluster requires 6 bytes of permanently resident memory. If

MPW_WRTCLUSTER is too small, it will take many I/O operations to empty the modified page list. If MPW_WRTCLUSTER is too large for the speed of the disk that holds the page file, other I/O operations will be held up for the modified page list write.

Normally the default value is adequate.

### 10.2.67 MVTIMEOUT Parameter

MVTIMEOUT is the time in seconds that a mount verification attempt will continue on a given disk volume. If the mount verification does not recover the volume within that time, the I/O operations outstanding to the volume will terminate abnormally.

### 10.2.68 NJOBLIM Parameter

NJOBLIM is not currently implemented.

### 10.2.69 NPAGEDYN Parameter

NPAGEDYN sets the size of the nonpaged dynamic pool in bytes. This figure is rounded down to an integral number of pages. NPAGEDYN establishes the initial setting of the nonpaged pool size, but the pool size can be increased dynamically.

Probably the simplest and best approach to take in setting a value for this parameter is to initially use the default value, then monitor the amount of space actually used with the DCL command SHOW MEMORY/POOL/FULL.

There is a minor physical memory penalty for allowing this growth. If NPAGEDYN is underconfigured, the penalty is 4 percent of physical memory from the configured value to the actual value on the running system. You can decrease the value if much space is being wasted, and you should increase it if little space is unused.

### 10.2.70 NPAGEVIR Parameter

NPAGEVIR defines the maximum size to which NPAGEDYN can be increased. If this value is too small, the system could hang. If NPAGEVIR is too large, there is a penalty of 1 percent of physical memory for any unused growth space (one longword for for each 512 bytes of difference between the system's actual usage of nonpaged pool and the value of NPAGEVIR).

### 10.2.71 PAGEDYN Parameter

PAGEDYN sets the size of the paged dynamic pool in bytes. The specified value is rounded down to an integral number of pages. Each 512 bytes of paged dynamic pool adds four bytes of permanently resident memory to the system page table; the paged dynamic pool has no other direct memory requirements.

The paged dynamic pool is used to allocate storage for system and group logical names, resident image headers, known file list entries,

and VAX-11 RMS file sharing structures. Substantial amounts of space for the pool can be overallocated with little effect on system performance.

The size of the paged pool can grow dynamically up to the maximum size this parameter specifies.

Normally the default value is adequate.

### 10.2.72 PAGFILCNT Parameter

PAGFILCNT defines the maximum number of paging files that can be installed.

### 10.2.73 PAPOLLINTERVAL Parameter

PAPOLLINTERVAL is the number of seconds between polling activity for the computer interconnect port driver. Each time the poller activates, it sends Request IDs to all possible remote ports.

If no computer interconnect device is configured on your system, this parameter is ignored.

The default value should always be adequate.

### 10.2.74 PAPOOLINTERVAL Parameter

PAPOOLINTERVAL is the interval in seconds after which a CI port driver's suspended request for message buffer allocation from nonpaged pool is awakened to repeat the request.

If no computer interconnect (CI) device or UDA 50/52 is configured on your system, this parameter is ignored.

The default value should always be adequate.

### 10.2.75 PASTDGBUF Parameter

PASTDGBUF is the number of datagram receive buffers to queue for the CI port driver's configuration poller; that is, the maximum number of start handshakes that can be in progress simultaneously.

If no computer interconnect device is configured on your system, this parameter is ignored.

Normally the default value is adequate.

### 10.2.76 PASTIMOUT Parameter

PASTIMOUT is the basic interval at which the CI port driver wakes up to perform time-based bookkeeping operations. It is also the period after which a start handshake datagram is assumed to have timed out. Note that the product obtained by multiplying the values of PASTRETRY and PASTIMOUT must be greater than, or equal to, the value of PAPOLLINTERVAL.

If no computer interconnect device is configured on your system, this parameter is ignored.

The default value should always be adequate.

### 10.2.77 PASTRETRY Parameter

PASTRETRY is the number of times that the CI port driver's cluster configuration poller will attempt to exchange start datagrams with a newly enabled port without receiving a response, before it gives up on the remote system. Note that the product obtained by multiplying the values of PASTRETRY and PASTIMOUT must be greater than, or equal to, the value of PAPOLLINTERVAL.

If no computer interconnect device is configured on your system, this parameter is ignored.

The default values should always be adequate.

### 10.2.78 PFCDEFAULT Parameter

During execution of programs, PFCDEFAULT controls the number of image pages read from disk per I/O operation when a page fault occurs. The read I/O operations can take place from an image file or from the paging file. The actual size of the cluster can be less than PFCDEFAULT, depending on the size of image sections and the pattern of page references.

The value of this parameter should not be less than 16 to ensure adequate I/O performance, nor greater than one-fourth the default size of the average working set to prevent a single page fault from displacing a major portion of a working set. Too large a value for PFCDEFAULT can hurt system performance. PFCDEFAULT can be overridden on an image-by-image basis with the CLUSTER option of the VAX-11 Linker.

Normally the default value is adequate.

### 10.2.79 PFRATH Parameter

PFRATH specifies the page fault rate above which the limit of a working set will be automatically increased. The unit of measure is faults per 10 seconds of processor time. At a setting of 120, for example, the system will automatically increase the limit of a working set if it is faulting more than 120 pages per 10 seconds.

Decreasing the value of this parameter tends to increase the limits of the working sets, while increasing its value tends to decrease their limits.

### 10.2.80 PFRATL Parameter

PFRATL specifies the page fault rate below which the limit of a working set is automatically decreased. The unit of measure is faults per 10 seconds of processor time. At a setting of 1, for example, the system automatically decreases the limit of a working set if it is faulting less than 1 page every 10 seconds.

Increasing the value of this parameter tends to decrease the limits of the working sets, while decreasing its value tends to increase their limits.

### 10.2.81 PQL_DASTLM Parameter

PQL_DASTLM sets the default limit on the number of pending ASTs for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.82 PQL_DBIOLM Parameter

PQL_DBIOLM sets the default buffered I/O count limit for the number of outstanding buffered I/O operations permitted to a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.83 PQL_DBYTLM Parameter

PQL_DBYTLM sets the default buffered I/O byte count limit for the amount of buffered space available to a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.84 PQL_DCPULM Parameter

PQL_DCPULM sets the default CPU time limit for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process). PQL_DCPULM specifies the time limit in increments of 10 milliseconds.

The default value of 0 imposes no limit on CPU time usage and is typically the correct value for this parameter.

### 10.2.85 PQL_DDIOLM Parameter

PQL_DDIOLM sets the default direct I/O limit for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.86 PQL_DENQLM Parameter

PQL_DENQLM sets the default enqueue limit for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.87 PQL_DFILLM Parameter

PQL_DFILLM sets the default open file limit for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.88 PQL_DPGFLQUOTA Parameter

PQL_DPGFLQUOTA sets the default paging file quota for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.89 PQL_DPRCLM Parameter

PQL_DPRCLM sets the default subprocess limit for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.90 PQL_DTQELM Parameter

PQL_DTQELM sets the default number of timer queue entries for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.91 PQL_DWSDEFAULT Parameter

PQL_DWSDEFAULT sets the default working set size for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.92 PQL_DWSEXTENT

the default working set extent for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.93 PQL_DWSQUOTA Parameter

PQL_DWSQUOTA sets the default working set quota for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.94 PQL_MASTLM Parameter

PQL_MASTLM sets a default limit on the minimum number of pending ASTs for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.95 PQL_MBIOLM Parameter

PQL_MBIOLM sets the minimum buffered I/O limit for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.96 PQL_MBYTLM Parameter

PQL_MBYTLM sets the minimum buffered I/O byte limit for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.97 PQL_MCPULM Parameter

PQL_MCPULM sets the minimum CPU time limit in increments of 10 milliseconds for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.98 PQL_MDIOLM Parameter

PQL_MDIOLM sets the minimum direct I/O limit for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.99 PQL_MENQLM Parameter

PQL_MENQLM sets the default limit on the minimum number of locks that can be queued at one time by a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.100 PQL_MFILLM Parameter

PQL_MFILLM sets the minimum open file limit for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.101 PQL_MPGFLQUOTA Parameter

PQL_MPGFLQUOTA sets the minimum paging file quota for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.102 PQL_MPRCLM Parameter

PQL_MPRCLM sets the minimum subprocess limit for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.103 PQL_MTQELM Parameter

PQL_MTQELM sets the minimum number of timer queue entries for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.104 PQL_MWSDEFAULT Parameter

PQL_MWSDEFAULT sets the minimum default working set size for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.105 PQL_MWSEXTENT Parameter

PQL_MWSEXTENT sets the minimum working set extent for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.106 PQL_MWSQUOTA Parameter

PQL_MWSQUOTA sets the minimum working set quota for a process created by the Create Process ($CREPRC) system service or the DCL command RUN (Process).

Normally the default value is adequate.

### 10.2.107 PROCSECTCNT Parameter

PROCSECTCNT sets the number of section descriptors that a process can contain. Each section descriptor increases the fixed portion of the process header by 32 bytes.

You should set a value greater than the maximum number of image sections in any section to be run, as indicated by the linkage memory allocation map for the image.

Normally the default value is adequate.

### 10.2.108 QUANTUM Parameter

QUANTUM defines:

- Processor time -- Maximum amount of processor time a process can receive before control passes to another process of equal priority that is ready to compute
- Balance set residency -- Minimum amount of service a compute-state process must receive before being swapped out to secondary storage

Normally the default value is adequate.

### 10.2.109 REALTIME_SPTS Parameter

REALTIME_SPTS reserves a number of system page table entries for mapping connect-to-interrupt processes into system space. This value should normally remain at the default (0) in an environment that is not real-time. Where connect-to-interrupt processes do use the system, this value should represent the maximum number of pages all concurrent connect-to-interrupt processes must map into system space. See the VAX/VMS Real-Time User's Guide for details.

### 10.2.110 REINITQUE Parameter

REINITQUE specifies whether or not the job controller should create a new empty queue file. Chapter 8 describes how to use this parameter in the event a queue file becomes corrupted and must be cleared. Setting the parameter to 1 indicates that you want the queue file reinitialized to the empty state.

### 10.2.111 RESHASHTBL Parameter

RESHASHTBL defines the number of entries in the lock management resource name hash table. Each entry requires four bytes. As a general guideline, there should be one resource hash table entry for every four locks in the system. Thus, RESHASHTBL should be one-fourth the value of LOCKIDTBL, rounded up to the closest power of two.

### 10.2.112 RJOBLIM Parameter

RJOBLIM defines the maximum number of remote terminals allowed in the system at any one time. This parameter is sampled when REMACP is started.

### 10.2.113 RMS_DFMBC Parameter

RMS_DFMBC defines the VAX-11 RMS multiblock count for sequentially organized files. This value defines the number of 512-byte blocks to be transferred on each I/O operation.

Normally the default value is adequate.

### 10.2.114 RMS_DFMBFHSH Parameter

RMS_DFMBFHSH is not currently implemented.

### 10.2.115 RMS_DFMBFIDX Parameter

RMS_DFMBFIDX defines the default VAX-11 RMS multibuffer count for indexed sequential disk operations. This value defines the number of I/O buffers that VAX-11 RMS allocates for each indexed file. For sequential access, a larger number that allows some of the index buckets to remain in memory can improve performance.

### 10.2.116 RMS_DFMBFREL Parameter

RMS_DFMBFREL defines the default VAX-11 RMS multibuffer count for relative disk operations. This value defines the number of I/O buffers that VAX-11 RMS allocates for each relative file.

Normally the default value is adequate.

### 10.2.117 RMS_DFMBFSDK Parameter

RMS_DFMBFSDK defines the default VAX-11 RMS multibuffer count for sequential disk operations. This value defines the number of I/O buffers that VAX-11 RMS allocates for sequential disk files.

Normally the default value is adequate. However, if read/ahead or write/behind operations are used, a larger number will improve performance.

### 10.2.118 RMS_DFMBFSMT Parameter

RMS_DFMBFSMT defines the default VAX-11 RMS multibuffer count for magnetic tape operations. This value defines the number of I/O buffers that VAX-11 RMS allocates for magnetic tape files.

Normally the default value is adequate.

### 10.2.119 RMS_DFMBFSUR Parameter

RMS_DFMBFSUR is not currently implemented.

### 10.2.120 RMS_EXTEND_SIZE

RMS_EXTEND_SIZE specifies the number of blocks by which to extend files as they are written. This number should be chosen to balance the amount of extra disk space wasted at the ends of each file versus the performance improvement provided by making large extends infrequently. When small disk quotas are used, a small number such as the disk cluster size should be specified to prevent the user's disk quota from being consumed. If the value of 0 is used, VAX-11 RMS will allocate large extents and truncate the file back to its actual usage when it closes.

### 10.2.121 RMS_FILEPROT Parameter

RMS_FILEPROT determines the initial default file protection of all processes. Thus, it determines file protection for all users who do not execute the DCL command SET PROTECTION/DEFAULT in their login command procedure or later. This parameter also affects the protection of files created by system processes, such as those that create the error log, the operator log, and others. The protection is expressed as a mask (see Section 3.2 for more information on specifying protection masks). By default, the mask is 64000 (decimal) or FA00 (hexadecimal), which represents the following protection:

```
(S:RWED,O:RWED,G:RE,W:)
```

### 10.2.122 RMS_PROLOGUE Parameter

RMS_PROLOGUE specifies the default prologue VAX-11 RMS uses to create indexed files. The default value 0 specifies that VAX-11 RMS should determine the prologue based on characteristics of the file, 2 specifies Prologue 2 or Prologue 1, and 3 specifies Prologue 3. The VAX-11 RMS prologues are described in the VAX-11 Record Management Services Reference Manual.

### 10.2.123 SCSBUFFCNT Parameter

SCSBUFFCNT is not currently implemented.

### 10.2.124 SCSCONNCNT Parameter

SCSCONNCNT is the total number of SCS connections that are configured for use by all System Applications, including the one used by directory service listen.

If no computer interconnect device or UDA 50/52 is configured on your system, this parameter is ignored.

The default value is adequate for all CI/UDA hardware combinations available with VAX/VMS Version 3.0.

### 10.2.125 SCSFLOWCUSH Parameter

SCSFLOWCUSH is an SCS flow control parameter for sequenced messages. For each connection, SCS tracks the number of receive buffers available. SCS communicates the number of available receive buffers to the SCS at the remote end of the connection. However, SCS does not need to do this for each new receive buffer added. Instead, SCS notifies the remote SCS of new receive buffers if the number of receive buffers already communicated to the remote SCS falls as low as the value of SCSFLOWCUSH.

If no computer interconnect (CI) device is configured on your system, this parameter is ignored.

Normally the default value is adequate.

### 10.2.126 SCSMAXDG Parameter

SCSMAXDG is the maximum number of bytes of application data in one datagram.

If no computer interconnect device is configured on your system, this parameter is ignored.

Normally the default value is adequate. It should equal the value of the LRPSIZE parameter.

### 10.2.127 SCSMAXMSG Parameter

SCSMAXMSG is the maximum number of bytes of application data in one message.

If no computer interconnect device is configured on your system, this parameter is ignored.

Do not change the default value.

### 10.2.128 SCSRESPCNT Parameter

SCSRESPCNT is the total number of response descriptor table entries (RDTEs) configured for use by all System Applications.

If no computer interconnect (CI) device or UDA 50/52 is configured on your system, this parameter is ignored.

Normally the default value is adequate.

### 10.2.129 SCSSYSTEMID Parameter

SCSSYSTEMID is the unique identifier of each system within a cluster. It must be equal to the DECnet-VAX node number. For Version 3.0 of VAX/VMS, this means that although the system identifier is stored as a six-byte number, the user can specify only the low-order byte. The high-order five bytes will be zero.

If no computer interconnect (CI) device or UDA 50/52 is configured on your system, this parameter is ignored.

### 10.2.130 SETTIME Parameter

SETTIME enables or disables solicitation of the time of day each time the system is booted. This parameter should normally be off (0), so that the system sets the time of day at boot time to the value of the processor time-of-day register. You can reset the time after the system is up with the DCL command SET TIME (see the VAX/VMS Command Language User's Guide).

### 10.2.131 SPTREQ Parameter

SPTREQ sets the number of system page table (SPT) entries required for mapping the following components:

| Component |
|---|
| Executive image |
| VAX-11 RMS image |
| SYSMSG.EXE file |
| Multiport memory structures |
| Each MASSBUS adapter |
| Each UNIBUS adapter |
| Each DR32 adapter |

The number of system page table entries required for all other purposes is automatically computed and added to the value of SPTREQ to yield the actual size of the system page table.

Normally the default value is adequate.

### 10.2.132 SRPCOUNT Parameter

SRPCOUNT sets the number of preallocated small request packets. Each packet requires 96 bytes of permanently resident memory. If SRPCOUNT is too large, physical memory is wasted. If SRPCOUNT is too small, the system increases its value automatically, as needed, to permit the system to perform properly. However, the system cannot increase SRPCOUNT beyond the value of SRPCOUNTV. Furthermore, there is a minor physical memory penalty for allowing this growth. If SRPCOUNT is underconfigured, the penalty is 4 percent of physical memory from the configured value to the actual value on the running system.

You can use the DCL command SHOW MEMORY/POOL/FULL to determine SRPCOUNT usage.

### 10.2.133 SRPCOUNTV Parameter

SRPCOUNTV establishes the upper limit to which SRPCOUNT can be increased.

If this parameter is set too low, system performance can be adversely affected by preventing the system from using this memory allocation mechanism for nonpaged pool requests. If SRPCOUNTV is set too high, there is a penalty of 1 percent of physical memory for any unused growth space (one longword for every five unused small request packets).

Normally the computed, installed value is appropriate.

### 10.2.134 SWPFILCNT Parameter

SWPFILCNT defines the maximum number of swapping files that can be installed.

### 10.2.135 SWPOUTPGCNT Parameter

SWPOUTPGCNT defines the process size in pages (blocks) to which the swapper should attempt to reduce a process before attempting to swap it out. The pages taken from the process are placed into the paging system. This parameter allows the swapper an alternative mechanism before actually performing swaps.

Normally the default value is adequate.

### 10.2.136 SYSMWCNT Parameter

SYSMWCNT sets the quota for the size of the system working set, which contains the pageable portions of the system, the paged dynamic pool, VAX-11 RMS, and the resident portion of the system message file.

While a high value takes space away from user working sets, a low value may seriously impair system performance. Appropriate values vary depending on the level of system use. When the system is running at full load, check the rate of system faults with the MONITOR PAGE command of the Monitor Utility (see the VAX-11 Utilities Reference Manual). An average system page fault rate of between zero and three page faults per second is desirable. If the system page fault rate is high, and especially if the system seems to be slow, you should increase the value of SYSMWCNT. However, do not set this parameter so high that system page faulting never occurs.

### 10.2.137 TIMEPROMPTWAIT Parameter

TIMEPROMPTWAIT defines the number of seconds that a VAX-11 processor should wait for the time and date to be entered when a system boot occurs, if the processor's time-of-year clock does not contain a valid time. (The time unit of micro-fortnights is approximated as seconds in the implementation.) If the time specified by TIMEPROMPTWAIT elapses, the system continues the boot operation and the date and time are set to the last recorded time that the system booted. For a VAX-11/730, which does not have a battery back-up clock, the system time must be supplied whenever power fails.

NOTE

DIGITAL recommends that you set the system time correctly before allowing the system to run, so that all functions employing time-stamping (such as the operator log, the error log, accounting records, file creation dates, and file expiration dates) will contain correct time values.

If TIMEPROMPTWAIT is 0, no prompt or wait occurs; the system boots immediately, using the time of the last boot as the system time. If TIMEPROMPTWAIT is a positive number less than 32768, one prompt is issued and the value dictates how many seconds you can take to respond with a time. If you do not provide a time before TIMEPROMPTWAIT elapses, the system boots, using the time of the last boot as the system time. If TIMEPROMPTWAIT is a number in the range of 32768 through 65535, the prompt for the time is issued at intervals starting with 2 and doubling until 256 seconds is reached. If no response is received, the prompts restart, with the two-second interval. This prompting process repeats indefinitely, until you specify a time.

### 10.2.138 TTY_ALTALARM Parameter

TTY_ALTALARM sets the size of the alternate type-ahead buffer alarm. This value indicates at what point an XOFF should be sent to terminals that use the alternate type-ahead buffers with the size specified by the TTY_ALTYPAHD parameter.

### 10.2.139 TTY_ALTYPAHD Parameter

TTY_ALTYPAHD sets the size of the alternate type-ahead buffer. Use this parameter to allow the block mode terminals and communications lines to operate more efficiently.

### 10.2.140 TTY_BUF Parameter

TTY_BUF sets the default line width for terminals.

### 10.2.141 TTY_CLASSNAME

TTY_CLASSNAME provide the two-character prefix for the terminal class driver name that is required when booting. Changing the prefix can be useful when debugging a new terminal driver.

### 10.2.142 TTY_DEFCHAR Parameter

TTY_DEFCHAR sets the default characteristics for terminals, using a code derived by summing the following hexadecimal values:

| Characteristic | Value | Function |
|---|---|---|
| PASSALL[1] | 1 | Passall mode |
| NOECHO | 2 | Noecho mode |
| NOTYPEAHEAD[1] | 4 | No type-ahead buffer |
| ESCAPE | 8 | Escape sequence processing |
| HOSTSYNC | 10 | Host can send XON/XOFF |
| TTSYNC | 20 | Terminal can send XON/XOFF |
| LOWERCASE | 80 | Lowercase |
| TAB | 100 | Mechanical tabs |
| WRAP | 200 | Wraparound at end of line |
| CRFILL[1] | 400 | Perform carriage return fill |
| LFFILL[1] | 800 | Perform line feed fill |
| SCOPE | 1000 | Terminal is a scope |
| HOLD_SCREEN | 4000 | VT52 holdscreen feature |
| EIGHT_BIT | 8000 | Eight-bit terminal |
| MBXDSABL | 10000 | Disable mailbox |
| NOBROADCAST | 20000 | Prohibit broadcast |
| READSYNC | 40000 | XON/XOFF on reads |
| FORM | 80000 | Mechanical from feeds |
| HALFDUP | 100000 | Set for half-duplex operation |
| MODEM | 200000 | Set for modem signals |

1. Do not set this characteristic as the default in TTY_DEFCHAR

Where a condition is false, the value is 0.

The upper byte is the page length. The default characteristics are 24 lines per page, terminal synchronization, wraparound, lowercase, scope, and half-duplex.

### 10.2.143 TTY_DEFCHAR2 Parameter

TTY_DEFCHAR2 sets a second longword of default terminal characteristics. The default characteristics are represented as a code that is derived by summing the following hexadecimal values:

| Characteristic | Value | Function |
|---|---|---|
| LOCALECHO | 1 | Enable local echo terminal logic; use with the NOECHO characteristic in TTY_DEFCHAR |
| AUTOBAUD | 2 | Enable autobaud detection |
| HANGUP | 4 | Hang up on logout |
| MODHANGUP | 8 | Allow modification of HANGUP without privileges |
| BRDCSTMBX | 10 | Allow sending of broadcasts to mailboxes |
| XON | 20 | (No effect in this parameter) |
| DMA | 40 | (No effect in this parameter) |
| ALTYPEAHD | 80 | Use the alternate type-ahead parameters |
| SETSPEED | 100 | Clear to allow setting of speed without privileges |
| ANSI_CRT | 1000000 | Terminal conforms to ANSI CRT programming standards |
| REGIS | 2000000 | Terminal has REGIS CRT capabilities |
| BLOCK_MODE | 4000000 | Block mode terminal |
| ADVANCED_VIDEO | 8000000 | Terminal has advanced video |
| EDIT | 10000000 | Terminal has local edit capabilities |
| DECCRT | 20000000 | Terminal is a DIGITAL CRT |

The default is AUTOBAUD.

### 10.2.144 TTY_DIALTYPE Parameter

TTY_DIALTYPE provides flag bits for dialups. Bit 0 is 1 for United Kingdom dialups and 0 for all others. Bit 1 controls the modem protocol used. The remaining bits are reserved for future use. See the VAX/VMS I/O User's Guide for more information on the flag bits.

### 10.2.145 TTY_DMASIZE Parameter

TTY_DMASIZE specifies the number of characters in the output buffer below which character transfers are performed, and above which DMA transfers occur, provided the controller is capable of DMA I/O.

### 10.2.146 TTY_OWNER Parameter

TTY_OWNER specifies the owner UIC against which terminal protection is set. The specification must represent the value of a standard 32-bit UIC with the group number in the high-order word and the member number in the low-order word. You should normally set TTY_OWNER to a value of hexadecimal 10004, which is UIC [1,4].

### 10.2.147 TTY_PARITY Parameter

TTY_PARITY is not currently implemented.

### 10.2.148 TTY_PROT Parameter

TTY_PROT sets the default protection for all terminals in relation to the UIC specified for the TTY_OWNER parameter below. The specification must represent the value of a standard 16-bit protection mask as described in Chapter 3. However, only read bits are meaningful. When a read bit is on, that category of user is prohibited from allocating terminals.

The default (FFF0) provides for system access only on terminals. You can change protection on a per-terminal basis with the DCL command SET PROTECTION/DEVICE to permit user allocation of remote and application terminals.

Note that this protection does not prevent logging in on the terminal; it only prevents allocation of the terminal by another process.

### 10.2.149 TTY_RSPEED Parameter

TTY_RSPEED defines the receive speed for terminals. If TTY_RSPEED is 0, TTY_SPEED controls both the transmit and receive speed. This parameter is only applicable for controllers that support split speed operations, such as the DZ-32 and the DMF-32.

### 10.2.150 TTY_SCANDELTA Parameter

TTY_SCANDELTA sets the interval for polling terminals for dial-up and hang-up events. Shorter intervals use more processor time; longer intervals may result in missing a hang-up event.

### 10.2.151 TTY_SILOTIME Parameter

TTY_SILOTIME defines the interval at which the DMF-32 hardware polls the input silo for received characters. The DMF-32 asynchronous terminal controller can delay the generation of a single input interrupt until multiple characters have accumulated in the input silo. TTY_SILOTIME specifies the number of milliseconds that the characters are allowed to accumulate prior to the generation of an input interrupt by the hardware.

Normally the default value is adequate.

### 10.2.152 TTY_SPEED Parameter

TTY_SPEED sets the default speed for terminals, using the following codes:

| Code | Baud Rate | Code | Baud Rate |
|---|---|---|---|
| 1 | 50 | 9 | 1800 |
| 2 | 75 | 10 | 2000 |
| 3 | 110 | 11 | 2400 |
| 4 | 134.5 | 12 | 3600 |
| 5 | 150 | 13 | 4800 |
| 6 | 300 | 14 | 7200 |
| 7 | 600 | 15 | 9600 |
| 8 | 1200 | 16 | 19200 |

### 10.2.153 TTY_TYPAHDSZ Parameter

TTY_TYPAHDSZ sets the size of the terminal type-ahead buffer.

### 10.2.154 UAFALTERNATE Parameter

UAFALTERNATE enables or disables the assignment of SYSUAF as the logical name for SYSUAFALT, causing all references to the user authorization file (SYSUAF) to be translated to SYS$SYSTEM:SYSUAFALT. Use of the normal user authorization file (SYS$SYSTEM:SYSUAF) can be restored by deassigning the system logical name SYSUAF. This parameter should be on (1) only where the system is being used by a restricted set of users. You must create a user authorization file named SYSUAFALT prior to its use (see Chapter 2).

### 10.2.155 UDABURSTRATE Parameter

UDABURSTRATE is not currently implemented.

### 10.2.156 USERD1 Parameter

USERD1 is a dynamic parameter that is reserved for definition at the user's site. The reserved longword is referenced by the symbol SGN$GL_USERD1 in the module SYS$SYSTEM:SYS.STB.

### 10.2.157 USERD2 Parameter

USERD2 is a dynamic parameter that is reserved for definition at the user's site. The reserved longword is referenced by the symbol SGN$GL_USERD2 in the module SYS$SYSTEM:SYS.STB.

### 10.2.158 USER3 Parameter

USER3 is a parameter that is reserved for definition at the user's site. The reserved longword is referenced by the symbol SGN$GL_USER3 in the module SYS$SYSTEM:SYS.STB.

### 10.2.159 USER4 Parameter

USER4 is a parameter that is reserved for definition at the user's site. The reserved longword is referenced by the symbol SGN$GL_USER4.

### 10.2.160 VIRTUALPAGECNT Parameter

VIRTUALPAGECNT sets the maximum number of virtual pages that can be mapped for any one process. Every 128 virtual pages requires 4 bytes of permanently resident memory in the system page table (as discussed under the BALSETCNT parameter). A program is allowed to divide its virtual space between the P0 and P1 tables in any proportion except that the P1 table must be large enough to map 320 pages.

When the System Dump Analyzer is used, you must insure that the value of VIRTUALPAGECNT is at least the size of the dump file plus approximately 2000 pages.

Normally the computed, installed value is adequate.

### 10.2.161 WSDEC Parameter

WSDEC specifies the number of pages by which the limit of a working set is automatically decreased at each adjustment interval (which is quantum end). At a setting of 35, for example, the system will decrease the limit of a working set by 35 pages each time a decrease is required.

Increasing the value of this parameter tends to increase the speed with which working set limits are decreased when the need arises.

### 10.2.162 WSINC Parameter

WSINC specifies the number of pages by which the limit of a working set is automatically increased at each adjustment interval (which is quantum end). At a setting of 150, for example, the system will increase the limit of a working set by 150 pages each time an increase is required.

Decreasing the value of this parameter tends to reduce the speed with which working set limits are increased when the need arises. Normally, you should keep this parameter at a high value because a rapid increase in limit is often critical to performance.

A value of 0 for WSINC disables the automatic adjustment of working set limits for all processes. Limits stay at their base values. You can disable the automatic adjustment of working set limits on a per-process basis by using the DCL command SET WORKING_SET.

### 10.2.163 WSMAX Parameter

WSMAX sets the maximum number of pages on a system-wide basis for any working set. The value of WSMAX also affects the allocation of permanently resident memory for the swapper map and system page table (and for this reason should not be set at an arbitrarily high number):

- Swapper map -- 4 bytes for each page of WSMAX
- System page table -- 4 bytes for each 128 pages of WSMAX, times BALSETCNT

Generally, you should use a reasonable value for WSMAX; that is, whatever the size of the largest working set will need to be. The default value is appropriate for normal time-sharing operations, while significantly larger values should be used for programs with very large virtual address spaces to reduce page faulting.

### 10.2.164 XFMAXRATE Parameter

XFMAXRATE limits the data transfer rate that can be set for DR32 devices. On some hardware configurations (especially those without interleaved memory), a high DR32 transfer rate could cause a machine check (CPU timeout). The VAX/VMS I/O User's Guide describes how to encode this parameter.

# CHAPTER 11

# SYSTEM GENERATION

When VAX/VMS is installed for the first time or is upgraded from a previous version, the system adjusts system parameters for your system hardware. This is done by setting values in the system via the System Generation Utility (SYSGEN), and by creating several files on the system disk to be used in subsequent bootstrap operations. You can perform further configurational tuning of the system by editing the intermediate text files produced by the AUTOGEN.COM command procedure and reinvoking the procedure (see Chapter 12). You can also use SYSGEN to alter system parameters directly. This chapter describes the parts of VAX/VMS that you can manipulate with SYSGEN:

- System parameters -- Create and modify a standard system parameter file for use in subsequent conversational bootstrap operations; modify the parameter values in the current system image (SYS$SYSTEM:SYS.EXE) for use in subsequent bootstrap operations; dynamically modify the parameter values of the active system (applies only to the dynamic system parameters)

- Devices and device drivers -- Connect devices and load their device drivers (most of this work is automatic)

- System files -- Create additional paging and swapping files

- Start-up command procedure -- Identify the current site-independent start-up command procedure

- Multiport (shared) memory -- Initialize multiport memory units

See the VAX-11 Utilities Reference Manual for full details of the System Generation Utility and its commands.

## 11.1 SYSTEM PARAMETERS

The bootstrap process initializes the active system parameter values in memory from the current system image on disk (that is, the starting parameter values are those in SYS$SYSTEM:SYS.EXE). In a conversational bootstrap operation, you can modify these values by reinitializing the active parameter values from a parameter file or the default list, and by setting new parameter values on an individual basis. At the end of the bootstrap operation, the system image is modified to conform to the active parameter values.

WARNING

Many of the system generation parameters can affect other parameters or the performance of the system. It is suggested that you make changes to system parameters by editing the PARAMS.DAT file and reinvoking the AUTOGEN.COM procedure.

After the system is booted and running, you can run SYSGEN to create or modify parameter files, modify the current system image, and modify the dynamic parameter values of the active system. The following sequence illustrates a typical procedure for using SYSGEN:

- Invoke SYSGEN -- Invoking SYSGEN initializes a work area to the active parameter values.
- Optionally issue a USE command -- You can reinitialize the work area to the values of a parameter file, the current system image, or the default values, if the active values do not provide a suitable base for subsequent operations.
- Issue SET commands -- You modify parameters on an individual basis. These modifications have no effect outside the SYSGEN work area.
- Issue a WRITE command -- You create a parameter file, modify the current system image on disk, or modify the active system on disk.

During these operations, you can use the SHOW command to examine the parameter values in the SYSGEN work area.

### 11.1.1 Creating a Parameter File

The creation of a new parameter file does not immediately affect the system. At a subsequent conversational bootstrap operation, however, you can initialize the active system with the values of the new file. The following example creates a new version of the AUTOGEN.PAR system parameter file with a new value for the REALTIME_SPTS parameter:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN SYSGEN
SYSGEN>  USE AUTOGEN
SYSGEN>  SET REALTIME_SPTS 10
SYSGEN>  WRITE AUTOGEN
SYSGEN>  EXIT
```

The next example creates a user file named SYS$SYSTEM:OURSITE.PAR, using the AUTOGEN.PAR file as a base:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN SYSGEN
SYSGEN>  USE AUTOGEN
SYSGEN>  SET REALTIME_SPTS 10
SYSGEN>  WRITE OURSITE
SYSGEN>  EXIT
```

### 11.1.2 Modifying the System Image

The modification of the current system image also does not immediately affect the system. At subsequent bootstrap operations, however, the active system is initialized with the new values. A conversational bootstrap operation permits you to modify these values further, while a nonstop bootstrap operation makes the new values the values of the active system. The following example modifies the REALTIME_SPTS parameter value in the system image:

```
$     SYS$SYSTEM:SYSGEN
SYSGEN>  USE CURRENT
SYSGEN>  SET REALTIME_SPTS 10
SYSGEN>  WRITE CURRENT
%OPCOM, 25-JUN-1982 16:04:06.30, message from user SYSTEM
%SYSGEN-I-WRITECUR, CURRENT system parameters modified by process
ID 00160030 into file SYS.EXE
SYSGEN>  EXIT
```

### 11.1.3 Modifying the Active System

Modification of the active system immediately affects that subset of the system parameters called the dynamic parameters by changing their values in the active system in memory. Chapter 10 identifies the dynamic parameters (as does the SYSGEN command SHOW/DYNAMIC). The other parameters cannot be changed immediately because they regulate structures that cannot be changed once the system is running. The following example modifies the active value of the PFCDEFAULT parameter:

```
$     SYS$SYSTEM:SYSGEN
SYSGEN>  SET PFCDEFAULT 127
SYSGEN>  WRITE ACTIVE
%OPCOM, 25-JUN-1982 16:04:06.30, message from user SYSTEM
%SYSGEN-I-WRITEACT, ACTIVE system parameters modified by process
ID 00160030 into file SYS.EXE
SYSGEN>  EXIT
```

Modification of the active system does not affect the current system image on disk. If, for example, you set new active parameter values (WRITE ACTIVE) and later want to use these values for subsequent bootstrap operations, the values must be explicitly written to the current system image on disk:

```
$     SYS$SYSTEM:SYSGEN
SYSGEN>  WRITE CURRENT
%OPCOM, 25-JUN-1982 16:04:06.30, message from user SYSTEM
%SYSGEN-I-WRITECUR, CURRENT system parameters modified by process
ID 00160030 into file SYS.EXE
SYSGEN>  EXIT
```

## 11.2 DEVICES AND DEVICE DRIVERS

Normally, you issue the AUTOCONFIGURE command to automatically connect all devices physically attached to the system and load their device drivers, saving a great deal of effort and reducing the possibility of error. Devices not attached to the system and devices with nonstandard names can be connected and their device drivers loaded with explicit CONNECT (or CONNECT and LOAD) commands. Great care must be exercised in issuing CONNECT and LOAD commands; see the VAX/VMS Guide to Writing a Device Driver.

Devices not connected automatically by AUTOCONFIGURE include the network communications logical device and the console block storage device. Connecting the network communications logical device requires the following explicit CONNECT command:

```
CONNECT NET/NOADAPTER/DRIVER=NETDRIVER
```

Connecting the console block storage device requires the following explicit CONNECT command:

```
CONNECT CONSOLE
```

The following example autoconfigures the devices physically attached to the system and explicitly connects the network software device and the console block storage device:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN>  AUTOCONFIGURE ALL
SYSGEN>  CONNECT NET/NOADAPTER/DRIVER=NETDRIVER
SYSGEN>  CONNECT CONSOLE
SYSGEN>  EXIT
```

Normally, the SYSGEN commands for connecting devices and loading device drivers are included in the site-independent start-up command procedure (see Chapter 7).

Another DIGITAL-supplied driver named CONINTERR (which resides in SYS$SYSTEM:CONINTERR.EXE) permits real-time processes to connect to interrupt vectors for quick response to and special handling of real-time events. The driver is not associated with any one device type. See the VAX/VMS Real-Time User's Guide for further information.

## 11.3 SYSTEM FILES

The system defines appropriate sizes for the paging, swapping, and dump files for your hardware configuration. The full file specification of each file is SYS$SYSTEM:file-name.type. The file names are PAGEFILE.SYS for the paging file, SWAPFILE.SYS[1] for the swapping file, and SYSDUMP.DMP for the dump file. Sizes are expressed in pages.

The VAX/VMS software distribution kit creates system files suitable for most systems. For special workloads or variant configurations, you must specify the file sizes with the CREATE command of the System Generation Utility or (to create primary files only) with a DIGITAL command procedure called SYS$UPDATE:SWAPFILES.COM. The following example illustrates the use of the CREATE command:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN SYSGEN
SYSGEN>  CREATE PAGEFILE.SYS /SIZE=16384
%SYSGEN-I-EXTENDED, SYS$SYSROOT:[SYSEXE]PAGEFILE.SYS;1 extended
SYSGEN>  CREATE SWAPFILE.SYS /SIZE=7168
%SYSGEN-I-EXTENDED, SYS$SYSROOT:[SYSEXE]SWAPFILE.SYS;1 extended
SYSGEN>  CREATE SYSDUMP.DMP /SIZE=2052
%SYSGEN-I-EXTENDED, SYS$SYSROOT:[SYSEXE]SYSDUMP.DMP;1 extended
SYSGEN>  EXIT
```

---

1. On any configuration where the system disk space is less than 25,000 blocks, the swapping file is named SWAPFILE1.SYS.

The next example uses the command procedure:

```
$ @SYS$UPDATE:SWAPFILES
To leave a file size at its current value type a
carriage return in response to its size prompt.
Current file sizes are:

Directory SYS$SYSROOT:[SYSEXE]

PAGEFILE.SYS;1    16384
SYSDUMP.DMP;1      4128
SWAPFILE.SYS;1     3072

Total of 3 files, 23584 blocks.

There are 128741 available blocks on SYS$SYSDEVICE.

Enter new size for paging file: RET
Enter new size for system dump file: 6052
Enter new size for swapping file: RET

%SYSGEN-I-EXTENDED, SYS$SYSROOT:[SYSEXE]SYSDUMP.DMP;1 extended

*****************************************************************
*  Please reboot in order for the new files to be used by the system.  *
*  After rebooting, purge obsolete copies of the files.               *
*  DO NOT delete the old files until after the reboot.                *
*****************************************************************
```

Both the command procedure and the CREATE command automatically extend the size of a paging or dump file if you specify a size that is larger than that of an existing file. (However, if you have explicitly installed a file with the SYSGEN command INSTALL, the file cannot be extended; instead, a new version of the file is created.) If you specify a smaller size for any of these types of system file, the command procedure or the CREATE command creates a new version of the system file. If the swapping file does not exist, you are queried and you can request that it be created. In any case, an appropriate message is issued whenever a file is created or extended. If a new version was created, you must delete the old version explicitly (but not until after the next bootstrap operation). In the case of a primary file (PAGEFILE.SYS, SWAPFILE.SYS, or SYSDUMP.DMP), the new or extended size file does not become effective until the system is shut down and restarted. In the case of a secondary file, the new file becomes effective when it is installed.

You can verify the suggested sizes through the following calculations:

- Paging file -- Size of the average program at your site (in pages) times the maximum number of processes (MAXPROCESSCNT system parameter). The system installation sets an initial size for your primary paging file. You can display statistics on your paging file usage with the DCL command SHOW MEMORY. Examine the data pertaining to the paging files. Aim to keep paging file usage less than half the size of the paging files.

  You can limit the usage of paging file space by user programs with the /PGFLQUOTA qualifier to the ADD and MODIFY commands in the Authorize Utility (see the VAX-11 Utilities Reference Manual). You should not reduce the value of /PGFLQUOTA below 1024. Size requirements of the paging file can vary widely depending on user applications. Sufficient space in the paging file is critical to system performance. If a paging

file starts to fill to the point where system performance is being affected, a message will be printed on the console terminal. Should this happen, you should increase the size of your paging file.

- Dump file -- Size of physical memory in pages (to save the contents of memory if the system fails) plus four pages (to provide continuity of the error log when the system is shut down or if the system fails).

- Swapping file -- Maximum number of processes (MAXPROCESSCNT system parameter) times the average of the working set quotas of processes runnning on the system. The system installation sets an initial size for SWAPFILE.SYS based on your hardware configuration. As an alternative to trying to calculate a more accurate swapping file size, you can monitor the swapping file usage with the DCL command SHOW MEMORY and watch its usage under load. You should aim to keep at least one-fourth of the swapping file space unused; otherwise system performance can be severely affected.

At bootstrap time, the system activates the latest versions of SYS$SYSTEM:PAGEFILE.SYS, SWAPFILE.SYS, and SYSDUMP.DMP. After bootstrapping, you can replace or augment the primary paging or swapping file with additional files (previously created with the SYSGEN command CREATE) by issuing the SYSGEN command INSTALL. The advantages to using secondary files are that they do not have to be on the system disk and they can span volumes in a volume set. Where secondary files are used, you should include SYSGEN INSTALL commands for the secondary files in the site-specific start-up command procedure (see Chapter 7). If you are using secondary files because there is a shortage of disk space on the system disk, you can reduce the size of PAGEFILE.SYS to a value as low as 3000 blocks or SWAPFILE.SYS to a value as low as 1000 blocks.

All processes created after installation of additional paging files use the paging file with the most space, while all processes created before its installation continue to use the paging file to which they are assigned. Swapping space is allocated from whichever swapping file has space.

Installation of additional paging or swapping files requires nonpaged dynamic memory for a bit map, just as the primary files do.

## 11.4 START-UP COMMAND PROCEDURE

Chapter 7 describes the site-independent start-up command procedure named SYS$SYSTEM:STARTUP.COM that DIGITAL supplies and recommends that you do not edit it. However, if necessary, you can create alternate site-independent start-up command procedures; for example, you can copy STARTUP.COM to other files of type COM and edit those files.

Following a bootstrap operation, the system executes the current site-independent start-up command procedure. Initially (that is, as supplied in the software distribution kit), the current site-independent start-up command procedure is STARTUP.COM. You can name an alternate site-independent start-up command procedure to be used during subsequent bootstrap operations with the SET/STARTUP command in SYSGEN. The SHOW/STARTUP command displays the current

site-independent start-up command procedure. The following example displays the current site-independent start-up command procedure, then specifies an alternate:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN>  SHOW/STARTUP

 Startup command file = SYS$SYSTEM:STARTUP.COM
SYSGEN>  SET/STARTUP SYS$SYSTEM:XSTARTUP.COM
SYSGEN>  WRITE CURRENT
%OPCOM, 25-JUN-1982 16:04:06.30, message from user SYSTEM
%SYSGEN-I-WRITECUR, CURRENT system parameters modified by process
ID 00160030 into file SYS.EXE
SYSGEN>  EXIT
```

## 11.5 MULTIPORT (SHARED) MEMORY

A single processor can attach one or two multiport memory units, each of which may vary in size from 256KB to 2MB. The front panel of each multiport memory unit displays the number of that unit. When you issue a SHARE (initialize) command, SYSGEN polls the processors with ports on the specified multiport memory unit. If no other processors are using the unit, the unit is initialized. If another processor is using the unit, the unit is connected. A SHARE (connect) command for an uninitialized multiport memory unit results in an error condition.

You should power down a multiport memory unit only after first shutting down and rebooting all systems connected to the unit.

A multiport memory unit managed by VAX/VMS contains the following structures, with space requirements as indicated:

- Common data page -- Description of VAX/VMS data structure and quotas for the shared memory
- Global section description (GSD) table -- Total global sections, as specified in the SHARE command, times 100 bytes, rounded up to the next full page
- Mailbox table -- Total mailboxes, as specified in the SHARE command, times 48 bytes, rounded up to the next full page
- CEF table -- Total common event flag clusters, as specified in the SHARE command, times 80 bytes, rounded up to the next full page
- PRQ pool -- Total interprocess or request messages, as specified in the SHARE command, times 64 bytes, rounded up to the next full page
- Dynamic pool -- Number of blocks allocated to the pool, as specified in the SHARE command, times the size of each block, as specified in the SHARE command
- Global section bit map -- One page
- Global sections -- Size of multiport memory unit minus the sum of the above preallocated structures (that is, the remaining space)

Figure 11-1 illustrates the appearance of a multiport memory unit where the SHARE command specifies the default structure values, assuming a 256KB unit with four active ports.

```
SYSGEN>  SHARE MPM0 SHR_MEM_1/INITIALIZE-
SYSGEN>  /GBLSECTIONS=32/MAILBOXES=32-
SYSGEN>  /CEFCLUSTERS=32/POOLBCOUNT=128-
SYSGEN>  /POOLBSIZE=128/PRQCOUNT=64
```

| | | | |
|---|---|---|---|
| COMMON DATA PAGE | 1 | = | 1 page |
| PRQ POOL | 64 x 64 | = | 8 pages |
| DYNAMIC POOL | 128 x 128 | = | 32 pages |
| GSD TABLE | 32 x 100 | = | 7 pages |
| MAILBOX TABLE | 32 x 48 | = | 3 pages |
| CEF TABLE | 32 x 80 | = | 5 pages |
| GLOBAL SECTION BIT MAP | | = | 1 page |
| GLOBAL SECTIONS | 512 - 57 | = | 455 pages |

← relative page 1

**Figure 11-1: Example of Multiport Memory Structures**

The following guidelines are suggested in selecting values for the SHARE qualifiers that regulate the sizes of the preallocated structures:

- /CEFCLUSTERS, /GBLSECTIONS, and /MAILBOXES -- You should simply specify the maximum number of each type of structure required by all processors at any one time. The same structure being used by many processes on one or more processors counts as just one structure.

- /POOLBCOUNT and /POOLBSIZE -- The primary use of the dynamic pool is to buffer mailbox messages. The size of a message is 28 bytes plus the data in the message. Since space from the pool is always allocated in whole blocks, the recommended block size is the median message size plus 28. A block size that is too small for a message requires extra system overhead to concatenate the message blocks into the user buffer and segment them out of the user buffer. The number of blocks should be sufficient to satisfy all messages that might be outstanding at once. If a mailbox request cannot be satisfied due to insufficient pool space, the requesting process enters a resource wait state or the request fails (if resource wait mode is not enabled), just as if the nonpaged dynamic pool were depleted. For this reason, you should tend to overestimate space requirements in the dynamic pool.

- /PRQCOUNT -- The system uses interprocessor request blocks internally to transfer requests among the VAX/VMS executive routines and mailbox drivers on the different processors. PRQs are allocated and deallocated rapidly, so that a large number should not be needed. The default value normally suffices. If an executive or driver request cannot be satisfied due to depletion of the PRQs, the requesting routine waits until a PRQ becomes available.

You should calculate the space remaining for the global sections and determine if it is sufficient. If the space is insufficient, you might reduce the size of the dynamic pool. However, this condition really suggests the need for a larger or additional multiport memory unit.

Where a multiport memory unit is a normal part of the system configuration, you should include the SYSGEN commands to initialize and connect it in the site-specific start-up command procedure (see Chapter 7).

# CHAPTER 12

# TUNING CONSIDERATIONS

Hardware resources -- mainly physical memory and secondary storage -- constitute the primary constraint on system performance. Adequate hardware resources for the workload generally provide adequate performance with little need for tuning. Inadequate hardware resources for the workload generally provide inadequate performance regardless of the tuning effort. Only in the middle ground of just-adequate or just-inadequate resources does tuning become a significant factor. This resource level normally occurs in situations where the user is trying to make do with a small system, or where the user's workload has been increasing over a period of time with no corresponding addition of resources. For practical purposes, tunable general-purpose systems can be considered in two categories:

- Small systems -- Small in this context means a system with tight resources. Normally, this would be a system of 2MB or less, although conceivably a large system with an enormous workload might fit in this category. The primary emphasis in tuning a small system is in optimizing the use of physical memory and achieving a balance between the use of physical memory and disk I/O.

- Large systems -- Large in this context means a system with plentiful resources for the workload. Normally, this would be a system with physical memory in excess of 3MB, although conceivably a small system with a light workload might fit in this category. The primary emphasis in tuning a large system is on decreasing disk I/O.

Before undertaking a major tuning effort, the manager of a small system should weigh the time and effort involved in the venture against the alternative solution: the purchase of additional physical memory or faster disks.

## 12.1 PRETUNING CONSIDERATIONS

Before starting the tuning effort, you should ensure that hardware resources are adequate for the workload, that the workload is distributed as uniformly as possible, and that frequently used code is shared.

### 12.1.1 Hardware Resources

VAX-11 hardware configurations are designed to handle the following workloads:

| Workload | Memory |
|---|---|
| 1-4 users | .5MB |
| 2-12 users | 1MB |
| 8-32 users | 2MB |
| 24-48 users | 3MB |
| 32-64 users | 4MB |

These figures are approximate. The precise number of users for a memory configuration depends on the size and makeup of the programs those users are running. If the processing requirements of all users consist of editing, compiling, linking, and running small programs, the system will probably support around the maximum number of users -- 32 on the 2MB system, for example. If, on the other hand, processing consists mainly of developing very large programs and/or sorting, the system will probably support around the minimum number of users. For user production programs, you can lean toward the maximum figure for programs that contain little data or use VAX-11 RMS to manipulate data on a per-record basis, and toward the minimum figure for programs that contain very large data structures. The trend for native VAX/VMS software products (assemblers, compilers, and so on) is to require larger amounts of memory, mainly through the use of larger data structures to reduce I/O activity (resulting in shorter response times and faster throughput).

System throughput is limited primarily by three factors: physical memory, processor speed, and disk speed. A properly tuned system balances the use of these resources so that they reach the saturation point at the same time. Disk I/O operations, for example, can be reduced by using more physical memory for the file system caches. Such an adjustment makes sense on a system with abundant physical memory, but will probably decrease throughput on a small system (because the use of physical memory reaches the saturation point while the disks can still handle more I/O transfers).

Actual disk capacity is a function of user file requirements. However, the system should ideally contain at least two disk drives and two disk drive controllers. You should make an effort to place the disk drives containing the execution files (system and user image files, paging file, swapping file, and dump file) and the disk drives containing user data on separate controllers. This procedure will cut down on the contention between system I/O and user I/O activities.

### 12.1.2 Workload Distribution

You should distribute the workload as evenly as possible over the time the computer is running. While scheduling interactive users evenly is normally not possible due to the weight of convention on standard working and sleeping hours, either or both of the following techniques are workable:

1. Run large jobs as batch jobs -- Force the submission of large jobs on a batch basis; regulate the number of batch streams

so that batch usage is high when interactive usage is low. For example, the Accounting Utility should be run as a batch job. Chapter 8 discusses batch jobs.

2. Restrict system use -- Do not permit more users to log in at one time than the system can support with an adequate response time. You can restrict the number of interactive users with the IJOBLIM and RJOBLIM system parameters (see Chapter 10) and the DCL command SET LOGINS (see the VAX/VMS Command Language User's Guide). You might also restrict use of the system by groups of users to certain days and hours of the day. See the Authorize Utility in Chapter 2.

### 12.1.3 Sharing of Code

The site-independent start-up command procedure creates permanent global sections for system programs and subroutines by installing them as known images with the shared attribute. You should add to these (in the site-specific start-up command procedure) user programs and subroutines that have reached a production status or are in general use. Users should, of course, be encouraged to write shareable code. Chapter 7 of this manual covers start-up command procedures, while Chapter 6 discusses installing known images.

## 12.2 SYSTEM PERFORMANCE (BACKGROUND DISCUSSION)

Figure 12-1 illustrates the physical memory and secondary storage requirements for supporting process on a VAX/VMS system.

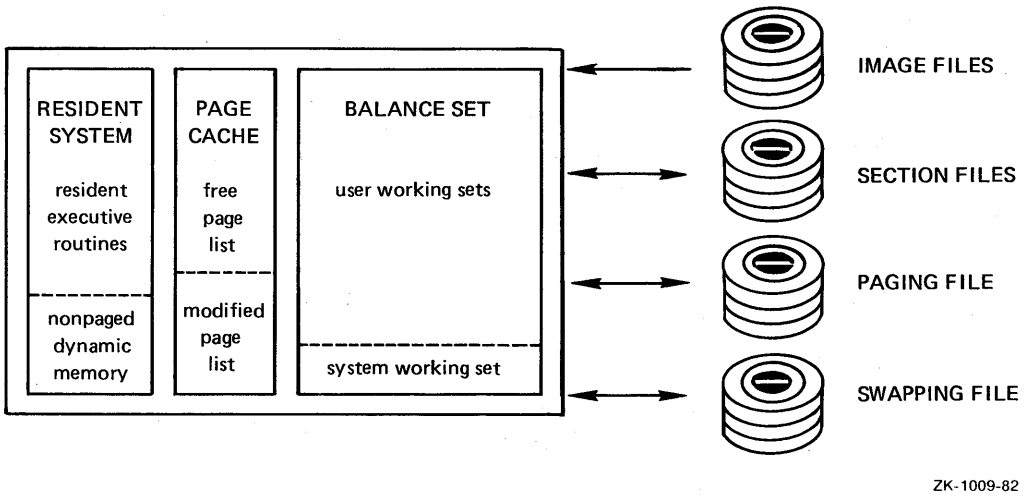



**Figure 12-1: VAX/VMS Memory Configuration**

Each active process is associated with a working set that contains the pages of code and data being used in the execution of that process. Initially, the working set expands to accommodate clusters of image and section pages (that is, code or data explicitly mapped by a process to its virtual address space) from their disk files as they are referenced. The size of the working set (unless altered by a system service) equals the cumulative number of pages brought in from the image and section files, until the working set limit is reached.

At that point, newly referenced pages are accommodated by pushing current pages out of the working set, on a first-in, first-out (FIFO) basis.

Whenever a referenced page does not exist in the working set, a page fault is said to occur. The system automatically adjusts the working set limit by deducting pages if the rate of page faults is low, and adding pages if the rate of page faults is high (unless you turn off the automatic working set adjustment feature).

If sufficient space exists, working sets reside in real memory. As memory fills up, the system attempts to make room for new working sets by removing pages from existing working sets. If sufficient space does not exist for all the working sets after they are reduced in size as much as possible, one or more of the working sets are moved from memory to the swapping file. The system determines which working sets to retain in physical memory and which to swap out to disk according to each process's state and priority, and the quantum:

- Current state -- The currently executing process resides in physical memory.
- Compute state -- Processes that can execute immediately if given processor time should reside in physical memory; they are the last candidates for outswapping.
- Waiting state -- Processes that are waiting for a local event flag, are hibernating, or are suspended constitute the primary candidates for outswapping.
- Priority -- Among processes in the same state, those with the lowest priorities are outswapped.
- Quantum -- A process in a compute state cannot be outswapped until it has received services for at least one SYSGEN-specified period of time called a quantum.

The system reevaluates physical memory versus disk residency requirements whenever a process's state changes. For example, if a local event flag is posted changing the state of a swapped-out process from wait to compute, the system will (if possible) place the process's working set in physical memory, even if this means taking pages from or outswapping the working set of another process.

One nonswappable working set is reserved for those portions of the system executive that are pageable. In addition, the system requires user working sets (these are swappable) for the system processes ERRFMT, OPCOM, JOB_CONTROL, and the file system ACP. Working sets are also required for any magnetic tape ACPs and print symbionts in use. DECnet uses working sets for NETACP, REMACP (optional), and EVL (optional).

Pages pushed out of a working set that has reached its limit initially remain in physical memory by going to the page cache: to the free page list if they were not written on, or to the modified page list if they were written on. The free page list expands to accommodate additional pages being pushed out of working sets. The free page list contracts as pages are allocated to processes requesting additional memory; pages leave the free list on a FIFO basis. However, the system cannot reduce the size of the free page list below a SYSGEN-specified lower limit.

The modified page list expands to accommodate additional pages being pushed out of working sets, but only to a SYSGEN-specified upper

limit. At that point, pages are written back to their section files or (in the case of demand-zero and copy-on-reference pages) to the paging file, then moved to the free list, until the size of the modified page list drops below a SYSGEN-specified lower limit.

If a page that was pushed out of a working set is referred to again, a page fault occurs. The system must return the page to the working set from either:

- The page cache -- If the page is still in physical memory (that is, on the free page list or the modified page list), a fast page fault occurs, as no disk I/O is required.
- An image, section, or paging file -- If the page has been overwritten, a slower page fault occurs, as the page must be reread from disk.

Although pages move through the working sets and page cache on a FIFO basis, the return of referred-to-again pages from the page cache to the working sets produces a net effect of removing from memory the least frequently used pages and retaining the most frequently used pages.

When an image terminates, all section pages that were modified are written back to the section files. All pages associated with the program in physical memory are released. (Shared pages, however, are not released if currently in use by another process.)

## 12.3 TUNING TOOLS

You control system performance overall with the system parameters, and on a per-user basis with the entries in the UAF. Certain DCL commands also aid in evaluating system performance. The effects of tuning can be monitored with the Monitor Utility (MONITOR) and the DCL command SHOW MEMORY.

### 12.3.1 System Parameters and Files

When your system is installed or upgraded, a DIGITAL-supplied command procedure named SYS$UPDATE:AUTOGEN.COM sets the values of system parameters, the sizes of the paging, swapping, and dump files, and the contents of the default installed image list (VMSIMAGES.DAT in SYS$MANAGER), based on your hardware configuration and estimates collected from typical workloads. You can invoke AUTOGEN any time after installation or upgrade to reset system parameters and system file sizes for the next time the system is booted. Depending on the adjustments you are making, you may have to invoke AUTOGEN several times, specifying different parameters. The format for invoking AUTOGEN is as follows:

```
@SYS$UPDATE:AUTOGEN [parameter]
```

You can enter one parameter value. Table 12-1 lists the possible parameter values and their effects.

You should use AUTOGEN from the system manager's account, as you need the associated privileges.

Table 12-1: AUTOGEN Parameter Values

| Parameter | Function |
|---|---|
| None | Updates the current system parameters and files, and creates a parameter file named SYS$SYSTEM:AUTOGEN.PAR using the results of a previous AUTOGEN GENPAR operation |
| CONFIG | Defines your hardware configuration in a file named SYS$SYSTEM:CONFIG.DAT; Figure 12-2 demonstrates the appearance of a CONFIG.DAT file |
| GENPAR | Generates parameter and files requirements in a file named SYS$SYSTEM:PARAMS.DAT using the results of a previous AUTOGEN CONFIG step; Figure 12-2 demonstrates the appearance of a PARAMS.DAT file |
| REBOOT | Updates the system parameters and files and reboots the system |
| SAVE | Saves old parameter values prior to a GENPAR operation |
| SHUTDOWN | Updates the system parameters and files and shuts down the system |

**CONFIG.DAT**

```
VERSION:=V3.0
CPUTYPE=1
SID=19923201
MEMSIZE=8192
SYSDISK=4
NUMTERMS=80
NUMTAPES=2
NUMDISKS=23
NUMCOMMS=12
NUMPRINTERS=2
NUMDEVICES=122
```

**PARAMS.DAT**

```
VERSION:=V3.0
CPUTYPE=1
SID=19923201
MEMSIZE=8192
SYSDISK=4
NUMTERMS=80
NUMTAPES=2
NUMDISKS=23
NUMCOMMS=12
NUMPRINTERS=2
NUMDEVICES=122
MAXPROCESSCNT=110
VIRTUALPAGECNT=18432
OLD_GBLSECTIONS=124
OLD_GBLPAGES=5192
OLD_WSMAX=3072
SCSSYSTEMID=160
MAXPRINTSYMB=32
ACP_SWAPFLGS=14
```

**Figure 12-2: Sample CONFIG.DAT and PARAMS.DAT Files**

If (based on the performance of your system and from the suggestions in this chapter) you decide to modify some system parameters or change the size of a system file, you should perform such modifications with AUTOGEN rather than with the SYSGEN Utility, as AUTOGEN analyzes your modifications and adjusts any related parameters. The steps for using AUTOGEN to change system parameter values and system file sizes are as follows:

1. Edit SYS$SYSTEM:PARAMS.DAT -- The system parameters appear as the full name of the parameter followed by an equal sign and the value of the parameter. The system files appear as the keywords PAGEFILE, SWAPFILE, and DUMPFILE followed by an equal sign and the size of the file in blocks. You can change the values of those parameters and files listed in PARAMS.DAT, add new values, or delete existing entries. Be exact in spelling the names of the parameters and files.

2. Invoke SYS$MANAGER:AUTOGEN.COM -- Invoke AUTOGEN in one of the following ways:

   - @SYS$UPDATE:AUTOGEN -- Creates a new AUTOGEN.PAR file and updates the current system parameters and files
   - @SYS$UPDATE:AUTOGEN REBOOT -- In addition, reboots the system using the default bootstrap procedure
   - @SYS$UPDATE:AUTOGEN SHUTDOWN -- In addition, shuts down the system

   A reboot is necessary to apply the new parameter and file values to the running system.

You may also want to adjust the parameter and file values if you change the hardware configuration of your system (for example, add more physical memory) or transport the operating system from one hardware configuration to another. If you are moving the system to another hardware configuration, you should perform the procedures that follow on the target configuration:

1. @SYS$UPDATE:AUTOGEN CONFIG -- Redefines the hardware configuration in a new CONFIG.DAT file
2. @SYS$UPDATE:AUTOGEN GENPAR -- Generates a new PARAMS.DAT file, based on the new configuration (at this point, you can edit any special requirements into the PARAMS.DAT file)
3. @SYS$UPDATE:AUTOGEN [REBOOT|SHUTDOWN] -- Creates a new AUTOGEN.PAR file and updates the current system parameters and files, optionally rebooting or shutting down the system

### 12.3.2 User Authorization File

You can affect performance on a per-user basis by resetting the following UAF entries:

- WSEXTENT -- Maximum working set for the user's process; cannot exceed the system parameter WSMAX
- WSQUOTA -- Maximum guaranteed working set for the user's processes; cannot exceed WSEXTENT
- WSDEFAULT -- Working set limit for the user's processes; cannot exceed WSQUOTA
- PRIORITY -- Base priority for the user's processes
- CPUTIME -- CPU time permitted the user's processes

Chapter 2 and the description of the Authorize Utility in the VAX-11 Utilities Reference Manual explain these and the other UAF entries.

### 12.3.3 DCL Commands

The following DCL commands (which are described in the VAX/VMS Command Language User's Guide) affect system performance:

| Commands | Function |
|---|---|
| SET WORKING_SET/EXTENT<br>RUN /EXTENT | Reset working set extent for the user's processes; cannot exceed UAF extent |
| SET WORKING_SET/QUOTA<br>RUN /MAXIMUM_WORKING_SET | Reset working set quota for the user's processes; cannot exceed UAF quota |
| SET WORKING_SET/LIMIT<br>RUN /WORKING_SET | Reset the working set default for the user's processes; cannot exceed UAF quota |
| INITIALIZE/QUEUE/BATCH/WSEXTENT[1]<br>START/QUEUE/BATCH/WSEXTENT[1]<br>SUBMIT/WSEXTENT<br>$JOB/WSEXTENT | Set working set extent for jobs submitted to a batch queue |
| INITIALIZE/QUEUE/BATCH/WSQUOTA[1]<br>START/QUEUE/BATCH/WSQUOTA[1]<br>SUBMIT/WSQUOTA<br>$JOB/WSQUOTA | Set working set quota for jobs submitted to a batch queue |
| INITIALIZE/QUEUE/BATCH/WSDEFAULT[1]<br>START/QUEUE/BATCH/WSDEFAULT[1]<br>SUBMIT/WSDEFAULT<br>$JOB/WSDEFAULT | Set working set limit for jobs submitted to a batch queue |
| INITIALIZE/QUEUE/BATCH/CPUMAXIMUM /CPUDEFAULT[1]<br>START/QUEUE/BATCH/CPUMAXIMUM /CPUDEFAULT[1]<br>SUBMIT/CPUTIME<br>$JOB/CPUTIME | Set the CPU time permitted batch jobs |
| SET PROCESS/[NO]SWAPPING | Enables or disables the swapping in and out of memory of a process's working set |
| SET WORKING_SET/[NO]ADJUST | Enables or disables the adjustment of a working set's size |
| SET/PRIORITY<br>RUN/PRIORITY | Reset the base priority of a process |
| SET RMS_DEFAULT/BLOCK_COUNT | Resets the blocking factor for VAX-11 RMS I/O operations |

1. These commands require the operator user privilege (OPER).

### 12.3.4 Monitor Utility

With the Monitor Utility (MONITOR), you can monitor activities indicative of system performance. Useful displays in this regard include:

- PROCESSES (processes display) -- For each current process, lists the current state and priority of the process, the physical memory being used in terms of shareable pages and total pages, the cumulative direct I/O operations, the cumulative page faults, and the cumulative CPU time

- IO (I/O system rates) -- Provides the sizes of the free and modified page lists, plus the following information, expressed in cumulative units per interval, per second during the preceding interval, and per second since the start of the display: direct I/O operations, buffered I/O operations, page faults, pages read, read I/O operations, pages written, write I/O operations, and total pages inswapped

- PAGE (page management statistics) -- Substantially the same as I/O SYSTEM RATES, plus the page faults accumulated by the system working set

- MODES (time in processor modes) -- Percentage of time spent in each processor mode, plus idle time

- POOL (nonpaged pool statistics) -- Amount of unused space in the nonpaged dynamic pool

- STATES (number of processes in each scheduler state) -- Provides a snapshot of the state of the system

The VAX-11 Utilities Reference Manual provides operating instructions and detailed descriptions for the Monitor Utility, and examples of its use.

The following DCL commands complement the Monitor Utility:

- SHOW MEMORY -- Displays a synopsis of physical memory usage, the number of processes and how many are swapped out, pool usage, paging and swapping file usage, and the physical size of the system

- SHOW MEMORY/POOL/FULL -- Displays detailed information about the nonpaged dynamic pool, the amount of growth space used, and the amount of physical memory used

- SHOW SYSTEM -- Displays information on the processes currently executing, including state, priority, elapsed CPU time, number of page faults, and physical memory occupied

- SHOW PROCESS/ACCOUNTING -- Displays information about the executing process, including the cumulative number of page faults, the peak working set size, the peak virtual size, elapsed total time, and elapsed CPU time

- SHOW STATUS -- Displays information about the executing process, including the working set limit, the amount of physical memory being used, the elapsed CPU time, and the number of page faults

- SHOW WORKING_SET -- Displays a process's current limit and quota, and authorized quota

- LOGOUT/FULL -- Displays information about the terminating process, including the number of page faults, the peak working set size, the peak virtual size, elapsed total time, and elapsed CPU time

The VAX/VMS Command Language User's Guide provides full explanations of these commands.

## 12.4 TUNING SMALL SYSTEMS

As the background discussion indicates, the factors affecting efficient performance are interrelated and somewhat complex. The major factors for small systems include working set size, page cache characteristics, priority, quantum, compute swapping rate, and system use of the system.

### 12.4.1 Working Set Size

Decreasing working set limits increases the number of working sets that can occupy real memory at one time, reducing the need for swapping, but increasing the need (in most cases) for paging. Normally, excessive paging is preferable to excessive swapping for the following reasons:

- Swapping involves the output and input of large numbers of pages.
- Paging does not necessarily involve disk I/O operations due to the page cache. In fact, the ratio of disk I/O operations to page faults may be quite low.

A process can, then, show larger and larger numbers of page faults with little effect upon performance. Nevertheless, a point does occur where the reduction of the working set limit significantly degrades a process's performance. The optimal working set limit, then, would be just above the point where performance drops sharply. A two-phase strategy is recommended for setting working sets at their optimal sizes:

1. Figure initial working set limits for different types of processing on a rule-of-thumb basis.
2. Adjust working set limits based on observed behavior.

**12.4.1.1 Initial Working Set Limits** - For processing involving system components, the following working set limits are suggested:

- Small (60 to 200 pages) -- For editing, and for compiling and linking small programs ("typical" interactive processing)
- Large (200 to 500 pages) -- For compiling and linking large programs, and for sorting ("typical" batch processing)

Working set limits for user programs depend on the code-to-data ratio of the program and the amount of data in the program. Programs that are mostly code -- that include a small or moderate amount of data, or use VAX-11 RMS to process data on a per-record basis -- should require only a small working set. The amount of code should not matter as long as it is reasonably linear. Programs that manipulate large amounts of data internally require large working sets.

The following guidelines are suggested for initially setting working set limits:

- System generation parameters -- Set WSMAX at the highest number of pages required by any program.

- UAF options -- For each user, set WSQUOTA at the number of pages required to achieve acceptable page faulting for interactive work. Set WSEXTENT to the number of pages required to achieve acceptable page faulting for batch work. Set WSDEFAULT at the median number of pages required by a program that the user will run interactively.

- Batch queues -- For each batch queue, set WSQUOTA (using the DCL commands INITIALIZE/QUEUE or START/QUEUE) at the largest number of pages required by a job that users will submit. Set WSDEFAULT at the median number of pages required by a job that users will submit.

This arrangement effectively forces users to submit large jobs for batch processing, as the jobs will not run efficiently when entered interactively. You can further restrict the user who attempts to run a large job interactively by imposing CPU time limits in the UAF.

**12.4.1.2 Adjustments** - The system automatically adjusts working set limits based on observed behavior and the values of the PFRATL, PFRATH, WSINC, WSDEC, AWSMIN, and AWSTIME system parameters.

You can monitor system performance and change system parameter values in the following instances:

- Initial limit -- If the system typically increases or decreases a working set by a significant amount immediately after its process starts, the default working set limit for affected users and batch streams should be changed accordingly. For example, if a user's working set limit is 100 pages, but the user's processes typically show a working set limit of 200 for the first few minutes of processing, the user's default working set value in the UAF should be increased to 200.

- Excessive paging I/O activity -- If paging appears to be generating excessive I/O activity, you might decrease the value of PFRATH to make the working set limit more sensitive to paging activity. Additionally or alternatively, you might decrease the value of PFRATL, increase the value of WSINC, or decrease the value of WSDEC.

- Excessive swapping I/O activity -- If swapping I/O operations appear high, you might increase the value of PFRATH to make the working set less sensitive to paging activity. Working sets will tend to remain at smaller sizes, perhaps increasing the number of working sets that will fit in the balance set. Additionally or alternatively, you might increase the value of PFRATL, decrease the value of WSINC, or increase the value of WSDEC.

Monitoring aids for adjusting working set limits include the following Monitor Utility operations:

- PROCESSES Display -- The size column (which displays shareable/total pages being used by each process) indicates

what portion of its working set each process requires. The state column pinpoints processes having scheduling problems: a repeated state of PAGE FAULT WAIT, or PFW, may mean that the working set is too small; repeated states of COMPUTE (OUT OF BALANCE SET), or COMO, by several processes usually means that there are too many users for the available resources or that their working sets are too large. See the description of MONITOR PROCESSES in the Monitor Utility chapter of the VAX-11 Utilities Reference Manual.

- IO and PAGE displays -- As working set sizes are decreased, the swapping rate should decrease. As working set sizes are increased, the swapping rate may increase. A very high swapping rate is caused by too many users for available resources or too low a value for the BALSETCNT system parameter. Use the DCL command SHOW MEMORY to ensure that free process entry slots do not exist when no free balance set slots exist. This condition forces swapping and can be remedied by increasing the number of balance set slots. If the number of balance set slots is adequate, the problem is available resources (primarily physical memory). See the description of MONITOR IO in the Monitor Utility chapter of the VAX-11 Utilities Reference Manual.

Setting the system parameter WSINC to a value of 0 inhibits automatic working set adjustments for all users.

### 12.4.2 Page Cache

The page cache, consisting of the free page list and modified page list, recirculates pages pushed out of the working set when they are needed again. Frequently used pages tend to stay in physical memory -- either in the working sets (unless outswapped) or in the page cache -- while less frequently used pages tend to be removed from memory. Size requirements for the free page list depend mainly on the amount of code being pushed out of the working set. Size requirements for the modified page list depend mainly on the amount of data being pushed out of the working sets. The cluster size for pages being read from disk to the working sets also affects the efficiency of the page cache.

**12.4.2.1 Cache Sizes** - The FREELIM system parameter sets the minimum size of the free page list. The list may expand as space permits. The FREELIM value set by AUTOGEN is normally adequate. A high percentage of page faults resulting in read I/O operations (IO and PAGE displays) indicates that the value of FREELIM might be increased, while a small percentage indicates that the value of FREELIM is correct or might be smaller. If your system employs swapping, a high swapping rate (IO and PAGE displays) indicates that the value of FREELIM might be reduced to give more space to the balance set. (However, paging and swapping I/O might be better regulated by adjusting working set limits; see Section 12.4.1.2 above.)

The following system parameters regulate the size of the modified page list:

- MPW_HILIMIT -- Largest size the modified page list is permitted to reach before pages are written to disk

- MPW_LOLIMIT -- Point below which writing of pages stops
- MPW_WRTCLUSTER -- Number of pages the system is permitted to write in one I/O operation

The parameter values set by AUTOGEN are normally adequate. You can monitor the effects of the the size of the modified page list by watching the write I/O operation and swapping rates on the MONITOR IO display. Increasing the upper and lower limits on the modified page list normally lowers the write I/O operation rate, but may increase the swapping rate. You can raise the limits in small increments to determine whether the changes significantly improve the write I/O operation rate without significantly degrading the swapping rate. If you increase MPW_HILIMIT, adjust MPW_WAITLIMIT appropriately (see Chapter 10).

You should not assign arbitrarily high values to FREELIM and MPW_LOLIMIT, as these values reduce the fluid pages in physical memory. The system calculates fluid pages by subtracting the resident system, the system working set, and the minimum size of the page cache from available physical memory. If the value you specify for WSMAX exceeds the number of fluid pages, the value is decreased and automatic nonpaged pool growth is disabled.

**12.4.2.2 Cluster Size** - The system parameter PFCDEFAULT controls the number of pages being read from disk to the working sets during one I/O operation. The values set by AUTOGEN are normally adequate. As a rule of thumb, you can specify a value of 32 where the median working set size is 100 pages, and a value of 64 or more where the median working set size exceeds 200 pages.

A potential problem with a large cluster size is that the pages being transferred into a working set that has reached its limit push out an equal number of pages, causing large turnovers of pages in the system. However, the actual cluster size during any one I/O operation is limited by other factors, such as the size of the image or section being transferred. Only in a few instances do large cluster sizes adversely affect system performance, but these instances are more than made up for by the savings in disk I/O that the larger cluster sizes promote.

For special cases, users can adjust cluster sizes on an individual basis with the PFC parameter of the CLUSTER option in the linker (for pages being read from image and section files, not the paging file).

### 12.4.3 Priority, Quantum, and Compute-bound Swapping Rate

Certain workloads may give processes a disproportionate share of service. The problem can be alleviated somewhat by adjustments to base priorities, the quantum, and the swapping rate for compute-bound processes.

**12.4.3.1 Very Large Working Sets** - Running several working sets that are very large for the amount of physical memory often causes serious performance degradation. An example might be two batch streams using (on the average) 350-page working sets in a 512KB system. Both batch streams cannot fit in physical memory at the same time, requiring an inswap and outswap of 350 pages each time one or the other is granted

control. The high swapping rate not only slows batch throughput, but increases interactive response time, as these users are also held up by the contention on the system disk.

When this situation exists, you normally notice a high swapping rate (MONITOR IO display).

The obvious solution to the above problem is to run one batch stream. However, if two batch streams are essential -- or perhaps the problem revolves around two very large production programs that must run concurrently -- you can improve the situation with one of the following methods:

- Priority -- Decreasing the base priorities for the batch streams from 4 to 3 can result in a better interactive response time, but will slow batch throughput even more.

- Quantum -- Increasing the quantum in this situation normally ensures better service all around. You may find that a rather large quantum -- 1, 2, or even 3 seconds -- is in order. The larger quantum means that the large working sets will be swapped less frequently. While interactive users must sometimes wait in blocks of several seconds to be serviced, this may be an improvement over the previous response time.

- UAF WSQUOTA -- Lowering the value of WSQUOTA for each process to the point that both fit in memory allows the swapper to move pages from one process to the other without having to swap either. This action increases overall page faulting, but in moderation should provide better performance.

**12.4.3.2 Compute-bound Programs** - Contention among compute-bound processes can be alleviated by leaving the quantum at its default setting and increasing the value of the SWPRATE system parameter to as much as 3 to 5 seconds. The compute-bound swapping rate sets the minimum interval (in real time, not processor time) between inswaps of compute-bound processes. (The system defines a compute-bound process as one whose current priority is the same as the default base priority.) A high SWPRATE value under such circumstances should reduce swapping I/O on the system.

Real-time processes are not affected by the value of the SWPRATE system parameter, and interactive processes rarely are. They will be inswapped and outswapped according to priority and quantum. If a compute-bound process must make way for an interactive process, it will be outswapped even if the time specified by its SWPRATE value has not passed (provided its quantum has passed).

### 12.4.4 System Requirements

You can tune system requirements for time and space by adjusting the nonpaged dynamic pool, system working set, and VAX-11 RMS buffers.

**12.4.4.1 Nonpaged Dynamic Pool** - System executive code takes up 150 to 200 pages of physical memory. The nonpaged dynamic pool takes up another 100 to 400 pages, depending a great deal on the size of the system. The system parameters IRPCOUNT, IRPCOUNTV, LRPCOUNT,

LRPCOUNTV, NPAGEDYN, NPAGEVIR, SRPCOUNT, and SRPCOUNTV determine the size of the nonpaged dynamic pool. See Chapter 10 for explanations of these parameters.

During the day-to-day running of the system, you should monitor the nonpaged dynamic pool with the DCL command SHOW MEMORY/POOL/FULL. If a large amount of unused space continually exists in the pool, you can reduce NPAGEDYN or the appropriate xRPCOUNT parameter. If very little unused growth space exists in the pool, you should increase NPAGEVIR or the appropriate xRPCOUNTV parameter.

Running out of space in the nonpaged dynamic pool degrades service more than using some extra space to ensure that this condition does not occur. If the nonpaged dynamic pool is depleted (the size of the pool reaches the limit imposed by NPAGEVIR), processes needing space from the pool will be placed in a miscellaneous resource wait (MWAIT) state until sufficient memory is returned to the pool by other processes or by the completion of I/O services. If many processes enter an MWAIT state, you may be forced to reboot the system.

**12.4.4.2 System Working Set** - The system also requires a working set for the pageable portions of the executive, the paged dynamic pool, VAX-11 RMS, and the resident portion of the system message file. The SYSMWCNT system parameter controls the upper limit (quota) of this working set. Although you usually would not reduce the fluid pages available to the system, SYSMWCNT should be set at a reasonable value. If programs start waiting for the completion of system services, performance degradation is likely to be sharp. During run-time activities, the system automatically decreases the limit if the page fault rate is low, depending on the values of the PFRATL, PFRATH, WSINC, WSDEC, AWSMIN, and AWSTIME system parameters.

An optimal quota for SYSMWCNT varies depending on how the user programs are using the system. Factors dictating a larger size include high use of VAX-11 RMS (especially ISAM services), a high number of logical names, and extensive placement of record locks.

You can monitor the system page fault rate by observing the rate of system page faults on the PAGE MANAGEMENT STATISTICS display with the MONITOR PAGE command (see the Monitor Utility chapter in the VAX-11 Utilities Reference Manual). The system page fault rate should be kept low (two faults per second or less on average -- peaks may be higher).

**12.4.4.3 VAX-11 RMS Buffers** - Where VAX-11 RMS operations are mainly random accesses of single small records, a process can decrease its VAX-11 RMS multiblock count to 1 with the DCL command SET RMS_DEFAULT. This action reduces the size of the VAX-11 RMS buffer required in memory. However, the value of the RMS_DEFAULT system parameter should be left at 16. For more information on tuning systems using VAX-11 RMS, see the VAX-11 Record Management Services Tuning Guide.

## 12.5 TUNING LARGE SYSTEMS

The major strategy in tuning a large system lies in trading more memory for less I/O. On an individual level, programs should either maintain larger data structures mapped into memory from section files (in place of per-record processing by means of VAX-11 RMS), or

increase the multibuffer count and specify read-ahead and write-behind processing for VAX-11 RMS operations. You should install as many programs as possible with the shared and header-resident attributes. On a system-wide basis, you should allot as much memory as practical to the Files-11 ACP data structures, primarily the header, directory, map, file identification, extent, and quota caches. The sections that follow provide guidelines for setting values for the ACP system parameters (see also Chapter 10).

The system parameters, UAF entries, and DCL commands affecting working set size, the page cache, priority, the quantum, the compute swapping rate, the nonpaged dynamic pool, the system working set, and the VAX-11 RMS buffers should, for the most part, be left at their standard values. Any changes should favor increasing the amount of physical memory available to system and user processes. In particular, you should provide working set default and quota values, and PFRATL, PFRATH, WSINC, WSDEC, AWSMIN, and AWSTIME system parameter values that minimize paging I/O, although limit and quota values should not be set arbitrarily or unreasonably high. On a large system, you should be able to minimize paging I/O without increasing swapping I/O. (If swapping I/O does increase, however, you should adjust working set limits as described in Section 12.4.1)

### 12.5.1 Header Cache

Because a file header, once accessed, is likely to require more accesses as the file continues to be processed, maintaining header blocks (one block per page) in memory can save a significant amount of I/O. The following system parameters control the header cache:

- ACP_WRITEBACK -- This switch should be set to a value of 1 to enable the deferred writing of file header blocks, which inhibits the write I/O operation for the file header that otherwise occurs each time a file is extended. The header is always written when the file is closed. Headers for relative and indexed files are always written when the file is extended. A risk is taken in that the file may be lost if a header has never been written and the sytem fails. This loss is usually not serious, since the file is probably in an inconsistent state. The value set by AUTOGEN is normally adequate.

- ACP_HDRCACHE -- The number of header blocks should equal or exceed the number of files that are likely to be open concurrently. You can calculate this value as the product of the number of concurrent users and the median number of files each user accesses concurrently. Reasonable values for a system configured for 48 to 64 users fall in the range of 80 to 200. The default value of deferred writing is normally adequate.

Even on a small system, you should try to allot extra pages to the header cache, as the expenditure in memory is well worth the I/O operations saved.

### 12.5.2 Directory Cache

Because a directory block, once accessed, is likely to require more accesses as files continue to be processed, maintaining directory blocks (one block per page) in memory can save a significant amount of I/O.

The ACP_DIRCACHE system parameter specifies the size of the directory cache. As with ACP_HDRCACHE, you should set a value that equals or exceeds the number of directories that are likely to be accessed concurrently. The value set by AUTOGEN is normally adequate.

The size of the directory cache stands next to the size of the header cache in importance in reducing Files-11 ACP I/O activity.

### 12.5.3 Quota Cache

If disk quotas are being enforced, you should cache one quota entry per active user. The ACP_QUOCACHE system parameter specifies the number of entries cached. One page holds 16 quota entries, so that, for example, a specification of 64 requires 4 pages of storage. The value set by AUTOGEN is normally adequate.

### 12.5.4 System Directory Cache

The ACP_SYSACC parameter specifies the number of directories for which to save access data on a volume mounted with the /SYSTEM qualifier. Its value can be overridden with the /ACCESS qualifier in the DCL command MOUNT. It should be set to the number of directories expected to be in active use on a system volume at one time. Each unit requires 96 bytes of nonpaged pool. However, too low an ACP_SYSACCESS or /ACCESS value partially negates the benefit of the ACP_DIRCACHE parameter.

### 12.5.5 Multiple ACPs

Multiple ACPs (that is, duplicate Files-11 ACPs) for the same disk types are almost never worthwhile. The memory spent on the extra ACP would be better spent on increasing the cache sizes of the first ACP. However, a separate ACP for slower disks may be worthwhile, as it prevents operations on the faster disks from being held up by an ACP clogged with requests for operations on the slower disks.

# INDEX

# INDEX

VAX/VMS System Management
and Operations Guide
AA-M547A-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 user/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

— — 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

BSSG PUBLICATIONS ZK1-3/J35
DIGITAL EQUIPMENT CORPORATION
110 SPIT BROOK ROAD
NASHUA, NEW HAMPSHIRE 03061

— — Do Not Tear - Fold Here — — — — — — — — — — — — — — — — — — — —