Educational Services

digital™

VMS System Management I
Student Workbook

EY-9766E-SG-0002

November, 1988

This document was prepared using VAX DOCUMENT, Version 1.0

TABLE OF CONTENTS

EXAMPLES

FIGURES

TABLES

About This Course

INTRODUCTION

The *VMS System Management I* course is designed to teach students how to manage a computer running the VMS operating system.

This Student Workbook is divided into a number of modules, each designed to cover a well-organized topic, or group of topics. Most modules include figures, tables and examples to enable students to better understand the material. A separate exercise book can be found at the back of this workbook to allow students to test their VMS system management skills.

This section ("About This Course") describes the contents of the course and suggests ways to use its materials most effectively. The following topics are discussed here:

- Resources
- Course Description
- Prerequisites
- Course Organization
- Course Goals
- Nongoals
- Course Conventions
- Course Map

RESOURCES

Students must have access to the following manuals to perform the recommended learning activities of this course. Students may be given their own copy of some of these manuals, and the instructor may provide others for reference during the week.

1. *Guide to Setting Up a VMS System*
2. *VMS SYSMAN Utility Manual*
3. *VMS Install Utility Manual*

4. *Guide to Maintaining a VMS System*
5. *VMS Analyze/Disk_Structure Utility Manual*
6. *VMS Backup Utility Manual*
7. *VMS Mount Utility Manual*
8. *VMS Authorize Utility Manual*
9. *VMS Accounting Utility Manual*
10. *VMS Monitor Utility Manual*
11. *VMS License Management Utility Manual*

Instead of the individual manuals listed above, you may have a copy of the *VMS System Manager's Manual.*

At least one copy of the entire extended VMS documentation set should be available for reference.

COURSE DESCRIPTION

VMS System Management I is designed to train the system manager or advanced system operator of a VAX computer running the VMS operating system.

The course gives a theoretical as well as a practical insight into system management. In addition to routine system management skills, it introduces tools for monitoring system performance and integrity. It discusses various techniques needed to assist users on the system, back up and restore system and user files, start up and shut down the system, manage system devices, and maintain data security.

PREREQUISITES

To derive the greatest benefit from this course, students must be able to:

- Log in to a VMS system.
- Use the appropriate VMS utilities and commands to:
  - — Display on–line help text.
  - — Create and manage directories and files.
  - — Submit print and batch jobs.
  - — Define and use logical names and command synonyms.
  - — Display information about the system.
  - — Create and maintain private disk volumes.
  - — Communicate with other system users.
- Develop DCL command procedures that:
  - — Control I/O.
  - — Create and access sequential files.
  - — Use symbols to manipulate constants and variables.
  - — Use lexical functions.
- Define virtual memory and virtual addressing, and describe how virtual memory and physical memory are related.
- Define a process and describe process concepts, including:
  - — Working set
  - — Paging
  - — Types of processes
  - — States
  - — Priority
  - — Privileges

- Describe the functions of the VMS swapper and scheduler.
- Describe the functions and characteristics of global sections.

These prerequisites can be satisfied by taking the following courses:

- VMS Utilities and Commands
- VMS System Architecture

COURSE ORGANIZATION

This course is organized into a series of modules. Each module has its own learning objectives and covers a single topic or group of closely related topics. Each module consists of:

- An **introduction**, which describes the purpose of the module, provides motivation for mastering its objectives, and outlines its contents.
- One or more **objectives**, which identify the skills taught in the module. Objectives are designed to focus your study efforts on a selected number of skills.
- The module **text**, which consists of:
  - Descriptive text organized in a list format
  - Illustrations, which clarify the relationships among various elements of a VMS system, or summarize steps of a particular process or command
  - Examples containing sample listings from actual interactive sessions on a VMS system
- A module **summary**, which reviews important concepts and skills taught in the module

Written and laboratory exercises are also provided with this course. Exercises help students to review and practice the skills learned during the lecture session.

COURSE GOALS

After completing this course, students should be able to:

- Manage system users, which requires:
  - — Maintaining such files as the User Authorization File and volume quota files
  - — Creating user file directories (UFDs)
  - — Controlling user processes
- Manage system resources, which requires:
  - — Managing disk and tape volumes
  - — Defining device characteristics
  - — Creating and managing print and batch queues
  - — Backing up and restoring files and volumes
- Start up and shut down the system
- Customize the system
- Install maintenance updates and optional software
- Establish security measures on the system and audit security–related actions done by users
- Monitor the system for behavior and performance problems, and submit Software Performance Reports when appropriate

NONGOALS

- Programming using system services, run–time library routines, or other VMS features (taught in *Utilizing VMS Features* courses)
- Advanced VMS concepts or system programming (taught in *VMS Internals* courses)
- Details of layered product features and functions (taught in layered product courses)
- Installation under unusual circumstances
- Details of system performance management and tuning (taught in *VMS System Performance Management*)
- Details of system security features (taught in *VMS System Security Features*)
- VAXcluster management (taught in *VAXcluster System Management*)

- Network management (taught in *DECnet Network Management*)
- Creating new DCL commands, help files, and error messages (mentioned in this course, but covered in detail in *VMS System Management II*)
- Troubleshooting error conditions (taught in *VMS System Management II*)
- System configuration (taught in *VMS System Management II*)
- The Files–11 structure of disk and tape volumes (discussed in VMS System Management II)
- Converting software or data to the VMS system from another operating system

COURSE CONVENTIONS

Table 1 describes the conventions used in the listings and command tables of the Student Workbook.

Table 1: Course Conventions

Convention	Meaning
CTRL/X	Press and hold the key labeled CTRL while you press another key (X). Many control keys have special meanings.

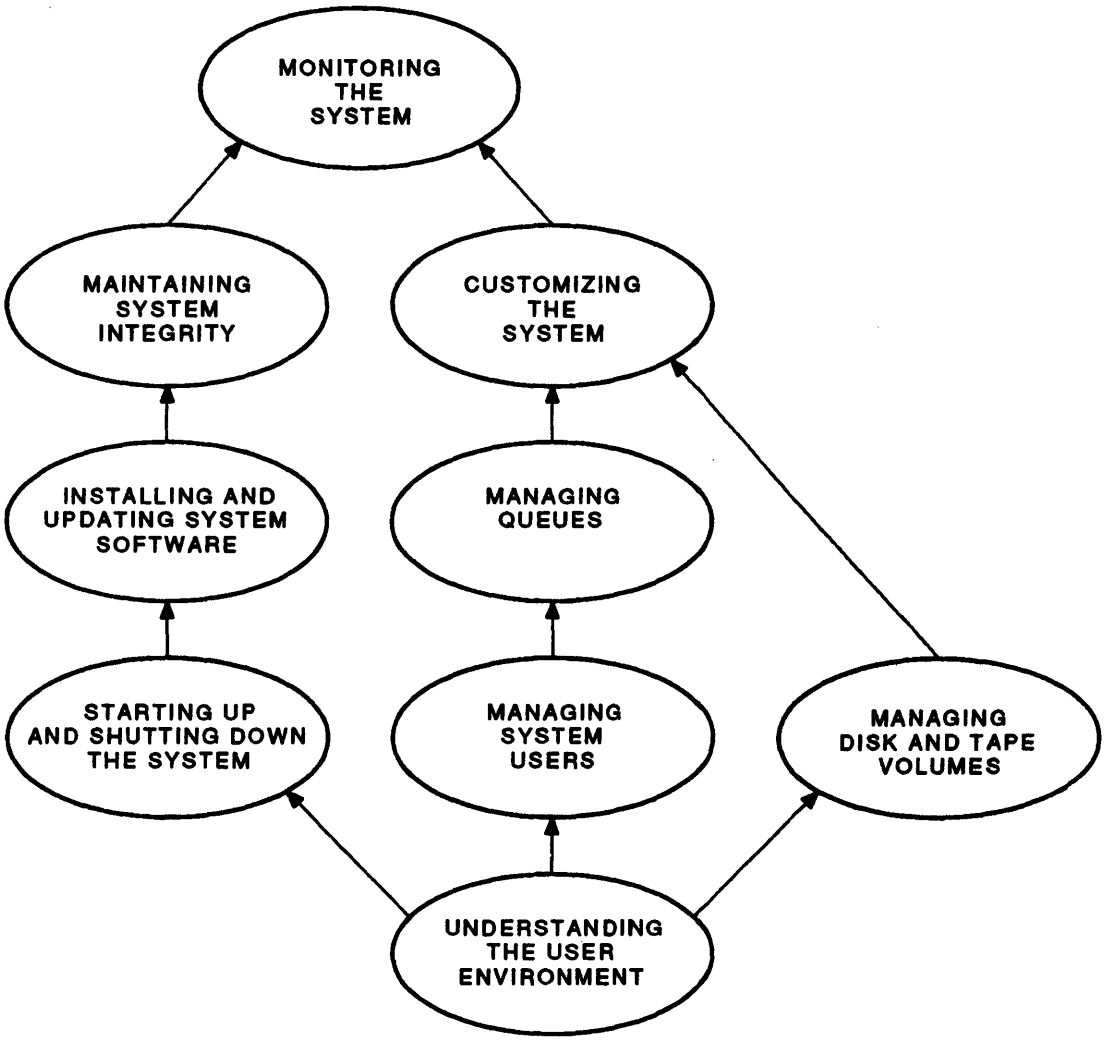
UPPERCASE	In commands, uppercase characters indicate words you type exactly as they appear. For example, you would type the following commands as they appear:
	`$ DIRECTORY`
	`$ TYPE LOGIN.COM`

lowercase	Lowercase characters represent elements that you must replace according to the description in the text. For example, you must follow certain rules when you replace "file–spec" in the following example: `$ TYPE file-spec`

Ellipsis
(. . .)
	Horizontal ellipses indicate that you can enter additional parameters, values, or information. For example, you can enter any number of file specifications in the following example:
	`$ TYPE file-spec, . . .`
	Vertical series of periods or ellipses mean that not all of the data that the system would display in response to the particular command is shown, or that not all the data a user would enter is shown.
	`$ TYPE MYFILE.DAT`
	`.`
	`.`
	`.`
	`$`

Square Brackets ([])	Square brackets indicate that the enclosed item is optional. (Square brackets are not optional, however, in the syntax of some file specifications.) For example, the logical name is optional in the following command: `$ MOUNT/FOREIGN $TAPE1` Braces indicate that you must select from the included items.
Quotation Marks and Apostrophes	The term quotation marks refers to double quotation marks ("). The term apostrophe refers to a single quotation mark (').

COURSE MAP



TTB_X0847_88

MODULE 1
UNDERSTANDING THE USER ENVIRONMENT

INTRODUCTION

When you begin work on a VMS system, you enter an environment consisting of devices, programs and data. The devices that compose the physical computer are called *hardware.* The programs that control the hardware and process the data are called *software.* To perform job–related tasks on the system, you must use both the hardware and the software.

This module provides an introduction to VAX hardware, and an overview of the VMS software environment.

OBJECTIVES

To maintain a VMS system, you should be able to:

- Identify the functions of each component of the hardware environment, namely:
  - — The central processing unit (CPU)
  - — The console subsystem
  - — Main memory
  - — The input/output subsystem
- Identify the purpose of an interconnect device, and list some common interconnect devices for VMS systems.
- Recognize the peripheral devices supported by VMS systems.
- Recognize the format for device names on a VMS system.
- List the characteristics of the following types of system configurations:
  - — Single processors
  - — Tightly coupled systems (multiprocessors)
  - — Loosely coupled systems (networks)
  - — VAXclusters

- Identify the major software components of the VMS operating system.
- Identify software tools and utilities distributed with VMS.
- Identify some optional software products that can be used on VMS systems.
- Identify important system directories, files, and logical names
  - — Describe the concept and uses of concealed root directories
- Identify the components of a user's process.

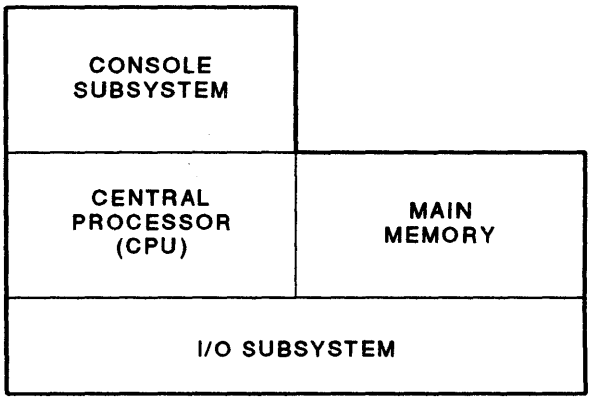
RESOURCES

1. *VMS Installation and Operations Manual* for your particular VAX system
2. *VAX Systems and Options Catalog*
3. *VMS DCL Dictionary*
4. DIGITAL *Terminals and Printers Handbook*
5. DIGITAL *Networks and Communications Buyer's Guide*
6. Refer to the user's guide for each device in which you are interested

COMPONENTS OF THE HARDWARE ENVIRONMENT

- VAX computer hardware is divided into four subsystems
- Each subsystem has a different function

Figure 1–1: VAX Hardware Subsystems



TTB_X0300_88

Central Processing Unit (CPU)

- Executes instructions one at a time
- Relative speeds and physical sizes vary

Console Subsystem

- Communicates directly with the CPU
- Is primarily used for:
  - — Starting up and shutting down the system
  - — Installing software
  - — Remote hardware diagnosis
- Components of the console subsystem include:
  - — Indicators and controls (implemented in software on some VAX processors)
  - — Console terminal
  - — Console storage device (not present on some VAX processors)
- Console subsystem runs in:
  - — **Console mode:** accepts console commands
  - — **Program mode:** use as a normal VMS terminal

Main Memory

- Main memory is used to store instructions and data
- Two types of memory on VAX systems:
  - — READ–ONLY memory (ROM)
  - — READ/WRITE memory
- Each VAX supports a different amount of memory

Input/Output Subsystem

- Consists of peripheral devices
- Provides input to, and output from, the system
- Common peripherals include:
  - — Terminals
  - — Printers
  - — Disk drives
  - — Tape drives

INTERCONNECT DEVICES

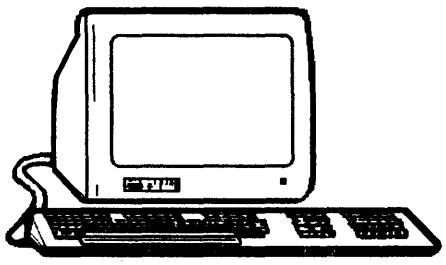
- Cables that connect various subsystems
  - — Example: connect peripherals and memory to processor
- Also called buses
- Types of buses include:
  - — MASSBUS
  - — UNIBUS
  - — Q–bus
  - — VAX Backplane Interconnect (VAXBI)
- Different VAX processors support different buses
- Different buses support different peripherals

PERIPHERAL DEVICES

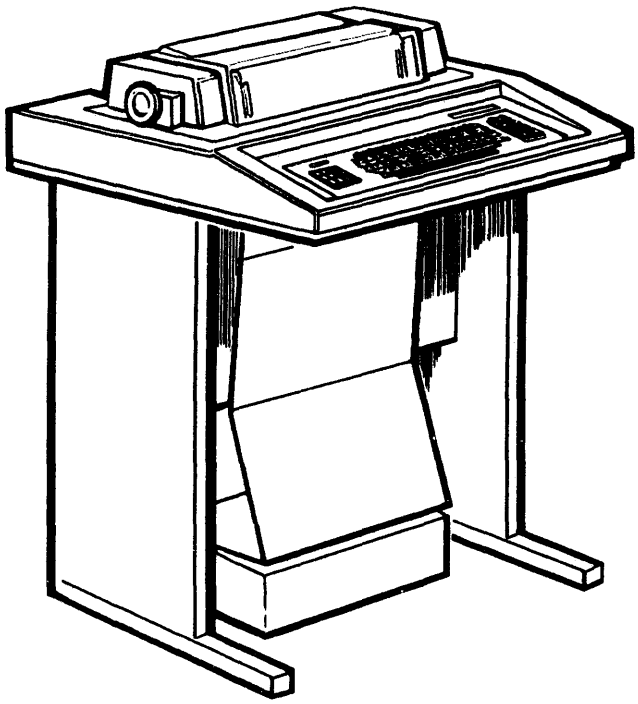
Terminals

- Used by people to communicate with the computer
- Two types of terminals:
  - — Hardcopy
  - — Video

Figure 1–2: Hard-Copy and Video Terminals



A VIDEO
TERMINAL



A HARD-COPY
TERMINAL

TTB X0302 88 SA

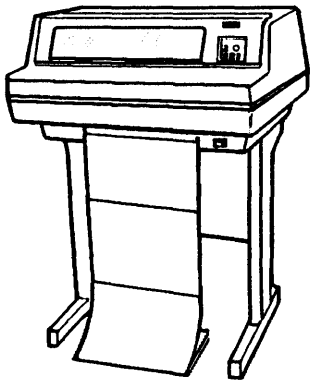
Printers and Printer/Plotters

- Printers provide hard–copy output from the system
- Various sizes and types include:

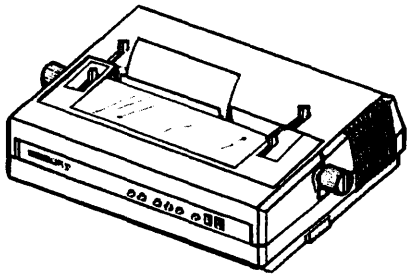
Line printers	*High speed*
Letter quality printers	*High quality*
Laser printers	*High quality print and graphics*

- Printer/plotters are used for both hard–copy and graphic output

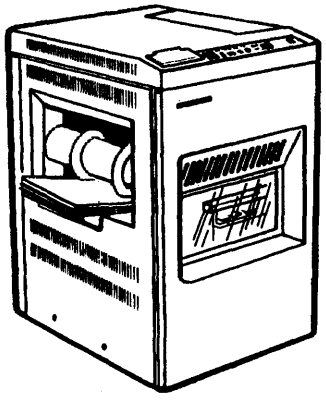
Figure 1–3: Printers and Printer/Plotter



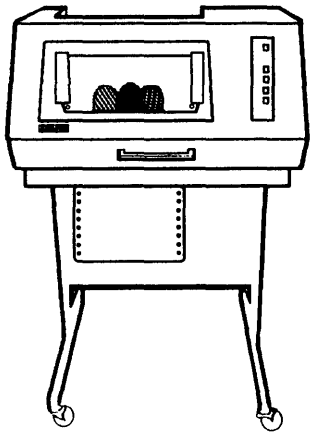
LINE PRINTER



LETTER-QUALITY PRINTER



LASER PRINTER



PRINTER/PLOTTER

TTB_X0303_88_S

Disk Drives

- Record and read data on magnetic or optical disks
- Often called *auxiliary memory*
- Sometimes called *mass storage devices*
- Disks used in the drives are called *storage media*
- Various types of disks include:
  - — Cartridges
  - — Disk packs
  - — Diskettes (also called "floppies")
  - — Two types of optical disks:

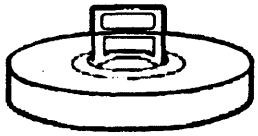
    Read–Only (CDROM)

    Commonly called "compact disks"
    Mastered by the manufacturer, read by the user
    Small format (5.25 inches in diameter)
    Typically used to provide large amounts of text information, such as reference manuals, abstracts, etc.

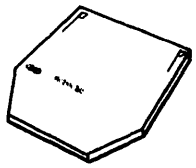
    Write–Once, Read–Many (WORM)

    Commonly called simply "optical disks"
    Available in both small and large (12 and 14 inch) format
    The disk can be written one time, then read many times
    Typically used for archival data storage, and large volume storage of data with slow retrieval requirements
- Disks can be:
  - — Fixed
  - — Removable

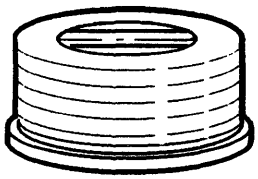
Figure 1–4: Examples of Disks



DISK CARTRIDGE
(TOP LOADING)



DISK CARTRIDGE
(FRONT LOADING)

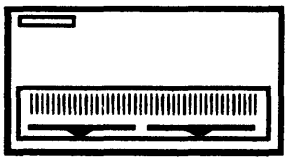


DISK PACK
(TOP LOADING)

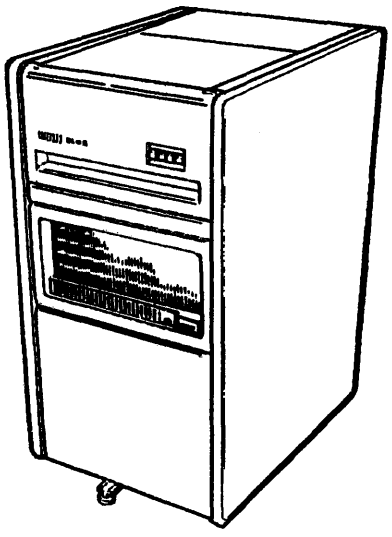
DISKETTE

TTB_X0304_88_S

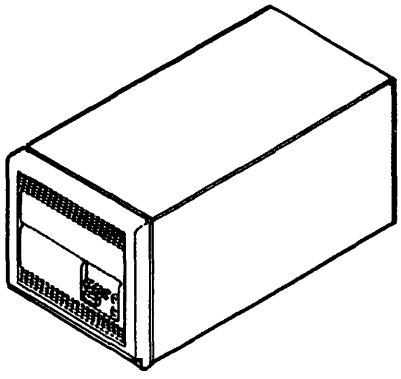
Figure 1–5: Disk Drives



DUAL DISKETTE DRIVE



DISK PACK DRIVE



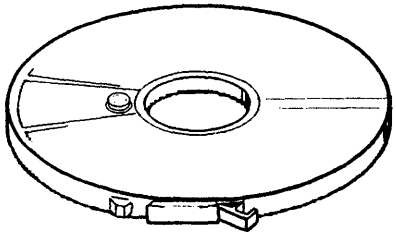
DISK CARTRIDGE DRIVE

TTB_X0305_88_S

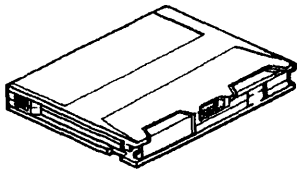
Tape Drives

- Record and read data on magnetic tapes
- Tapes usually store:
  - — Backup copies of data
  - — Infrequently used data
- Reel tapes
  - — Various lengths
  - — Various densities
- Tape cartridges
  - — Resemble cassette tapes
  - — VMS systems treat TU58 cartridges as disks

Figure 1–6: Tape Media



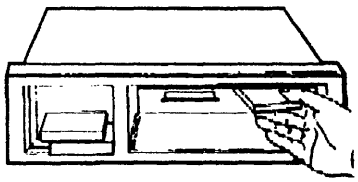
REEL TAPE



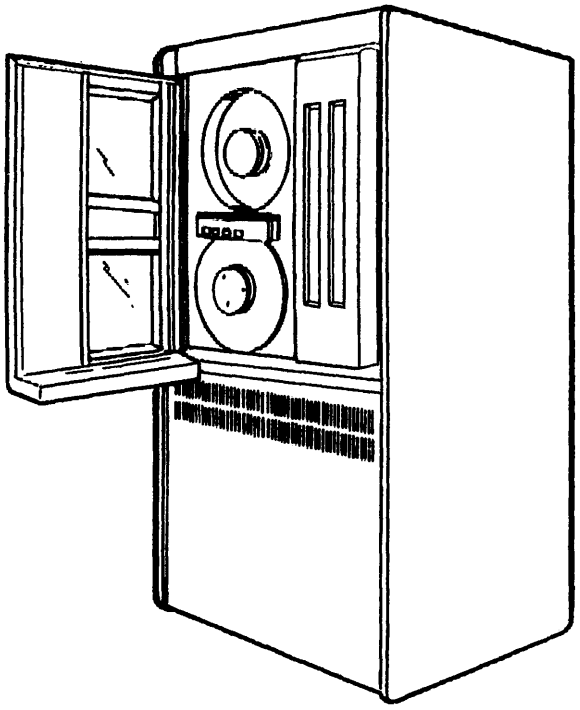
TAPE CARTRIDGE

TTB_X0306_88_S

Figure 1–7: Tape Drives



CARTRIDGE TAPE DRIVE



REEL-TO-REEL TAPE DRIVE

TTB_X0307_88_S

Hierarchical Storage Controller (HSC)

- Intelligent disk and tape controller
- Performs all disk I/O operations
- Allows data sharing in a VAXcluster
- Can also be used with single VAX systems
- Examples of HSC subsystems:
  - — HSC50
  - — HSC70

DEVICE NAMES

- All devices have a unique name in the format: **ddcu**

  dd A two–letter device code.

  c A one–letter code that specifies the hardware controller for the device. (Controllers provide the interface between the bus and the device, or between two buses.)

  u The unit number of the device.

- Device code specifies device type
- The hardware controller number:
  - — Identifies the device controller
  - — Is represented by a letter from A to Z
  - — Is assigned by the system
- The unit number:
  - — Indicates the position of the device on the controller
  - — Can be changed by:
    1. Setting a button or switch on the device
    2. Installing a unit plug on the device

Table 1–1: Device Codes

Code	Device
CS	Console storage device
DU	RA80 or RA81 disk drive
LP	Line printer
MF	TU78 tape drive

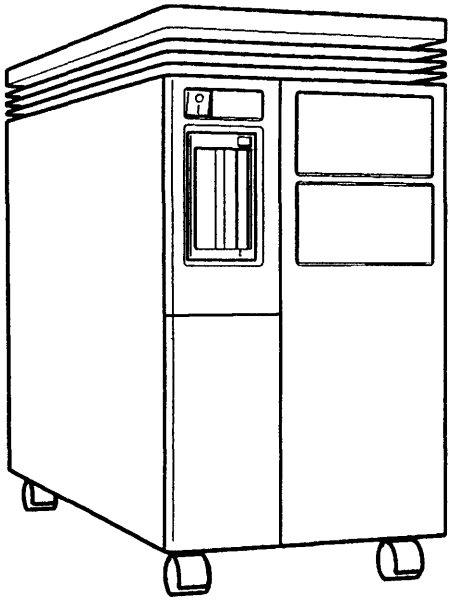
SYSTEM CONFIGURATIONS

- You can build different configurations with:
  - VAX processors
  - Interconnects
  - Peripheral devices
- System configurations can be classified as:
  - Single processors
  - Multiple processors
- A system can be:
  - A single VAX processor and its peripheral devices
  - A collection of VAX processors

Single Processor Configurations

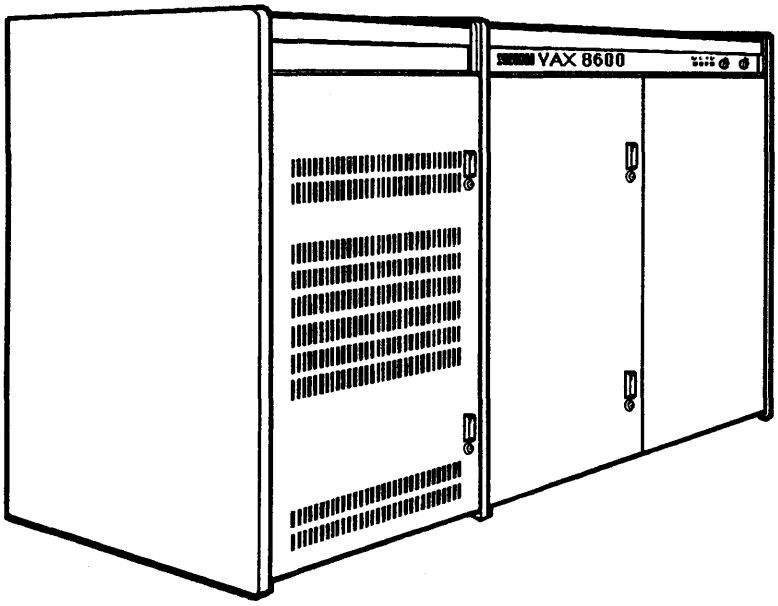
- Any single VAX processor and its peripheral devices
- The family of single VAX processors includes:
  - VAX 8810
  - VAX 8600
  - VAX 8500
  - VAX 8200
  - VAX–11/780
  - VAX–11/750
  - MicroVAX 3000
  - MicroVAX II

Figure 1–8: MicroVAX II



TTB_X0308_88

Figure 1–9: VAX 8600



TTB X0309_88_S

Multiple Processor Configurations

- Two or more communicating processors
- Three classifications:
  - — Tightly coupled configurations (multiprocessors)
  - — Loosely coupled configurations (networks)
  - — VAXclusters

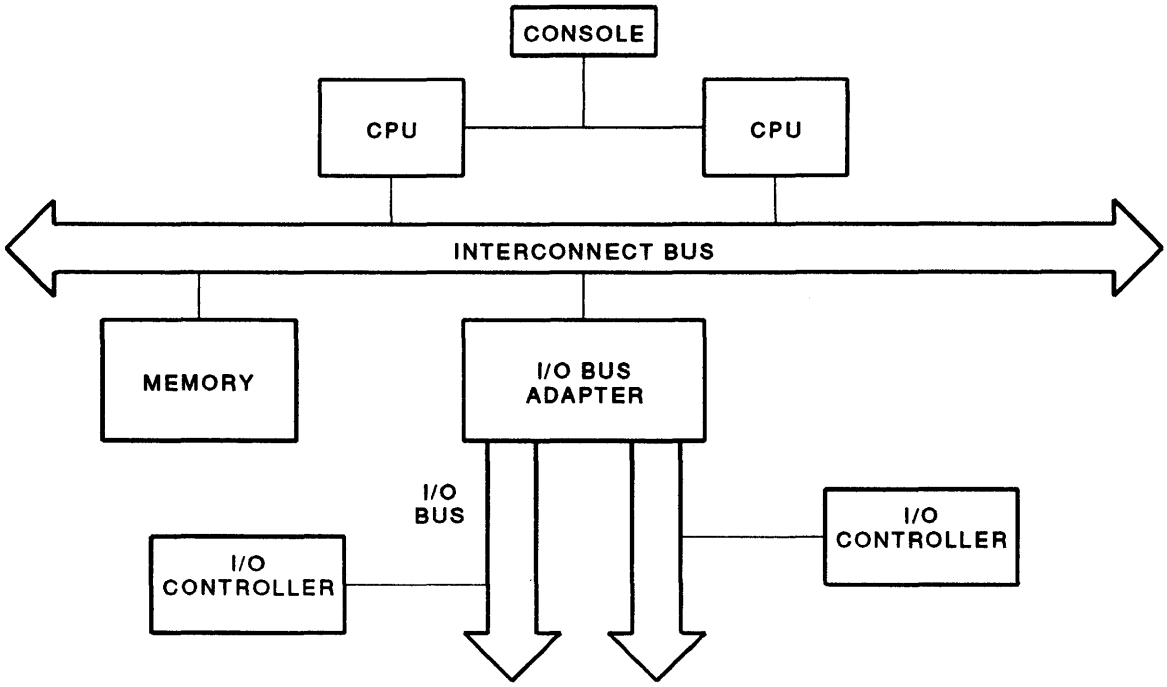
Table 1–2: Comparing Multiple Processor Configurations

	Tightly Coupled <————> Loosely Coupled		
System Characteristic	Multiprocessor	VAXcluster	Network
CPU booting	Together	Separate	Separate
CPU failure	Together	Separate	Separate
CPU cabinet location	Single or adjacent	Same local area	Can be widely separated
Security domain	Single	Single	Multiple
Management domain	Single	Single	Multiple
Operating system	Shared (VMS)	Separate (all VMS)	Separate (some may not be VMS)
File system	Integrated	Integrated	Separate
Growth potential	Limited	Very great	Very great

Tightly Coupled Configurations

- Share operating system code
- Cannot operate independently
- Operated and managed as single domains
- Provide high performance
- Used primarily in compute-intensive applications
- Example: VAX 8800 Series processors
  - — Two or more CPUs share memory by means of a high speed bus
  - — Multiple processors are transparent to the system users

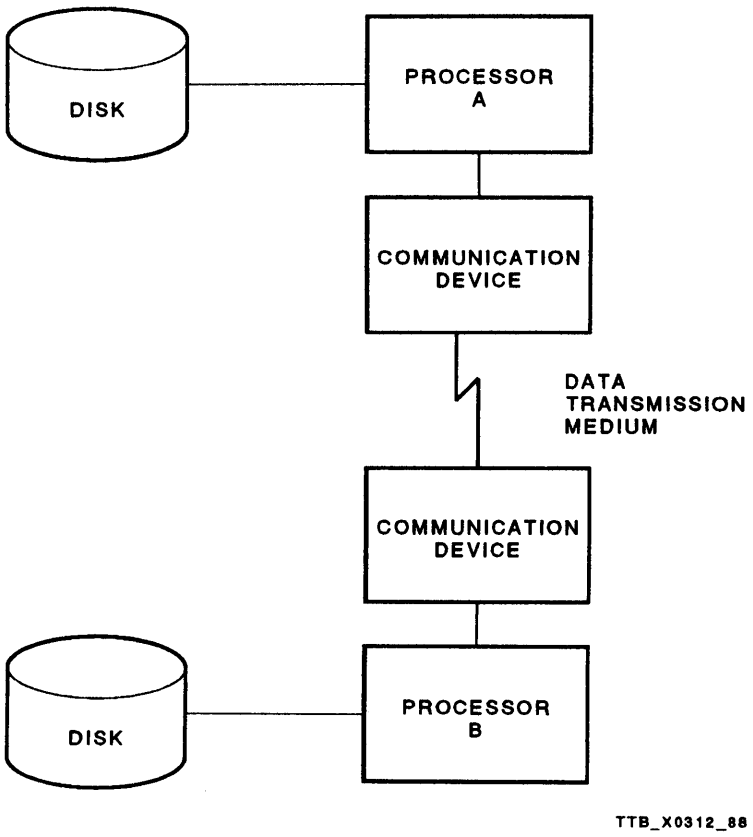
Figure 1–10: Tightly Coupled System Configuration



Loosely Coupled Configurations

- Example: Network
- Can consist of two or more communicating processors
- VMS systems can be connected to:
  - — Other DIGITAL systems
  - — Other manufacturers' systems
- Local Area Network (LAN) – limited geographical area
- Wide Area Network (WAN) – larger area
- DIGITAL–to–DIGITAL networks are established using:
  - — Two or more processors
  - — Hardware communication devices
  - — Data transmission devices
  - — Terminal servers (optional)
  - — DECnet software

Figure 1–11: DECnet Network



Communication Devices

- Synchronous devices
  - — Provide communication between computers
  - — Run at relatively high speeds
  - — Examples: DEUNA, DELUA, DEQNA, DMR11
- Asynchronous devices
  - — Typically provide communication between terminals and computers
  - — Are slower than synchronous devices
  - — Examples: DL11, DZ32, DMZ32
- Synchronous/Asynchronous Devices
  - — Perform both synchronous and asynchronous communications
  - — Examples: DMF32
- Different communication devices vary by:
  - — Characteristics
  - — Function
  - — Performance

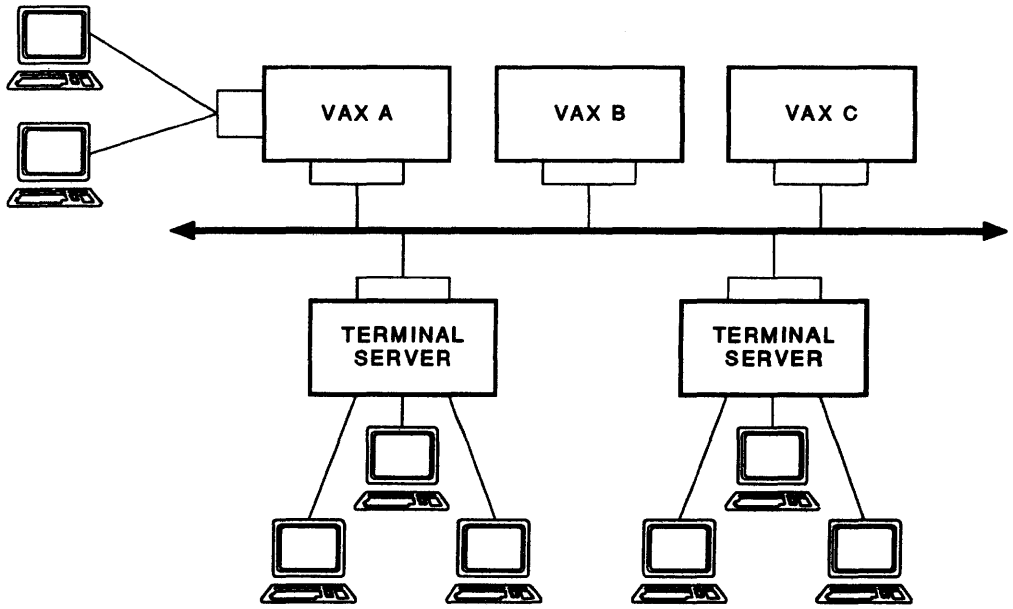
Data Transmission Media

- Transmits data between communication devices
- Various types of media include:
  - — Ethernet cable
  - — Fiber optic cable
  - — Telephone lines (twisted–pair)
  - — Satellite links
- Different transmission media vary by:
  - — Transmission speeds
  - — Distances they can span
  - — Operating and performance characteristics
  - — Capabilities and services

Terminal Servers

- Optional
- Provide increased flexibility
- Connect terminals to computers in a LAN
- Each terminal can access other systems connected to the same LAN
- Can also be used to make printers accessible to other systems on the LAN

Figure 1–12: Configuration with Terminal Servers



Clustered Systems (VAXcluster Systems)

- Flexible multiprocessing systems
- Have characteristics of both loosely and tightly coupled systems
- Managed as single domains
- VAXcluster nodes can share:
  - — Disk and tape devices and printers
  - — A common file system
- VAXcluster nodes can be configured:
  - — To present an identical user environment on every node
  - — So that each node boots and fails separately
- In addition to providing the functions of a network, a VAXcluster system provides:
  - — Higher availability of system resources
  - — Faster and easier sharing of information and resources between nodes
- May start with one or two VMS systems connected by way of Computer Interconnect or Ethernet
  - — Can be expanded as computing needs increase
- VAXcluster hardware components include:
  - — VMS computer systems, composed of:

    VAX processors
    Local peripherals
  - — HSC storage subsystems
  - — A Computer Interconnect (CI), composed of:

    A CI bus
    An SC008 Star Coupler
    An intelligent I/O port (a CI780, CI750, or CIBCI)
  - — Ethernet hardware

- A VAXcluster configuration:
  - — Can be very similar to a network configuration
  - — Can contain the same components as a network:

    VAX processors
    Communication devices
    Transmission media
    Terminal servers (optional)
    DECnet software
- Major difference between a VAXcluster and a network:
  - — VMS cluster software, which synchronizes access to shared resources

Figure 1–13: CI–Only VAXcluster Configuration

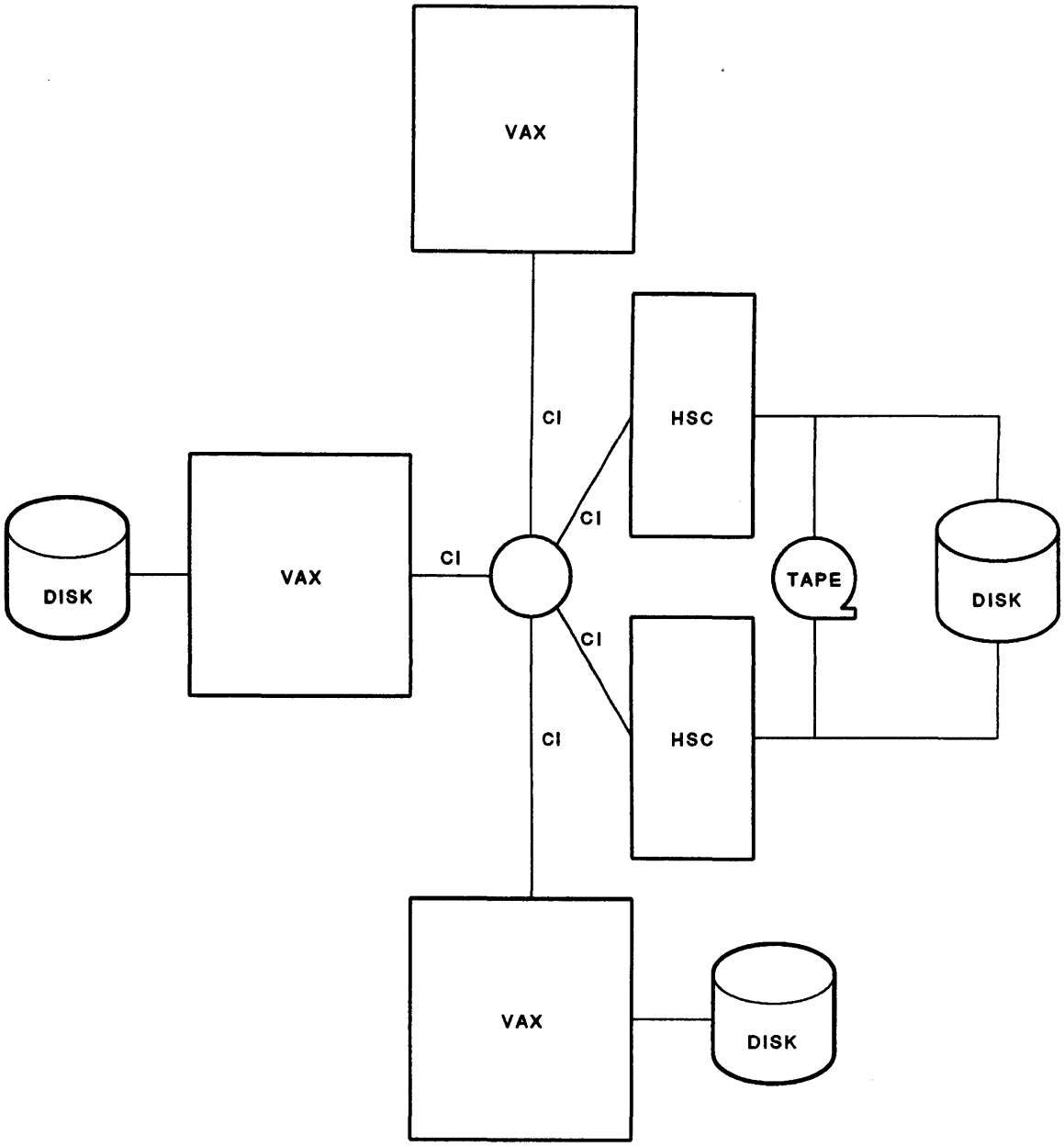


Figure 1–14: Local Area VAXcluster Configuration

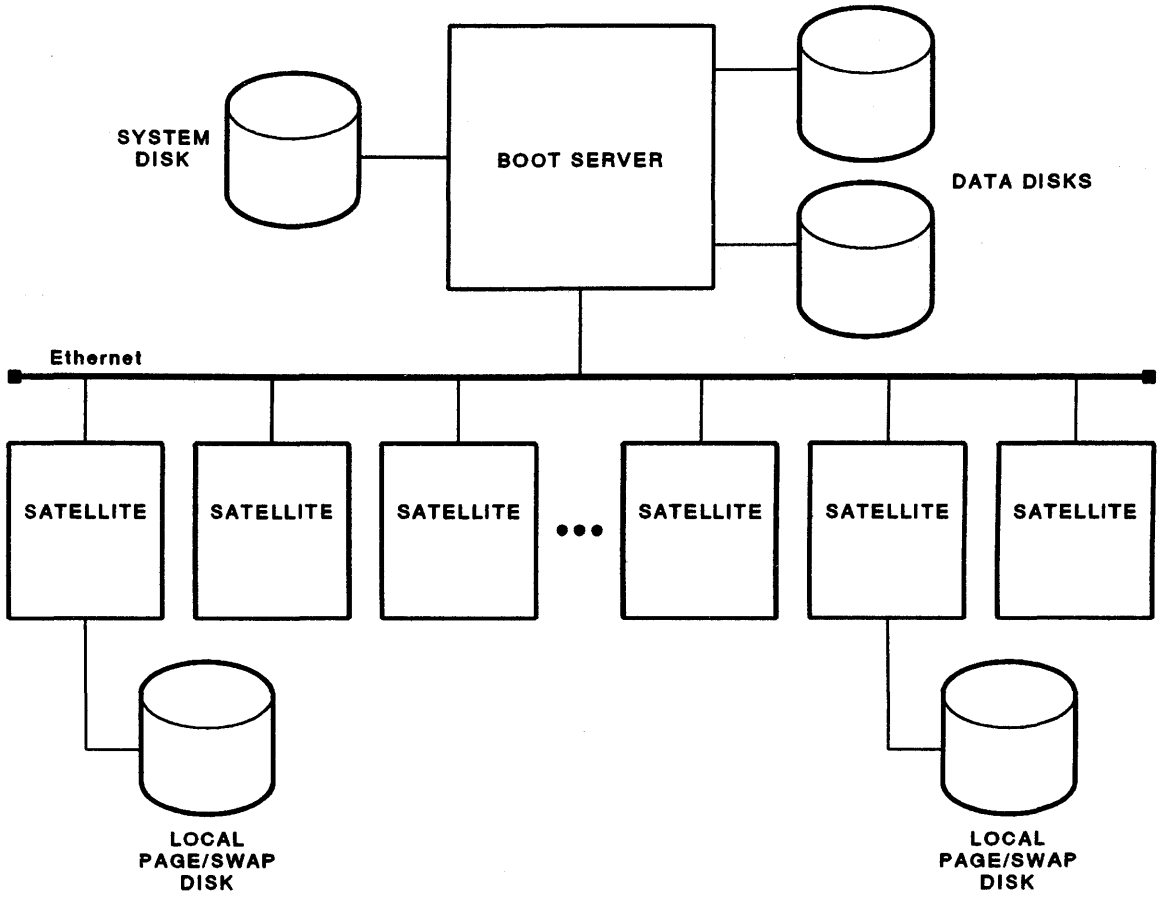
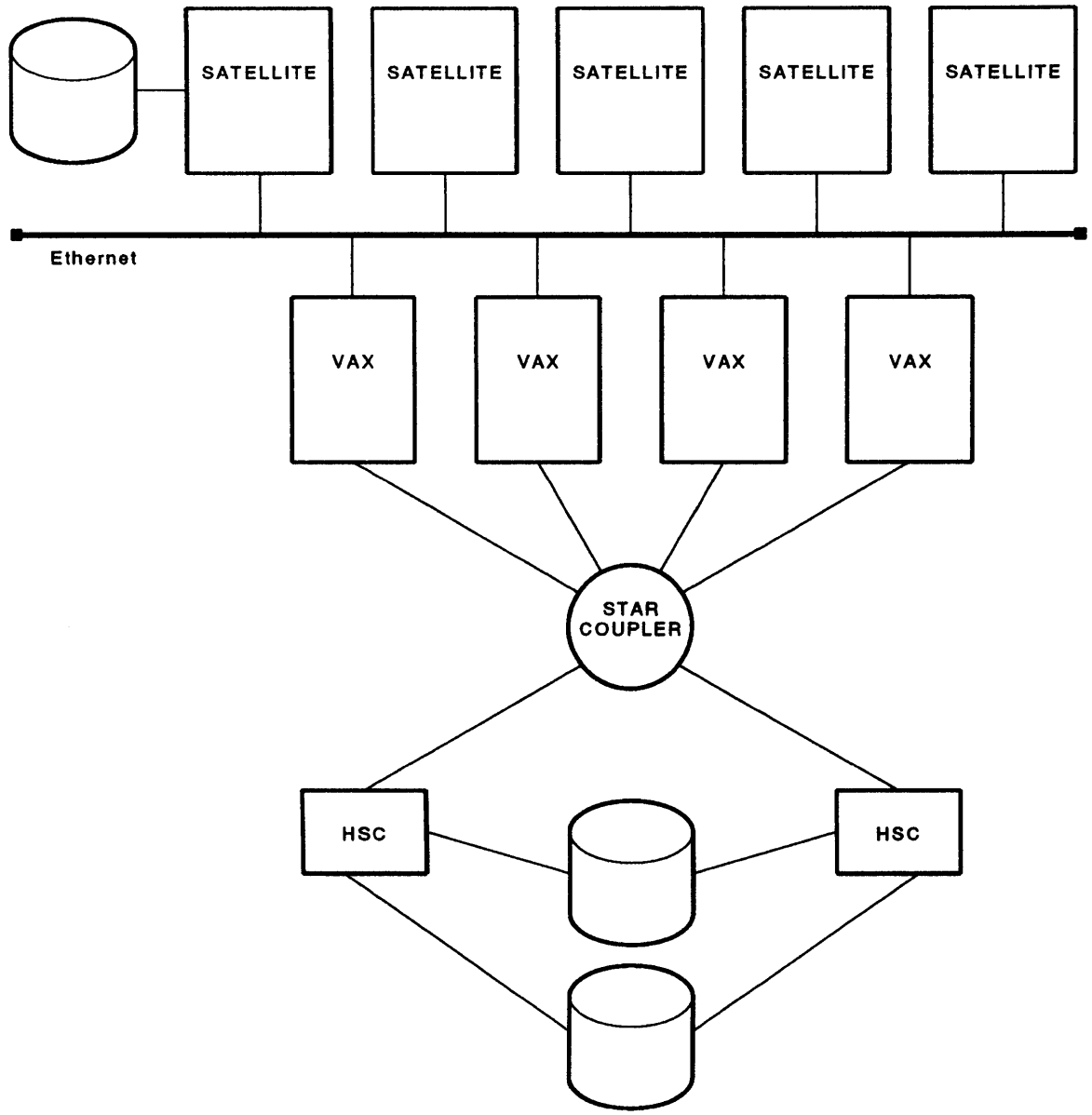


Figure 1–15: Mixed–Interconnect VAXcluster Configuration



Review of VMS Concepts

- Topics to be covered:
  - DCL and the VMS Operating System
  - Programs, Images, and Utilities
  - Processes
  - Hardware and Software Contexts
  - Virtual Address Space
  - Working Sets and Balance Set
  - Paging
  - Scheduling
  - Swapping
  - Processor Access Modes

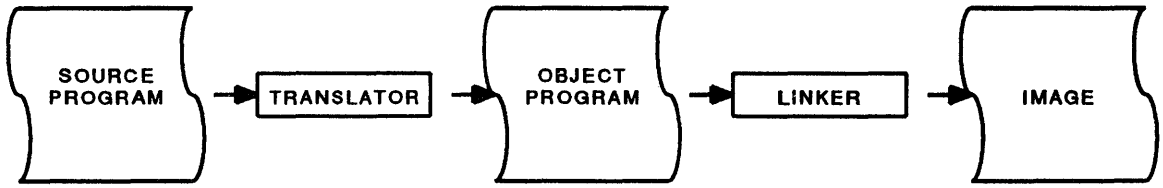
DIGITAL Command Language (DCL) and the VMS Operating System

- The **VMS Operating System** performs three major functions:
  - — Provides communications between users and devices
  - — Creates a protected working environment for each user
  - — Schedules the use of system resources among all users in a fair manner
- Many operating system activities are transparent to the user
  - — Some activities initiated by the system
  - — Other activities initiated by users through *commands*
- The **DIGITAL Command Language** (DCL)
  - — A language definition of English–like commands
  - — A set of routines that execute user commands
- The **Command Language Interpreter** (CLI)
  - — Translates English–like commands from a user
  - — Dispatches translated commands for execution

Programs, Images, and Utilities

- A *program:*
  - — Also called a *source program*
  - — A logical sequence of instructions
  - — Written in human–readable form (computer language)
- An *object file:*
  - — A collection of machine code (binary) instructions
  - — Created as a result of *translating* (or *compiling*) the source program
- An *image:*
  - — Also called an *executable image*
  - — A sequence of machine code instructions with addressing information
  - — Created as a result of *linking* object files together
- A *utility:*
  - — Program supplied with VMS
  - — Sometimes has a command language all its own
  - — Typically uses DCL to call (invoke) the utility

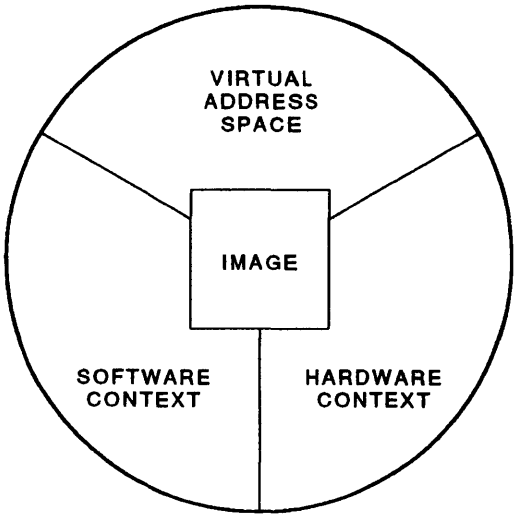
Figure 1–16: Translating a Program into an Image



Processes

- A *process* is:
  - — The total environment in which an image executes
  - — The fundamental entity that VMS schedules for execution
- A process is composed of four parts:
  - — Virtual address space
  - — Hardware context
  - — Software context
  - — Image
- Hardware and software contexts are collectively called the *process context*
  - — VMS needs this information to save/restore the process during *context switching*

Figure 1–17: Components of a VMS Process



TTB_X0258_88

- A *subprocess* is:
  - — Just another process
  - — Controlled by the parent process
  - — Defined by the attributes and capabilities of the parent process (by default)
  - — Automatically deleted when the parent process is deleted
  - — Able to create subprocesses
- Total number of subprocesses a job can create depends on the job's *subprocess quota* which are authorized by the VMS system manager

NOTE

A job is a unit of work performed by the CPU. In the VMS operating system environment a job is the accounting unit equivalent to a process and subprocess.

- Images come and go, but the process remains until deleted by:
  - — The user logging out of the system
  - — The process' parent process deletes it
  - — The parent process is deleted
  - — Invoking one of the following the DCL command:
    - **STOP process_name**
    - **STOP/IDENTIFICATION=pid**
    - **STOP PROCESS/ID=pid**
- A job possesses a *process structure*
  - — Arranged in a hierarchical ("tree") structure

Hardware and Software Contexts

- The process *hardware context* is:
  - — The state of the VAX processor at any given moment
  - — Specifically the contents of the processor registers
  - — What VMS requires to be able to restore a process so it can continue execution at exactly the point it was interrupted
- The process *software context* is:
  - — Analogous to the hardware context
  - — Identified by a unique *process identification number* (PID)
  - — Accounting information
  - — A collection of information about what the process can and cannot do

    Priority
    Privileges
    Resource Limits (Quotas)
- The VMS system manager authorizes the priorities, privileges, and quotas associated with a process

Virtual Address Space

- *Virtual address:*
  - — A symbolic or logical representation of a physical memory location in hardware
- *Virtual address space:*
  - — Range of virtual addresses referenced by the image
- *Virtual memory system:*
  - — A way for a program to be written as if it had a very large number of contiguous memory locations available for use
- VMS manages the translation of virtual memory addresses to physical memory locations in hardware
  - — Transparent to the user and program
- *Memory management subsystem:*
  - — Part of the VMS operating system
  - — Consists of three components:

    Page fault handler
    Modified page writer
    Swapper

Working Sets and Balance Set

- Managing virtual memory:
  - — Involves moving pieces (called *pages*) of the image to and from auxiliary storage
- Process *working set:*
  - — Pages of physical memory currently used by a process
  - — Limited by various *working set quotas:*

    Working set default
    Working set quota
    Working set extent
- The system *balance set:*
  - — Set of processes whose working sets are resident in physical memory
  - — Analogous to the **working set** of a process

Paging

- *Page fault:*
  - Occurs when an image references a virtual address not currently in the working set
  - Handled by the memory management routine called the *page fault handler*
- Pages are *faulted* into the process' working set as the image executes
  - Called *paging on demand*, or *demand paging*
- *Working set extent:*
  - A kind of "overdraft" allowed when a process reaches its working set quota but still needs additional pages
  - Managed by the VMS system manager
  - When the working set extent is reached, the process must surrender pages from the working set if additional pages are needed
- Which pages are surrendered are determined by the *working set list*
  - Ordered list of pages in the working set
  - Pages in the list are ordered according to age
  - Oldest page is surrendered first
  - New pages are appended to the end of the list

Free Page List

- List of unused (free) pages available for use by processes
- An ordered list of pages similar to the process working set list
  - — Oldest pages are at the top of the list, given to processes when they request additional pages
  - — Unmodified pages surrendered by processes are placed at the bottom of the list
- Controlled by the memory management subsystem
- Size of the list can be controlled by the VMS system manager
- When a process incurs a page fault for a page previously included in the working set:
  - — The page fault handler retrieves the page from either:

    The free page list
    The image file (if the page has already been given to another process)
- This procedure is performed for **unmodified** pages only

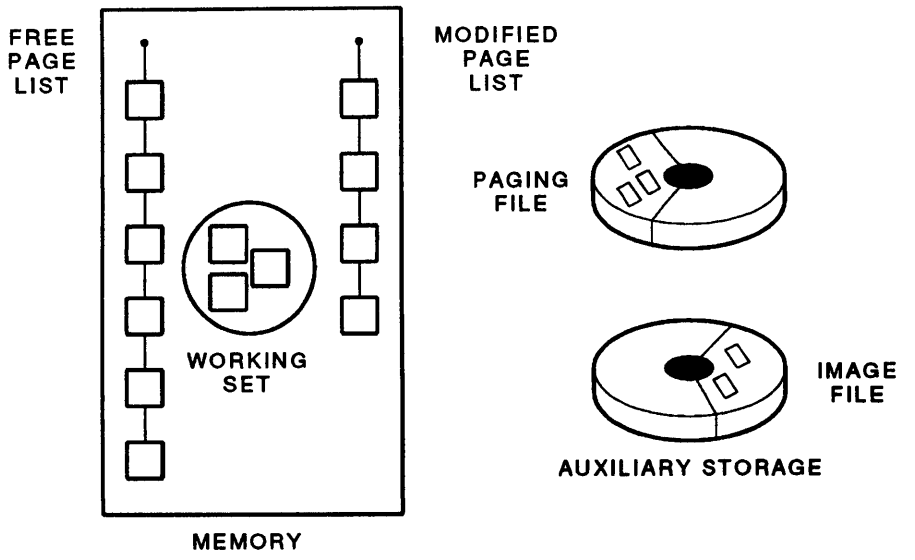
Modified Page List

- Similar to the free page list, but for modified pages (those written into by a process)
- Controlled by the memory management subsystem
- When the free page list falls below a certain minimum limit:
  - The *modified page writer* takes pages away from the modified page list and writes them into a *paging file*
  - The pages taken from the modified page list are placed on the free page list
- A paging file:
  - A specially designated file in auxiliary storage
  - Created under the direction of the VMS system manager
  - Used to retain pages removed from the modified page
  - Controlled by the memory management subsystem
  - There can be more than one paging file in a VMS system

    Multiple paging files can improve system performance
    The memory management subsystem manages access between the files
- When a process incurs a page fault for a **modified** page:
  - The page fault handler will find the page in one of these locations:

    Modified page list
    Free page list
    Paging file

Figure 1–18: Pages in Memory and in Auxiliary Storage

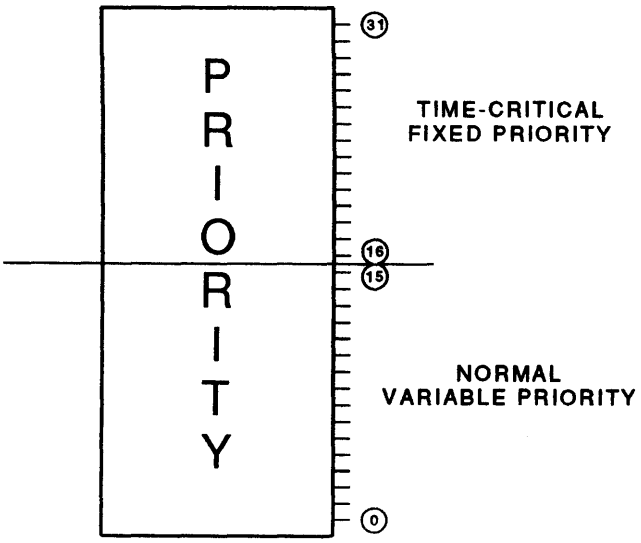


Scheduling

- A technique for providing fair sharing of the processor among a number of competing processes
- This function is managed by the VMS *scheduler*
- Processes are scheduled for execution using a combination of two techniques:
  - — Priority scheduling
  - — Time–sliced ("round–robin") scheduling
- Priority levels:
  - — Two groups of 16 levels each

    Real–time priority levels (16–31)
    Normal priority levels (0–15)

Figure 1–19: Priority Levels in VMS



Swapping

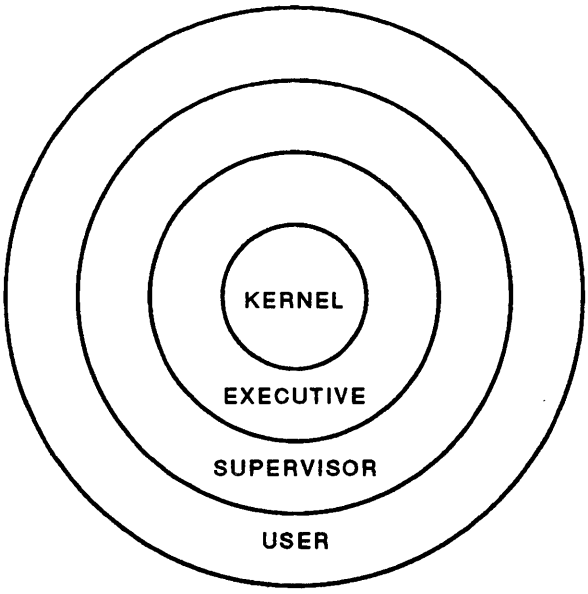
- The technique used to move entire process working sets from physical memory to auxiliary storage
  - — Required to allow a large number of processes to share the system
  - — Not all process working sets can necessarily fit into the available system memory
  - — Managed by the VMS *swapper*
  - — Closely coordinated with scheduling
- The *swapping file:*
  - — A special system file in auxiliary storage
  - — Managed by the swapper
  - — Created under the direction of the VMS system manager

Processor Access Modes

- A mechanism to safeguard memory from:
  - — Accidental actions (untested or faulty programs)
  - — Malicious actions (reading or writing to memory not owned by the process)
- Four different access modes (in decreasing order of privilege):
  1. Kernel
  2. Executive
  3. Supervisor
  4. User
- A process can change modes during image execution
  - — Implicitly, through execution of certain system service routines
  - — Explicitly, by executing a VAX change mode instruction

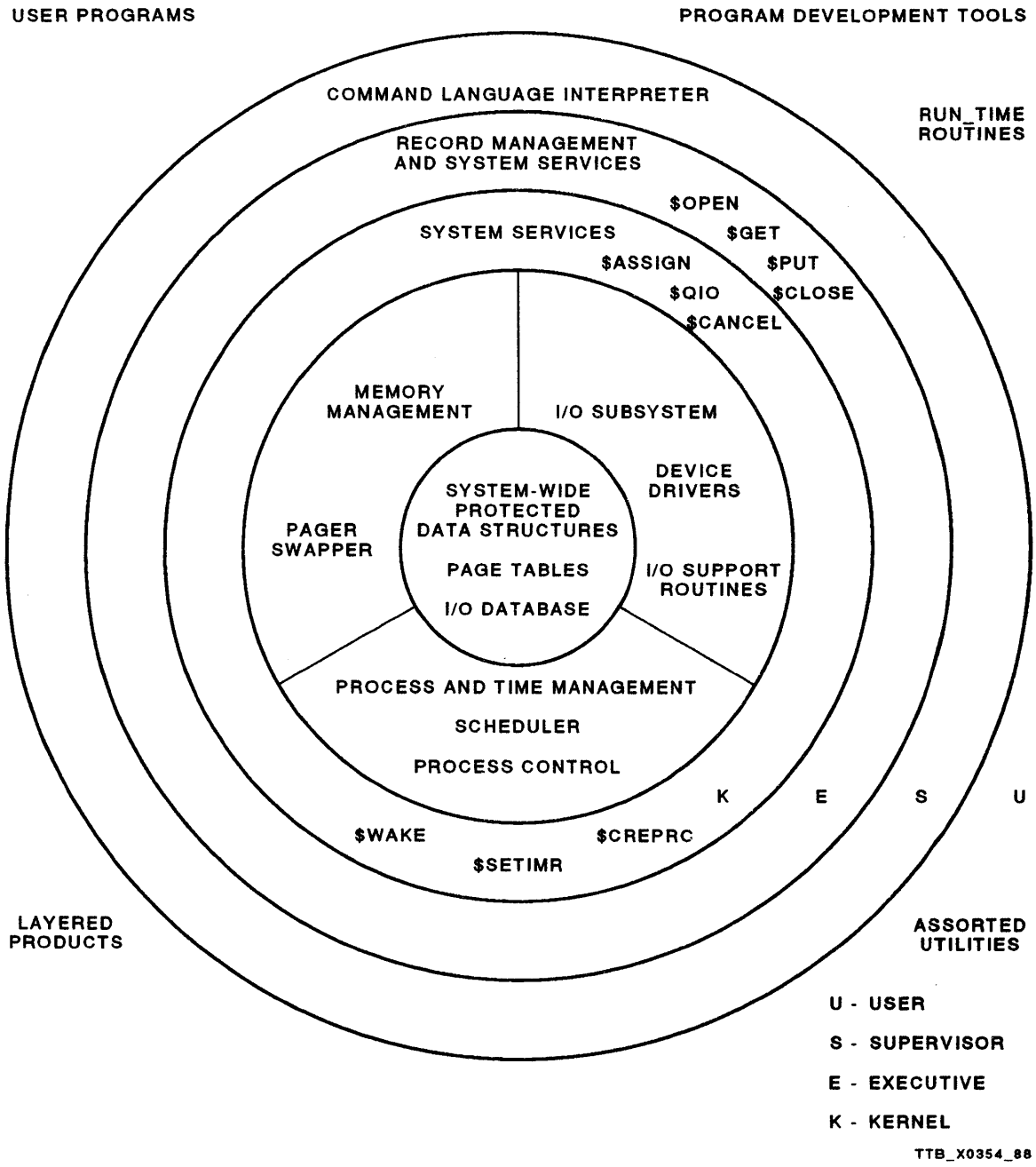
    Requires special privileges, the most common being CMKRNL
    Authorized by the VMS system manager on a per–user basis

Figure 1–20: Access Mode Hierarchy



TTB_X0353_88

Figure 1–21: Layered Design of the VMS Operating System



System Files, Directories, and Logical Names

System Files

- VMS stores its image files, data files, and other files in special locations in auxiliary storage:

System volume	Disk volume, which holds system files
System device	Disk drive on which the system volume is mounted
System directories	Directories on the system volume that store the system files (see Table 1–3)

- A list of all the files distributed in the system directories can be found in the *VMS Installation and Operations Manual* for your particular VAX processor.

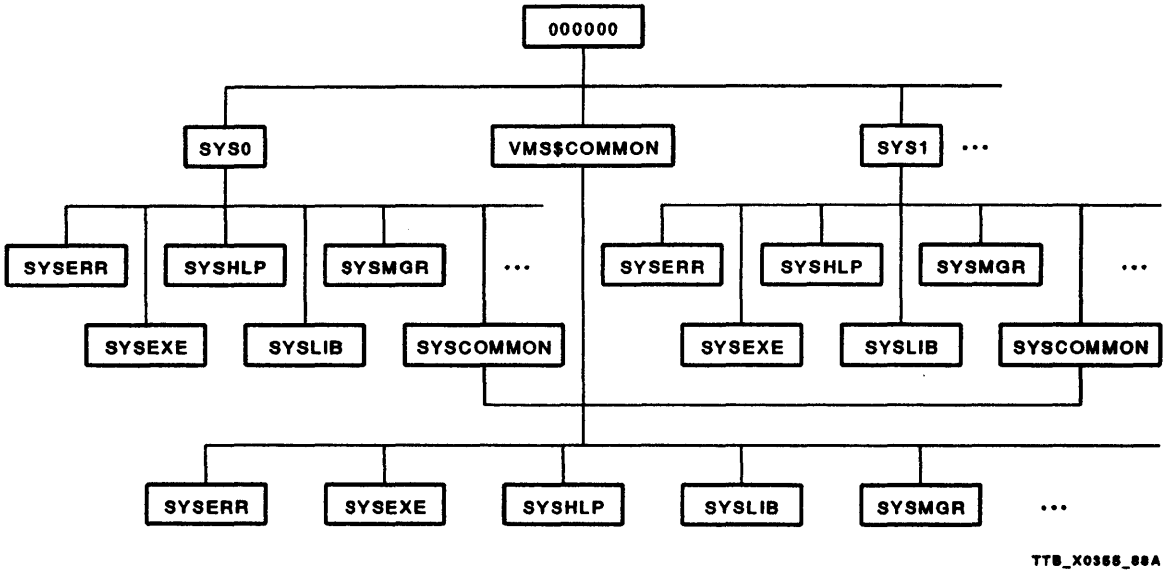
System Directories

- Each system directory has its own logical name, specifying:
  - — The system device
  - — The specific system directory
- To use the directory name to refer to a file in a system directory, use the logical name SYS$SYSROOT as the device name.
  - — Defined as a *concealed root directory*

Table 1–3: System Directories

Directory Name Used in Combination with SYS$SYSROOT:	Logical Name	Principal Contents
[SYSEXE]	SYS$SYSTEM	Executable images, user authorization file, site–independent startup and shutdown command procedures, system parameter files, paging file, swap file, crash dump file
[SYSLIB]	SYS$LIBRARY SYS$SHARE	Macro libraries, object libraries, shareable image libraries, shareable images
[SYSHLP]	SYS$HELP	Help libraries
[SYSMSG]	SYS$MESSAGE	Error message files
[SYSMGR]	SYS$MANAGER	Site–specific startup and shutdown command procedures, print forms definitions, operator's log, accounting file
[SYSMAINT]	SYS$MAINTENANCE	Hardware diagnostics (not distributed with VMS)
[SYSTEST]	SYS$TEST	UETP command procedures, images, data
[SYSERR]	SYS$ERRORLOG	Error log file
[SYSUPD]	SYS$UPDATE	Command procedures used for system installation, upgrade, and update
[SYSUPD.EXAMPLES]	SYS$EXAMPLES	Sample source programs such as device drivers and system services
[SYSCBI]	SYS$INSTRUCTION	Computer–based instruction software
[SYS$LDR]	SYS$LOADABLE_IMAGES	Loadable executive images and device drivers
[SYS$STARTUP]	SYS$STARTUP	Start–up command procedures

Figure 1–22: System Directory Tree Diagram



Concealed Root Directories

- Allows flexible handling of multiple versions of VMS on the same system
- The system root directories are of the form: [SYSn]
  - — Where "n" is a single hexadecimal digit from 0 through F
  - — A typical VMS system will only have one system root directory: [SYS0]
- Be sure not to mix the use of root directory names and top–level directory names:

  Incorrect: `SYS$SYSROOT:[SYS0.SYSEXE]LINK.EXE`

  Correct: `SYS$SYSROOT:[SYSEXE]LINK.EXE`

System File and Directory Protection

- System directories distributed with sufficient protection
- The VMS system manager should **NOT** change their protection
- Be sure you fully understand the VMS file protection mechanisms
  - — Consult the command description for **SET PROTECTION** in the *VMS DCL Dictionary*.
- Don't forget you can actually *over-protect* certain files, usually by accident

Software Available for VMS Systems

Table 1–4: Programs and Utilities Distributed with VMS

Editors and Word Processing Tools	
EDT	SOS Interactive Text Editor
TPU	RUNOFF Text Formatter
EVE	SLP Batch Text Editor
Program Development Tools	
VAX MACRO Assembler	VAX Librarian
VAX Linker	VAX Common Run–Time Library
VAX Debugger	
File Manipulation Tools	
VMS COPY Utility	
VMS DIRECTORY Utility	
VMS BACKUP Utility	
Device and Volume Manipulation Tools	
VMS ALLOCATE Utility	
VMS INITIALIZE Utility	
VMS MOUNT Utility	
Assorted Utilities	
File Sorting and Merging Utility (SORT/MERGE)	Personal Mail Utility (MAIL)
Information Utilities (SHOW)	Telephone Utility (PHONE)

Table 1–5: Optional Programs and Utilities Available for VMS

Languages	
VAX APL	VAX DSM
VAX BASIC	VAX FORTRAN
VAX BLISS–32	VAX LISP
VAX C	VAX PASCAL
VAX COBOL	VAX PL/I
VAX CORAL 66	VAX RPG II
VAX DIBOL	Professional Host Tool Kit Languages
Development Productivity Tools	
VAX Code Management System (CMS)	VAX Module Management System (MMS)
VAX Language Sensitive Editor (LSE)	VAX Source Code Analyzer (SCA)
DEC Test Manager (DTM)	
Graphics Tools	
VAX ReGIS Graphics Library (RGL)	VAX DECslide Graphic Presentation Utility
VAX DECgraph Plotting Package	VAX Graphics Kernel System (GKS)
Information and Data Management Products	
VAX Common Data Dictionary (CDD)	VAX Forms Management System (FMS)
VAX DATATRIEVE	VAX Terminal Data Management System (TDMS)
VAX Database Management System (DBMS)	Rdb
Office Products	
ALL–IN–1 Office Menu	
VAX DECalc Spreadsheet Calculator	
VAX DECspell	
Networks and Communications Products	
DECnet–VAX	
DECnet/SNA Gateway Products	

User Working Environment

- The software working environment (process) established with the *User Authorization File* (UAF)
- Each user has a record in the UAF
- Each process contains various *informational* parameters, among these:
  - — User name
  - — Password
  - — Account name
  - — Process name
  - — UIC
  - — Process identification number (PID)
  - — Interactive terminal specification (interactive processes only)
  - — Default device and directory specification
- Each process also contains various *control* parameters:
  - — Priority
  - — Resource limits (Quotas)
  - — Privileges
- Many parameters come from the user's UAF record; others are defined or recorded by VMS when the user logs in

Table 1–6: Parameters that Identify Interactive Processes

Parameter	Function	Syntax/ Examples
Account Name	Identifies a user whose computer usage is combined for billing purposes.	1–8 characters `CC2E4`
Default Device and Directory Specification	Names the device and directory where, by default, you keep your files.	(See MODULE 2) `DISK$USER:[SMITH]`
Interactive Terminal/ Specification	Names the terminal you are using.	`TXC3`
Password	Confirms your right to log in.	1–31 characters `phantom`
Process Identification Number (PID)	Identifies your process to the operating system.	8 hexadecimal digits `000004B3`
Process Name	Identifies your process to other users.	1–15 characters `SMITH`

User Identification Code (UIC)	Names the owner of a given mass storage volume (disk pack or tape reel), file, or other data structure; also used to name the owner of a given process.	1–15 alphanumeric characters, or 2 octal numbers separated by a comma, and enclosed by brackets. When using 2 octal numbers, the numbers lie between 0 and 37776, and 0 and 177776, respectively.
		`[SMITH]`
		`[214,377]`

User Name	Names the record in the user authorization file through which you have gained system access.	1–12 characters `SMITH`

Table 1–7: Parameters that Control Interactive Processes

Parameter	Function	Syntax/ Examples
Priority	Determines how soon operations are executed, relative to other processes present on the system.	Decimal integer between 0 and 31

Privileges	Determines which protected system operations you are allowed to perform.	Refer to the *VMS DCL Dictionary* for a complete list of privileges.
		`OPER`
		`GROUP`
		`GRPNAM`

Resource Limits	Determines how much of certain system resources you can consume.	Decimal integers, the range of which depends on the parameter. See the *VMS DCL Dictionary* for a comprehensive list. Open file quota: 20

Example 1–1: Displaying the Values of Your Process Parameters

```
$
$ SHOW PROCESS/ALL

29-FEB-1988 18:31:45.57   TXB4:              User: SMITH
Pid: 00000145   Proc. name: SMITH_1          UIC: [ADMIN,SMITH]
Priority:   4   Default file spec: DISK$USER:[SMITH]

Devices allocated: TXB4:

Process Quotas:
 Account name: PUBLIC
 CPU limit:                        Infinite  Direct I/O limit:        18
 Buffered I/O byte count quota:        4000  Buffered I/O limit:      18
 Timer queue entry quota:                10  Open file quota:         19
 Paging file quota:                    9064  Subprocess quota:         2
 Default page fault cluster:             64  AST quota:               22
 Enqueue quota:                          30  Shared file limit:        0
 Max detached processes:                  0  Max active jobs:          0

Accounting information:
 Buffered I/O count:         176  Peak working set size:         335
 Direct I/O count:            47  Peak virtual size:            2210
 Page faults:               1077  Mounted volumes:                 0
 Images activated:             8
 Elapsed CPU time:       0 00:00:05.62
 Connect time:           0 00:02:00.50

Process privileges:
 TMPMBX                may create temporary mailbox
 NETMBX                may create network device

Process rights identifiers:
 INTERACTIVE
 LOCAL

Process Dynamic Memory Area
    Current Size (bytes)          25600    Current Total Size (pages)      50
    Free Space (bytes)            21744    Space in Use (bytes)          3856
    Size of Largest Block         21712    Size of Smallest Block           8
    Number of Free Blocks             3    Free Blocks LEQU 32 Bytes        2

Processes in this tree:

Bob Smith
  SMITH_1 (*)
$
$
$ SHOW WORKING_SET
  Working Set      /Limit= 200   /Quota= 500    /Extent= 1000
  Adjustment enabled    Authorized Quota= 500  Authorized Extent= 1000
$
$
```

Learning Aids

- Software documentation
  - — Extensive documentation exists for every component of the VMS system
  - — Places to search when looking for a particular topic:

    Overview of VMS Documentation
    VMS Master Index

  - — Other major sources of information on VMS and optional software:

    VMS System Software Handbook
    VMS Software Product Description (SPD)
    VMS Software Languages and Tools Handbook

- Course offerings
  - — DIGITAL offers a wide variety of courses in two formats:

    Lecture/Lab (at DIGITAL facilities or on the customer's site)
    Self–Paced Instruction (SPI)

- Contact your nearest DIGITAL Customer Training Center or your DIGITAL sales representative for more information

SUMMARY

- VAX system configurations can be classified as:
  - — Single processors
  - — Multiple processor configurations
- A single processor configuration is any single VAX processor and its peripheral devices.
- There are three types of multiple processor configurations:
  - — Tightly coupled configurations – multiprocessors
  - — Loosely coupled configurations – networks
  - — VAXcluster Systems – configured midway between multiprocessors and networks
- Multiple processor configurations consist of:
  - — Processors
  - — Interconnects
  - — Peripheral devices
  - — Communication devices
  - — Transmission media
  - — Terminal servers (optional)
- Device names have the form: **ddcu**
- A local area network (LAN) spans a limited geographical area.
- A wide area network (WAN) spans a larger area.
- Important difference between a network and a VAXcluster System:
  - — Sharing of information between nodes is much faster and easier in a cluster due to software–synchronized access to shared resources in a cluster

- Major software components of VMS
  - DCL
  - Processes

    Software context
    Hardware context
    Virtual address space
    Image
  - Memory management subsystem

    Page fault handler
    Modified page writer
    Swapper
  - Scheduler
- Important memory–related system files
  - Paging files
  - Swap files
- Four processor access modes
  - Kernel
  - Executive
  - Supervisor
  - User
- System directories
  - Contain system files
  - Have system–wide logical names assigned to them
  - Contain subdirectories of the system directory with concealed logical name of SYS$SYSROOT
  - Should be suitably protected to ensure security

MODULE 2
MANAGING SYSTEM USERS

INTRODUCTION

The resources of a single VMS operating system are limited. A given configuration has a fixed amount of physical memory and disk storage. It also has a fixed number of peripheral devices. All system users must share these resources.

VMS restricts each user's access to system resources. By controlling these restrictions, the system manager can promote fair and effective resource sharing.

VMS also restricts each user's ability to affect other users and VMS itself. The system manager and individual users can maintain system security by controlling these restrictions.

The principal VMS features that regulate individual processes are:

- Priority
- Privileges
- Limits
- Disk quotas
- Protection
- Access mode

To define restrictions on individual user processes and to record system activity, the system manager uses certain VMS utilities and DCL commands. These utilities and commands create and maintain the following files, which are discussed in this module:

- User authorization file (SYS$SYSTEM:SYSUAF.DAT)
- Quota files on disk volumes ([000000]QUOTA.SYS)
- Login files (for example, SYS$MANAGER:SYLOGIN.COM)

The system manager or operator must also respond to requests for assistance and send messages to users. The manager uses several utilities, including the MAIL and PHONE utilities, to accomplish this.

OBJECTIVES

To divide limited system resources among users and processes, a system manager should be able to:

- Identify the relationships between process characteristics and the information stored in UAF records.
- Add a user account to the system by creating the necessary user authorization records, directories, and disk quotas.
- Remove a user account from the system.
- Regulate the use of system resources.
- Regulate processes running on the system.
- Restrict the abilities of certain users or applications.
- Communicate with system users and operators.

RESOURCES

The following publications are referenced in this module:

1. *VMS DCL Dictionary*
2. *VMS System Manager's Manual*
3. *Guide to Setting Up a VMS System*
4. *Guide to Maintaining a VMS System*
5. *VMS SYSMAN Utility Manual*
6. *VMS Authorize Utility Manual*
7. *Guide to VMS System Security*

DEFINING THE USER ENVIRONMENT

User Authorization File (UAF)

- Users are identified on VMS systems by *user names*
- The system manager defines user names in the *user authorization file* (UAF)
- User name records are often called *accounts*
- Each UAF record consists of a number of *fields*:
  - — User name
  - — Encoded password
  - — Default device and directory
  - — Access restrictions
  - — Quotas and limits
  - — Privileges
  - — UIC and identifiers
- The system manager can modify any record field
- Field values should reflect the needs of users and system resources
- The AUTHORIZE utility is used to manage the UAF:
  - — Specifies who may use the system
  - — Defines the user's default device and directory
  - — Limits the amount of space in memory for each user
  - — Permits interaction between users by establishing groups
  - — Runs a command procedure for any user at login time
  - — Limits the number of files a user may have open at any one time
  - — Sets limits on other resources

Example 2–1: UAF Record

```
$
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF> SHOW SMITH

1 Username:  SMITH                          2 Owner:  MARY SMITH
3 Account:  J11MGT                          4 UIC:    [11,200] ([J11MGT,SMITH])
5 CLI:       DCL                             6 Tables:  DCLTABLES
7 Default:   WORK1:[SMITH]
8 LGICMD:    SYS$MANAGER:GRP11LOGIN
9 Login Flags:  Diswelcome Disnewmail
10Primary days:    Mon Tue Wed Thu Fri
  Secondary days:                       Sat  Sun
  Primary   000000000011111111112222    Secondary  000000000011111111112222
  Day Hours 012345678901234567890123    Day Hours  012345678901234567890123
  Network:  --------##########------              -----  No access  ------
  Batch:    --------##########------              -----  No access  ------
  Local:    ##### Full access ######              ##### Full access ######
  Dialup:   --------##########------              -----  No access  ------
  Remote:   --------##########------              -----  No access  ------
  Expiration:             (none)    Pwdminimum:   6   Login Fails:     0
  Pwdlifetime:          180 00:00   Pwdchange:       (pre-expired)
  Last Login:             (none) (interactive),        (none) (non-interactive)
11Maxjobs:           0  Fillm:        20  Bytlm:      4096
  Maxacct jobs:      0  Shrfillm:      0  Pbytlm:        0
  Maxdetach:         0  BIOlm:         6  JTquota:    1024
  Prclm:             2  DIOlm:         6  WSdef:       150
  Prio:              4  ASTlm:        10  WSquo:       200
  Queprio:           4  TQElm:        10  WSextent:    500
  CPU:         (none)   Enqlm:        10  Pgflquo:   10000
12Authorized Privileges:
    GROUP TMPMBX NETMBX
  Default Privileges:
    TMPMBX NETMBX

UAF> EXIT
%UAF-I-NOMODS, no modifications made to system authorization file
%UAF-I-NAFNOMODS, no modifications made to network authorization file
%UAF-I-RDBNOMODS, no modifications made to rights database
$
$
```

Table 2–1: Fields in a UAF Record Usually Unique to Each User

Field	Meaning	AUTHORIZE Qualifier	Key†
User Name	Identifies user for whom VMS creates process. Typed in response to **Username:** prompt.	User name is a parameter, not a qualifier.	1
Password	Confirms identity of user. Typed in response to `Password:` prompt. Note that it is possible to create an "OPEN" account by specifying a null password (**/PASSWORD=""**). The `PASSWORD:` prompt is not output for OPEN accounts.	`/PASSWORD`	‡
UIC	Used to enforce protection of devices, volumes, and files. Also used to regulate ability of process to communicate with and affect other processes. (UIC may be displayed in a numerical or character string form.)	`/UIC`	4
Owner	Records full name of user, for convenience of the system manager. Not used by VMS. To allow spaces in the name, enclose it in quotation marks.	`/OWNER`	2
Default	Used by VMS to establish both initial default device and initial default directory at login time. Note that it is preferable to identify the default device by a logical name based on the volume label (such as WORK1) rather than by a physical device name (such as DUA1).	`/DEVICE` `/DIRECTORY`	7

†Refer to Example 2–1 for key numbers.

‡Not displayed, to preserve integrity.

Table 2–2: Fields in a UAF Record Usually Common to Groups of Users

Field	Meaning	AUTHORIZE Qualifier	Key†
Account	Used by ACCOUNTING utility to identify processes whose activities are billed in common. To include spaces in the account name, enclose it in quotation marks.	/ACCOUNT	3
CLI	Command language interpreter. Usually, DCL.	/CLI	5
Tables	Specifies which CLI table the process will use. (Create a new CLI table by using the Command Definition utility to copy an existing CLI table. Then use the same utility to modify the new table by removing or adding commands to it.)	/CLITABLES	6
LGICMD	Specifies a system–wide or group–wide login command file. Commands in this file are executed before the user can issue commands interactively or through a batch command file. The manager typically assigns the same login file to all members of a group, or to all users on the system. (If this field is left blank, VMS executes the LOGIN.COM file in the user's default directory before executing user commands.)	/LGICMD	8
Login flags	Special restrictions during and after logging in to the system.	/FLAGS	9
Primary days Secondary days	Access allowed during specific times and days, and with specific access modes.	/ACCESS /PRIMARY /INTERACTIVE /NETWORK /BATCH /LOCAL /DIALUP /REMOTE	10
Limits	Used to restrict use of various resources by process.	One qualifier for each limit. For example: /TQELM /ASTLM	11
Privileges	Privileges to add to user record or remove from user record. Default privileges are given to user at login. Authorized privileges are given when user enters **SET PROCESS/PRIVILEGE** command.	/PRIVILEGES /DEFPRIVILEGES Qualifiers have a keyword for each privilege. For example, GRPNAM (to add this privilege) and NOGRPNAM (to delete it).	12

†Refer to Example 2–1 for key numbers.

AUTHORIZE Utility

- Standard UAF file shipped with VMS
  - — SYS$SYSTEM:SYSUAF.DAT
- Contains four DIGITAL–supplied UAF records:
  - — SYSTEM
  - — FIELD
  - — SYSTEST
  - — DEFAULT

Table 2–3: Standard User Records in the User Authorization File

User Name (and Initial Password) of UAF Record	Purpose of This Record
SYSTEM (MANAGER)[1]	For software installation, system bootstrapping, and system problem diagnosis. Do not log in as SYSTEM for routine system management functions. There are too many dangerous privileges, particularly BYPASS. Create a UAF record with UIC [1,4] and SETPRV, but without dangerous privileges, to use for routine system management functions. NOTE: You should substitute the READALL privilege for BYPASS.
SYSTEST (UETP)[2]	Runs the User Environment Test Package (UETP), which tests VMS hardware and software
FIELD (SERVICE)[2]	Runs hardware diagnostics
DEFAULT (USER)[1]	Not possible to log in as DEFAULT. The AUTHORIZE command ADD uses this record to supply default values for new UAF records.

[1] These records are present whenever a UAF is created. They cannot be removed or renamed. The password of DEFAULT cannot be changed. System security requires that you change the password of SYSTEM.

[2] These records are present in the UAF that is distributed with VMS. It is possible to remove them, but removal is not recommended. System security requires that you change the passwords of SYSTEST and FIELD.

Table 2–4: Starting the AUTHORIZE Utility

Step	DCL Commands	Comments
1	Log in as SYSTEM	You need to be in a system account (group number 0–10) to run the AUTHORIZE utility. To accomplish this, log in using the SYSTEM user name or another user name that specifies a system UAF record, or give yourself the SYSPRV privilege while in a nonsystem process. The SYSPRV privilege allows you to do the same things you can do in a system process.
2	`$ SET DEFAULT SYS$SYSTEM`	The AUTHORIZE utility always manages the SYSUAF.DAT file stored in the current default directory for your process. To affect the system–wide UAF file, SYS$SYSTEM:SYSUAF.DAT, set your default to SYS$SYSTEM.
3	`$ RUN AUTHORIZE`	If the SYSUAF.DAT file does not exist, AUTHORIZE asks whether you want to create a SYSUAF.DAT. If you answer "yes," it creates one. If you know SYS$SYSTEM:SYSUAF.DAT really does exist, answer "no." Then: 1. Make sure that the system logical name SYSUAF points to SYS$SYSTEM:SYSUAF.DAT, if the logical name exists. The module **Customizing the System** discusses how to display and modify system logical names in more detail. 2. Follow steps 1–3 again.

Table 2–5: Summary of AUTHORIZE Commands

Command	Function
ADD	Add a user
MODIFY	Modify a user record
REMOVE	Remove a user
RENAME	Rename a user record
COPY	Copy one user record to another
LIST	Create a file of user names that can be printed
SHOW	Display user records
HELP	List commands and qualifiers available
EXIT	Return to DCL

Figure 2–1: Adding and Using a UAF Record

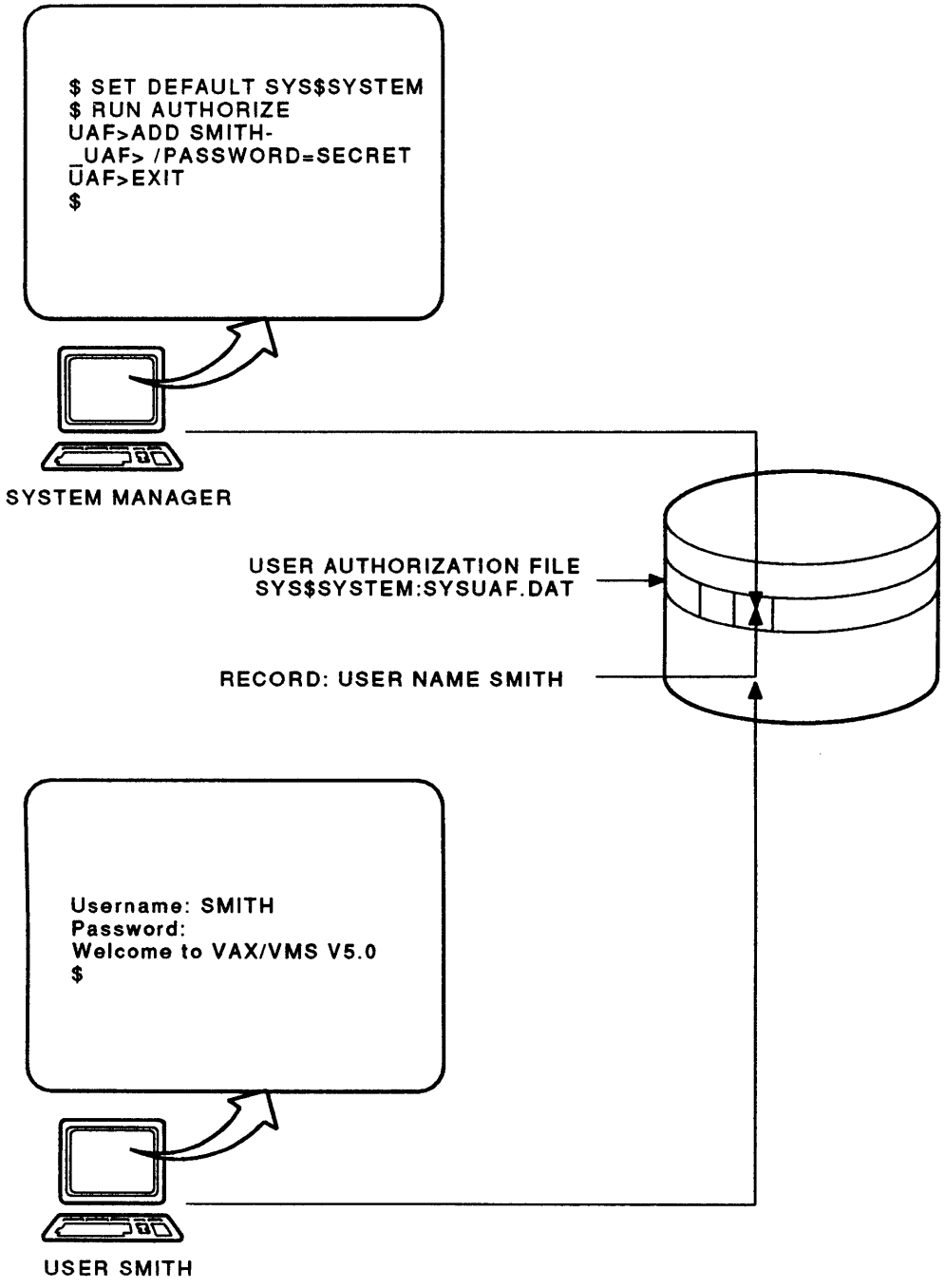


Table 2–6: Managing the User Authorization File with the AUTHORIZE Utility

Operation on the UAF	Command Format and Examples
Displays the complete UAF record for account JONES on your terminal.	`UAF> SHOW user-name` `UAF> SHOW JONES`
Displays an abbreviated list of all accounts in the UAF file on your terminal.	`UAF> SHOW/BRIEF *`
Displays an abbreviated list on your terminal of all accounts starting with the letter "B."	`UAF> SHOW/BRIEF B*`
Writes a brief list of all accounts to the file SYSUAF.LIS.	`UAF> LIST user-name` `UAF> LIST *`
Writes a list of complete values for all accounts starting with the letter "B" to the file SYSUAF.LIS.	`UAF> LIST/FULL B*`
Adds a new user record, copying all values not specified by qualifiers from the DEFAULT record (If the **/PASSWORD** qualifier is omitted, the password "USER" is supplied.)	`UAF> ADD new-user-name [/qualifier...]` `UAF> ADD SMITH /PASSWORD=SECRET`

Adds a new user record, copying all values (except password) not specified by qualifiers from an existing user record (You must specify the password, even if it is to remain the same.)	`UAF> COPY existing-user-name new-user-name -`
	`_UAF> /PASSWORD=password [/qualifier][,...]`
	`UAF> COPY SMITH JONES /PASSW=FRANK`

Modifies an existing user record.	`UAF> MODIFY user-name /qualifier[,...]` `UAF> MODIFY SMITH /DEVICE=WORKDISK`
Modifies all existing user records.	`UAF> MODIFY * /qualifier[,...]` `UAF> MODIFY * /PRCLM=5`
Modifies all existing user records with the same group UIC.	`UAF> MODIFY [group-number,*] /qualifier[,...]` `UAF> MODIFY [310,*] /LGICMD=GRP310:LOGIN.COM`

Renames an existing user record (You must specify the password, even if it is to remain the same.)	`UAF> RENAME existing-user-name -`
	`_UAF> new-user-name /PASSWORD=password`
	`UAF> RENAME PRAUSS MASON /PASSWORD=JAR`

Modifies the DEFAULT record (The qualifier **/PASSWORD** does not affect the default password, which is always "USER.")	`UAF> MODIFY DEFAULT /qualifier[,...]`
	`UAF> MODIFY DEFAULT -`
	`_UAF> /DEV=DISK_SCRATCH /DIR=[GUEST]`

Removes an existing user record.	`UAF> REMOVE user-name` `UAF> REMOVE MASON`

The AUTHORIZE utility lists UAF records alphabetically by user name.

Example 2–2: Brief List of UAF Records

```
$
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF> LIST
%UAF-I-LSTMSG1, writing listing file
%UAF-I-LSTMSG2, listing file SYSUAF.LIS complete
UAF> EXIT
%UAF-I-NOMODS, no modifications made to system authorization file
%UAF-I-NAFNOMODS, no modifications made to network authorization file
%UAF-I-RDBNOMODS, no modifications made to rights database
$
$
$ TYPE SYSUAF.LIS

       Owner      Username        UIC     Account   Privs  Pri  Default Directory

JO ANN ALBERT     ALBERT        [21,5]              Normal   4  WORK1:[ALBERT]
BOB CAMPBELL      CAMPBELL    [101,25]              Normal   4  WORK2:[CAMPBELL]
NANCY CLARK       CLARK        [11,20]              Normal   4  WORK1:[CLARK]
DECNET            DECNET     [240,240]  NETWORK    Normal   4  SYS$SYSDEVICE:[DECNET]
JOHN DOE          DOE         [370,64]  J65        Normal   4  CLASS_DISK:[PAYROLL.DOE64]
FIELD SERVICE     FIELD         [1,10]  SYSTEM     All      4  SYS$SYSROOT:[SYSMAINT]
PETER HARBO       HARBO         [31,7]              Normal   4  WORK1:[HARBO]
SYSTEM DEFAULT    JONES       [200,200]             Normal   4  WORK1:[USER]
                  LIGHT       [101,110]             Normal   4  WORK2:[LIGHT]
                  ROGERS       [21,210]             Normal   4  WORK1:[ROGERS]
SYSTEM DEFAULT    SEMINAR       [53,1]              Normal   4  WORK2:[SEMINAR]
MARY SMITH        SMITH        [11,200]  J26MGT     Group    4  WORK1:[SYSMGR]
SYSTEM MANAGER    SYSTEM         [1,4]   SYSTEM     All      4  SYS$SYSROOT:[SYSMGR]
SYSTEST-UETP      SYSTEST        [1M7]   SYSTEM     All      4  SYS$SYSROOT:[SYSTEST]
                  WALDMAN     [31,310]              Normal   4  WORK2:[WALDMAN]

$
```

Creating a User Account

- Creating accounts for your user environment requires advanced considerations:
  - — Variety of user categories
  - — Security and sharing
  - — Management of auxiliary storage (disks)

Preparing to Create the Account

- Primary considerations when going to add a new user:
  - — How should the user be categorized?
  - — Where should the user's disk files reside by default?
- When the user's requirements are understood, determine nondefault field values:
  - — User name and password
  - — User identification code (UIC)
  - — Default device and directory
  - — Special resource quotas
  - — Additional security parameters
  - — Login command procedures to be executed

Creating the Default Disk Directory

- When a user logs in, VMS uses the UAF record to define the default device and directory
  - — Called the *user file directory* (UFD)
  - — Usually named similar to the user name
  - — The logical name SYS$LOGIN created to point to the UFD
- AUTHORIZE does **not** create the default directory
  - — System manager creates the default directory using DCL commands (or a DCL procedure)
- Each disk volume contains a *master file directory* (MFD)
  - — The "top-level" directory on the volume
  - — Contains UFDs (and possibly other files)
- A corresponding entry in the volume's disk quota file must be created
  - — Only if disk quotas are enabled on the volume

Setting the Disk Quotas

- The System Management (SYSMAN) Utility contains DISKQUOTA commands
  - — Creates and maintains disk quota files
  - — Assigns disk quotas to users
  - — Monitors disk usage
- SYSMAN performs operations on the device name by the /DEVICE qualifier

Table 2–7: Creating a User's Default Directory or Other UFD

Step	Commands	Comments
1	`$ SET DEFAULT volume-name`	If the volume is not mounted, follow the instructions for mounting it in MODULE 4. Use the name of the device where the volume is loaded or the logical name of the volume for the volume–name part of this command. More detailed information on device names and volumes are presented in MODULE 4.

2	`$ CREATE/DIRECTORY -`
	`_$ /OWNER=uic ufd-name -`
	`_$ /PROTECTION=(prot-mask)`
		The UIC supplied with the /**OWNER** qualifier should be the user's own UIC, as recorded in the UAF. A user should always own a private UFD (one that is not shared with anyone else). The /**OWNER** qualifier requires SYSPRV. (See Figure 2–2)
		NOTE: Users may not create their own UFD unless they have write access to the Master File Directory (MFD) for the volume. Also, be sure to protect the new UFD with the appropriate protection mask.

3	`$ DIR volume-name:[000000]`	List the contents of the MFD to check that you did create a User File Directory in it, or set your default to [000000] on the volume and list the name of the directory as shown in Figure 2–2.

Figure 2–2: Creating a User File Directory (UFD)

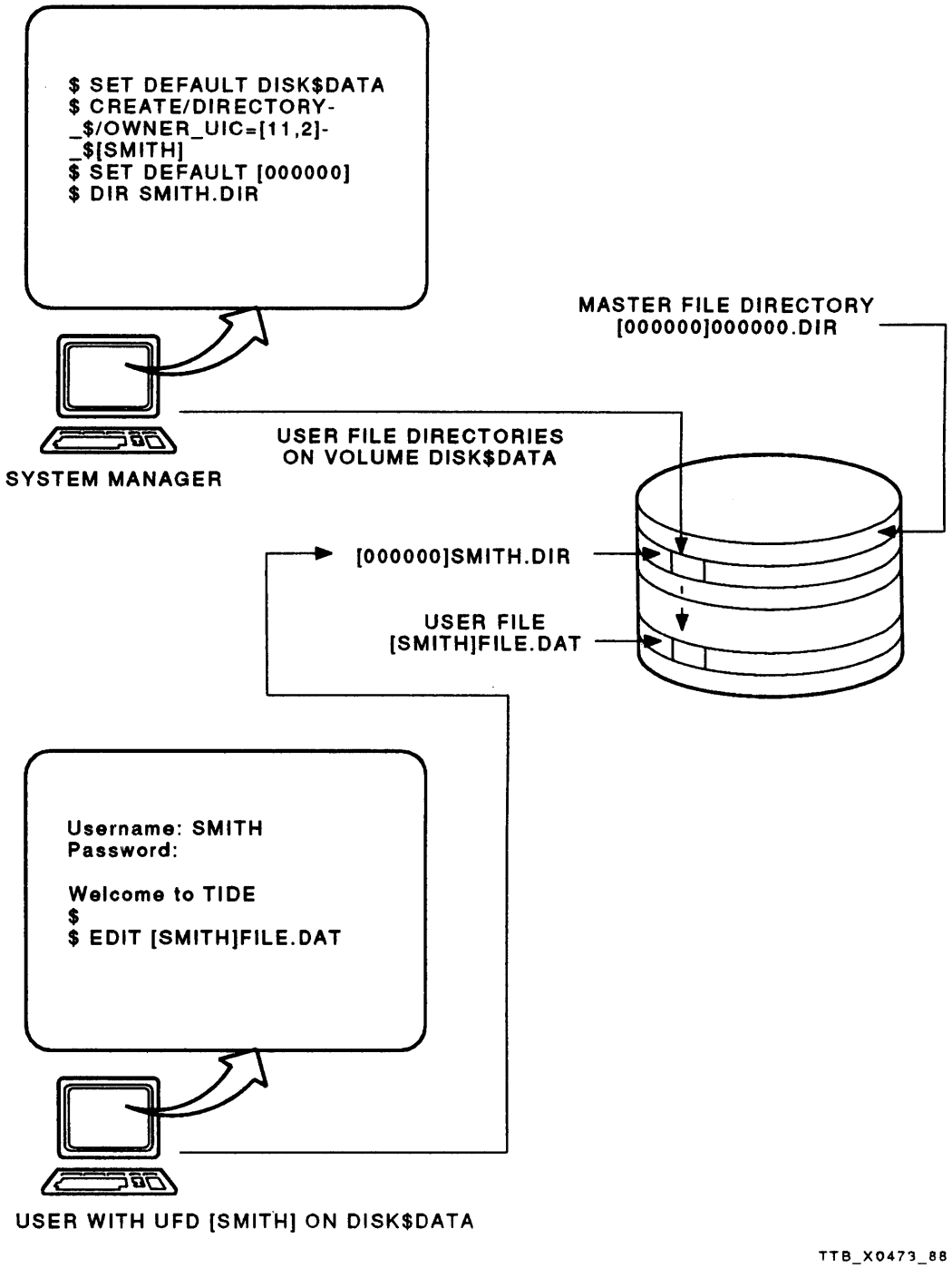


Table 2–8: Basic Steps to Add a New User to the System

Command†	Comments
`$ SET DEFAULT SYS$SYSTEM`	Set your default disk directory to the system directory in which SYSUAF.DAT resides.

`$ RUN SYSMAN`
`SYSMAN> DISKQUOTA ADD [11,15] -`
`_SYSMAN> /DEVICE=DISK$USER /PERMQUOTA=200 -`
`_SYSMAN> /OVERDRAFT=50`
	Invoke the System Management (SYSMAN) utility to assign the user disk quota.

`$ RUN AUTHORIZE`	Invoke the AUTHORIZE utility.

`UAF> ADD SMITH/PASSWORD=PERCHANCE -`
`_UAF> /DEVICE=DISK$USER/DIRECTORY=[SMITH] -`
`_UAF> /UIC=[ADMIN,SMITH]`
	Add the new account SMITH, using the default UAF record values for those fields not specified in the ADD command. ADMIN and SMITH identifiers should exist in order for /UIC qualifier to work.

`UAF> EXIT`	Exit from the AUTHORIZE utility and return to DCL.

`$ CREATE/DIRECTORY USER1:[SMITH] -`
`_$ /OWNER=[ADMIN,SMITH] -`
`_$ /PROTECTION=(S:RWED,O:RWE,G:RE,W:E)`
	Create the user's default disk directory on the default device as specified when the UAF record was created. Note that the disk directory can be created before the UAF record is created. The protection mask given to the new directory is only an example; the exact protection mask should reflect your particular system's security policy.

†You must be logged in to either the SYSTEM account or an account with the SYSPRV privilege enabled.

Example 2–3: Adding a System User

```
$
$ SET DEFAULT SYS$SYSTEM
$ RUN SYSMAN
SYSMAN> DISKQUOTA ADD [11,15] /DEVICE=DISK$USER /PERMQUOTA=500 /OVERDRAFT=100
SYSMAN> EXIT
$
$ CREATE/DIRECTORY /OWNER_UIC=[11,15] DISK$USER:[BROWN]
$
$ RUN AUTHORIZE
UAF> ADD BROWN /PASSWORD=ENIGMA /UIC=[11,15] -
_UAF> /DEVICE=DISK$USER /DIRECTORY=[BROWN] -
_UAF> /OWNER="Mary Brown" /ACCOUNT=GRP11
%UAF-I-ADDMSG, user record successfully added
%UAF-I-RDBADDMSGU, identifier BROWN value: [000011,000015] added to RIGHTSLIST.DAT
UAF> EXIT
%UAF-I-DONEMSG, system authorization file modified
%UAF-I-RDBDONEMSG, rights database modified
$ LOGOUT

             (Now try logging into the new BROWN account)

Username:  BROWN
Password:
             Welcome to VMS Version 5.0

$ SHOW PROCESS
17-NOV-1988 14:23:22.80   TTC1:        User:  BROWN
Pid:  00001231   Proc. name:  BROWN    UIC:  [GRP11.BROWN]
Priority:  4  Default file spec:  DISK$USER:[BROWN]

Devices allocated:  TTC1:
$
$ SHOW DEFAULT
  DISK$USER:[BROWN]
$
$ SHOW QUOTA
  User [BROWN] has 6 blocks used, 494 available, of 500
  authorized and permitted overdraft of 100 blocks on DISK$USER
$
$
```

Modifying the DEFAULT Record

- The AUTHORIZE **ADD** command is similar in function to the **COPY** command.
  - — Simulates copying of the DEFAULT record for the new user.
- The DEFAULT record is used as a template to specify most field values.
  - — The password field should **always** be specified when using the **ADD** command.
- Modify the DEFAULT record to reflect typical user account requirements.

Tailoring User Accounts

UAF record fields fall into four basic categories:

1. Identification and environment
2. Access and security
3. Quotas and resource limits
4. Privileges

Example 2–4: UAF Record Field Categories

```
1 Username:  SMITH                              Owner:  MARY SMITH
  Account:  J11MGT                              UIC:    [11,200] ([J11MGT,SMITH])
  CLI:      DCL                                  Tables:  DCLTABLES
  Default:  WORK1:[SMITH]
  LGICMD:   SYS$MANAGER:GRP11LOGIN
2 Login Flags:  Diswelcome Disnewmail
  Primary days:    Mon Tue Wed Thu Fri
  Secondary days:                       Sat  Sun
  Primary   000000000011111111112222    Secondary  000000000011111111112222
  Day Hours 012345678901234567890123    Day Hours  012345678901234567890123
  Network:  --------##########------              -----  No access  ------
  Batch:    --------##########------              -----  No access  ------
  Local:    ##### Full access ######              ##### Full access ######
  Dialup:   --------##########------              -----  No access  ------
  Remote:   --------##########------              -----  No access  ------
  Expiration:            (none)    Pwdminimum:   6   Login Fails:     0
  Pwdlifetime:         180 00:00   Pwdchange:       (pre-expired)
  Last Login:            (none) (interactive),        (none) (non-interactive)
3 Maxjobs:          0  Fillm:        20  Bytlm:      4096
  Maxacct jobs:     0  Shrfillm:      0  Pbytlm:        0
  Maxdetach:        0  BIOlm:         6  JTquota:    1024
  Prclm:            2  DIOlm:         6  WSdef:       150
  Prio:             4  ASTlm:        10  WSquo:       200
  Queprio:          4  TQElm:        10  WSextent:    500
  CPU:         (none)  Enqlm:        10  Pgflquo:   10000
4 Authorized Privileges:
    GROUP TMPMBX NETMBX
  Default Privileges:
    TMPMBX NETMBX
```

Notes on Example 2–4

In Example 2–4, each of the following field categories are grouped together, starting with a numbered identifier:

1 Identification and environment fields

2 Access and security fields

3 Quotas and resource limits fields

4 Privileges fields

Identification and Environment Fields

- Certain UAF record fields are used for:
  - — Accounting
  - — User identification
  - — Environment initialization

Table 2–9: AUTHORIZE Qualifiers for Identification and Environment Fields

Qualifier	Function
`/ACCOUNT=account-name`	A 1 to 8 alphanumeric character string identifying the account.
`/CLI=cli-name`	Name of the default command language interpreter.
`/CLITABLES=table-name`	Specifies user–defined CLI tables for the account.
`/DEVICE=device-name`	Specifies the default disk device at login.
`/DIRECTORY`	Specifies the default disk directory at login.
`/LGICMD=file-spec`	Specifies the name of the login command procedure to be executed after the system–wide login procedure is executed (if one exists).
`/OWNER=owner-name`	Specifies a 1 to 31 character string identifying the owner of the account.
`/UIC=uic`	Specifies the user identification code (UIC) for the account.

Login Command Procedures

- Define the user environment using *login command procedures*
  - — System–wide login command procedure
  - — Other login command procedures (created by system manager or user)
- Login command procedures are executed each time an interactive process or batch job is created
- VMS finds these command procedures in:
  - — System logical name SYS$SYLOGIN (system–wide procedure)
  - — LGICMD field in each user's UAF record
  - — Default user login file LOGIN.COM
- VMS login command procedure execution sequence:
  - — If SYS$SYLOGIN defined, execute command file it designates
  - — If LGICMD contains name of command procedure, execute it
  - — If LGICMD blank, execute SYS$LOGIN:LOGIN.COM
- Four common uses of login command procedures:
  - — Personal login only
  - — System and personal login
  - — System and group login
  - — Captive login

Table 2–10: Typical Login Command Procedures (DCL)

Function of Login Command Files	Definition of System Logical Name SYS$SYLOGIN and UAF Record Field LGICMD
Personal login only: DCL users create the file LOGIN.COM in their own login default directories to customize their own environments.	SYS$SYLOGIN undefined LGICMD undefined
System and personal login: The system manager creates the DCL command file SYS$MANAGER:SYLOGIN.COM to customize a common user environment for all users on the system, and users each create a file, LOGIN.COM, in their own login default directories to customize their own environments.	SYS$SYLOGIN defined as SYS$MANAGER:SYLOGIN LGICMD undefined
System and group login: The system manager creates the DCL command file SYS$MANAGER:SYLOGIN.COM to customize a common user environment for all users on the system, and group managers create a DCL command file to customize a common user environment for all users in their group.	SYS$SYLOGIN defined as SYS$MANAGER:SYLOGIN LGICMD defined as the file created for the group that the user belongs to (see Example 2–4).
Captive login: The system manager creates a file in the SYS$MANAGER directory, such as SYS$MANAGER:CAPTIVE.COM, to customize the user environment and prevent a user from changing it. Typically, the command procedure examines each DCL command the user enters, to decide whether or not to allow it to be executed. It may even implement a private command language for the user. Turnkey accounts may also use the captive login method. Typically, more than one person uses a turnkey account. When users log in to a turnkey account, the login procedure runs a program for them, and they communicate with that program. Normally, a user of a turnkey account never sees the DCL prompt (see Example 2–5).	SYS$SYLOGIN definition optional; affects captive and non–captive users. LGICMD contains the name of a captive command procedure, such as SYS$MANAGER:CAPTIVE, or a logical name translating to the captive command procedure name. A captive command procedure must contain a loop to prevent it from exiting, and the FLAGS field of the UAF record must specify the CAPTIVE, DISCTLY, and LOCKPWD flags. Table 2–13 discusses the FLAGS field of the UAF record further.

Example 2–5: Using a Turnkey Account

```
Username:  SPECULATE
Password:

          Welcome to SPECULATE

Speculate>  USE INVESTMENT DATA
Spec:  Consider it done.
Speculate>  COMPUTE FOR NEXT 10 YEARS
SPEC:  Please specify commodity.
Speculate>  GOLD
Spec:  $933,999,456,657.32
Speculate>
             .
             .
             .
```

(User continues to interact with the SPECULATE program)

```
             .
             .
             .
Speculate>  BYE
SPECULATE logged out at 20-OCT-1983  16:00:15.16
```

Access and Security Fields

- Used to limit the use of the account according to time/day or access mode
- Used to limit certain capabilities once logged into the system
- Used to authenticate user requests for access to files and other resources

Access Times and Modes

- You can limit an account's access to the system in three ways:
  - — Time of the day
  - — Day of the week
  - — Access mode
- You can combine these values to further specify the account's ability to gain access to the system:

```
UAF> MODIFY SMITH/LOCAL=(PRIMARY, 8-17, SECONDARY, 8-17)
UAF> MODIFY SMITH/DIALUP=(PRIMARY, 17-8, SECONDARY, 17-8)
```

Table 2–11: Login Access Modes

Mode	Description
INTERACTIVE	Any kind of interactive login
LOCAL	Directly connected terminals (no LATs or modems)
DIALUP	Modem connections using telephone services (or network services emulating telephone services)
REMOTE	Virtual terminal connection across DECnet
BATCH	Batch jobs (noninteractive access method)
NETWORK	DECnet noninteractive network access, for example: file transfers, electronic mail to/from other nodes, etc.

Table 2–12: AUTHORIZE Qualifiers for Access Fields

Qualifier	Function
`/ACCESS[=(range[,...])`	Specifies the hours of access for all modes of access.
`/BATCH[=(range[,...])`	Specifies hours of access permitted for batch jobs.
`/DIALUP[=(range[,...])`†	Specifies hours of access permitted for dialup jobs.
`/INTERACTIVE[=(range[,...])`	Specifies hours of access permitted for interactive logins.
`/LOCAL[=(range[,...])`†	Specifies hours of access permitted for interactive logins initiated on local terminals.
`/PRIMEDAYS=([NO]day[,...])`	Specifies the primary and secondary days of the week for logins. Specify primary days as MON, TUE, WED, THU, FRI, SAT, and SUN. Specify secondary days as NOMON, NOTUE, NOWED, etc.
`/NETWORK[=(range[,...])`	Specifies hours of access permitted for network batch jobs.
`/REMOTE[=(range[,...])`†	Specifies hours of access permitted for interactive logins initiated by network remote terminals.

†These are interactive logins, so you can use the /INTERACTIVE qualifier to specify all three interactive access methods.

Login Flags

Login flags are used to restrict certain activities of the user job.

Table 2–13: Login Flag Parameters

/FLAG Parameter†	Purpose
AUDIT	Audits all security–relevant actions
AUTOLOGIN	Restricts this account to autologins only
CAPTIVE	Prevents user from changing any defaults at login
DEFCLI	Prevents user from changing default CLI or CLI table
DISCTLY	Disables CTRL/Y interrupts
DISFORCE_PWD_CHANGE	Disables forced user expired password changes
DISMAIL	Prevents mail delivery to this user
DISNEWMAIL	Suppresses "New Mail..." announcements
DISRECONNECT	Disables automated reconnections
DISREPORT	Disables time of last login and other security reports
DISUSER	Disables this account completely
DISWELCOME	Suppresses "Welcome to..." login message
GENPWD	Requires user to use generated passwords
LOCKPWD	Prevents user from changing password
PWD_EXPIRED	Marks password as expired
PWD2_EXPIRED	Marks second password as expired

†Any flag can be prefixed with "NO" to turn off the flag's intended purpose, for example: **/NOLOCKPWD**

Security Fields

Security fields are used to authenticate user requests for access to files and other resources

Table 2–14: AUTHORIZE Qualifiers for Security Fields

Qualifier	Function
`/ADD_IDENTIFIER`	Adds identifiers for the user name and account name to the rights database.
`/EXPIRATION=time`	Expiration date and time of the account.
`/GENERATE_PASSWORD[=keyword]`	Invokes the password generator to generate user passwords. Details of the possible keywords are discussed in the module **Maintaining System Integrity**.
`/MODIFY_IDENTIFIERS`	Specifies whether the identifier associated with a user record is to be modified in the rights database.
`/PASSWORD=(pwd1[,pwd2])`	Specifies the primary and optional secondary passwords.
`/PWDEXPIRED`	Specifies whether a password is valid only for the first login.
`/PWDLIFETIME=time`	Specifies the length of time a password is valid, entered as a delta–time value.
`/PWDMINIMUM=value`	Specifies the minimum number of characters allowed for a password.
`/REMOVE_IDENTIFIER`	Specifies whether the user name and account name identifiers should be removed from the rights database when the UAF record is removed from SYSUAF.DAT. Works only for the **REMOVE** command.

Table 2–15: AUTHORIZE Qualifiers for Quota Fields

Qualifier	Function
/ASTLM=value	Number of ASTs the user can have queued at any one time.
/BIOLM=value	Maximum number of buffered I/O operations the user can have outstanding at any one time.
/BYTLM=value	Maximum number of bytes of nonpaged system dynamic memory that the user's job may consume at any one time.
/CPUTIME=time	The maximum CPU time a user's process can take per session, specified as a delta–time value.
/DIOLM=value	Maximum number of direct I/O operations (usually disk) that the user can have outstanding at any one time.
/ENQLM=value	Maximum number of locks that can be queued at any one time.
/FILLM=value	Maximum number of files that can be open at one time.
/JTQUOTA=value	The initial maximum number of bytes with which the job–wide logical name table is to be created.
/MAXACCTJOBS=value	Maximum number of batch, interactive, and detached processes that may be active at any one time for all users of the account. The default value of 0 represents an unlimited number.
/MAXDETACH=value	Maximum number of detached processes allowed at any one time.
/MAXJOBS=value	Maximum number of batch, interactive, detached, and network processes that may be active at any one time.
/PGFLQUOTA=value	Maximum number of pages the user's process can use in the system paging files.
/PRCLM=value	Maximum number of subprocesses that can exist at one time for the user's process.
/PRIORITY=value	The default base priority for all processes created by the user.
/SHRFILLM=value	Maximum number of shared files the user may have open at any one time.
/TQELM=value	Total number of entries in the timer queue, plus the number of temporary common event flag clusters the user can have at any one time.
/WSDEFAULT=value	The number of pages in the user's default working set.
/WSEXTENT=value	The number of pages in the user's working set extent.
/WSQUOTA=value	The number of pages in the user's working set quota.

Privileges

- Two sets of privileges are specified in each UAF record:
  - — *Authorized* privileges
  - — *Default* privileges
- Authorized privileges
  - — Privileges enabled only by explicit use of the DCL command **SET PROCESS/PRIVILEGE**
- Default privileges
  - — Privileges automatically enabled once the user has logged in

Table 2–16: AUTHORIZE Qualifiers for Privilege Fields

Qualifier†	Function
`/DEFPRIVILEGES=([NO]privname[,...])`	Specifies the list of privileges that are enabled at login time. The keyword **[NO]ALL** disables or enables all user privileges.
`/PRIVILEGES=([NO]privname[,...])`	Specifies the list of privileges granted (but not enabled) at login time.

†Any privilege keyword used with either qualifier may be prefixed with "NO" to turn off the privilege.

Table 2–17: VMS Privileges

Privilege	Description
ACNT	May suppress accounting message
ALLSPOOL	May allocate spooled device
ALTPRI	May set any priority value
BUGCHK	May make bug check log entries
BYPASS	Bypasses UIC checking
CMEXEC	May change mode to exec
CMKRNL	May change mode to kernel
DETACH	May create detached processes
DIAGNOSE	May diagnose devices
EXQUOTA	May exceed quota
GROUP	May affect other processes in same group
GRPNAM	May insert in group logical name table
GRPPRV	Group access via system protection
LOG_IO	May do logical I/O
MOUNT	May execute mount ACP function
NETMBX	May create network device
OPER	Operator privilege
PFNMAP	May map to specific physical pages
PHY_IO	May do physical I/O
PRMCEB	May create permanent common event clusters
PRMGBL	May create permanent global sections
PRMMBX	May create permanent mailbox
PSWAPM	May change process swap mode
READALL	May read anything as the owner

Table 2–17: VMS Privileges (Cont.)

Privilege	Description
SECURITY	May perform security functions
SETPRV	May set any privilege bit
SHARE	May assign channels to nonshared device
SHMEM	May create/delete objects in shared memory
SYSGBL	May create system–wide global sections
SYSLCK	May lock system–wide resources
SYSNAM	May insert in system logical name table
SYSPRV	May access objects via system protection
TMPMBX	May create temporary mailbox
VOLPRO	May override volume protection
WORLD	May affect other processes in the world

Removing a User Account

- Basic steps for most systems:
  - — Remove the UAF record
  - — Dispose of any remaining disk files
  - — Remove the user's entries in volume quota files (if any)
  - — Remove associated mail information for the account
- All steps require that the SYSPRV privilege be enabled
- Removing the UAF record:
  - — Use the AUTHORIZE **REMOVE** command
- Disposing of remaining disk files:
  - — Save any necessary files using the BACKUP utility (or any other acceptable method, based on your site's data management policies)
  - — Delete all files
- Removing quota file entries on volumes used by the user:
  - — Use the SYSMAN utility to remove entries from each volume (requires OPER privilege)

```
$ SET DEFAULT SYS$SYSTEM
$ RUN SYSMAN
SYSMAN> DISKQUOTA REMOVE uic /DEVICE=volume-name
SYSMAN> EXIT
$
```

- Removing mail information:
  - — Use the MAIL utility to remove the user's mail address

```
$ MAIL
MAIL> REMOVE username
MAIL> EXIT
$
```

Example 2–6: Command Procedure to Delete a UFD and Its Subdirectories

```
$! DELTREE.COM (P1 = name of device, P2 = name of directory)
$!
$!   Command procedure to delete a UFD and all its subdirectories.
$!   Procedure deletes files from bottom of specified directory
$!   structure to top of structure.
$!   Requires SYSPRV privilege or a system UIC
$!   Back up all useful files before running this procedure.
$!
$    SET NOON
$!
$!   Get name of directory structure to delete if not already known
$!
$   IF P1 .EQS. "" THEN INQUIRE P1 "Device, omit colon"
$   IF P2 .EQS. "" THEN INQUIRE P2 "Directory (UFD, omit brackets)"
$!
$!    Set protection to allow deletion of all files in structure
$!
$   SET PROTECTION=(S:RWED)  'P1':['P2....]*.*;*
$!
$!    Set up counter and loop label
$!
$   COUNTER=8
$LOOP:
$!
$!   Delete files
$!
$   DELETE/LOG 'P1':['P2...]*.*;*
$!
$!   If more files, delete them also
$!
$   COUNTER=COUNTER-1
$   IF COUNTER .GT. 0 THEN GOTO LOOP
$!
$!     Delete UFD from MFD
$!
$   SET PROTECTION=(S:RWED) 'P1':[000000]'P2.DIR
$   DELETE/LOG 'P1':[000000]'P2.DIR;1
$
```

Steps for Removing a User

To remove a user, log in as SYSTEM, or be sure your process the SYSPRV privilege enabled. If you employ disk quotas, then you will also need OPER privilege. The following comments are keyed to Example 2–7.

1 Remove Mary Brown's UAF record. (NOTE: If Mary is just on a long vacation, you should instead disable her account by entering the MODIFY command with the qualifier /**FLAGS=DISUSER**, rather than remove it using these instructions.)

2 Remove Mary Brown's UFD and subdirectory files on her default volume, using the command procedure DELTREE.COM shown in Example 2–6. (Before this step, you should examine her files and back up those you want to save, or transfer them to another user's directory.)

3 Remove Mary Brown's record in her default volume's quota file.

NOTE

If Mary has quota records in quota files on other volumes, you should also remove those records. However, do not remove any quota records if Mary's UIC is not unique; if you remove the record, then other users of that UIC will not be able to use space on that volume.

4 Remove Mary Brown's mail information.

Example 2–7: Removing a System User

```
   $
   $ SET DEFAULT SYS$SYSTEM
   $
   $ RUN AUTHORIZE
1  UAF> REMOVE BROWN
   %UAF-I-REMMSG, record removed from SYSUAF.DAT
   %UAF-I-RDBREMMSGU, identifier BROWN value: [000011,000015] removed from RIGHTSLIST.DAT
   UAF> EXIT
   %UAF-I-DONEMSG, system authorization file modified
   %UAF-I-RDBDONEMSG, rights database modified
   $
2  $ @DELTREE DISK$USER BROWN
   $
   $ RUN SYSMAN
3  SYSMAN> DISKQUOTA REMOVE [11,15] /DEVICE=DISK$USER
   SYSMAN> EXIT
   $ MAIL
4  MAIL> REMOVE BROWN
   MAIL> EXIT
   $
```

MANAGING USERS ON A DAILY BASIS

- Common daily management activities:
  - — Managing user processes
  - — Managing disk space
  - — Communicating with users

Table 2–18: Methods of Regulating VMS Processes

Method of Regulation	Comments
Access to CPU	
Priority	VMS uses process priority to schedule process execution. Priority ranges from 0 (lowest) to 31 (highest). Timesharing processes have priorities 0–15. VMS can automatically adjust the priorities of timesharing processes to promote greater CPU sharing. Real–time processes have priorities 16–31. VMS does not automatically adjust the priorities of real–time processes.
Ability to affect VMS and system users	
Privileges	In many cases, exercising a privilege results in the use of physical memory or other system resources. There is no restriction on the number of times a process can exercise a privilege.
Ability to use certain system resources	
Limits	Limits restrict the amount of a system resource a process can use at a given time. Limits are expressed numerically. Limits are also called quotas in some documentation. Do not confuse limits with disk quotas described below.
Ability to store files on a disk volume	
Disk quotas	Disk quotas restrict the number of blocks of storage a process can use on a volume. Disk quotas are established on volumes (which can be moved from device to device) rather than on devices. Disk quotas do not have to be established for every volume on a system; you can select which volumes have this restriction.

Table 2–18: Methods of Regulating VMS Processes (Cont.)

Method of Regulation	Comments
Access to devices, volumes, files, and interprocess communications	
Protection	Every process has a UIC. Devices, volumes, files, and interprocess communication structures have an owner UIC and a protection code. One type of VMS protection compares the process UIC with the owner UIC and protection code of the protected structure. Another type of VMS protection compares a user's access rights list with the access control list that belongs to the protected structure.
Ability to modify or delete certain data structures, such as logical names	
Access mode	Data structures created at more powerful access modes can be used at less powerful access modes; however, they cannot be modified or deleted at less powerful access modes.
Ability to use or not use certain DCL commands	
CLI tables	If a command is listed in the default DCL command table for the system, SYS$LIBRARY:DCLTABLES.EXE, all users on the system can use the command. However, you may create additional CLI tables and assign them to particular users to broaden or restrict their use of DCL commands.

Restricting User Activity Using DCL Commands

- A process can always control itself and its subprocesses
- Process control over subprocesses might be used for:
  - — Handling several subprocesses simultaneously
  - — Exception handling for subprocess problems
- Processes can be restricted by:
  - — Changing their execution priority
  - — Suspending their execution
  - — Resuming their execution
  - — Stopping execution and deleting the process

Table 2–19: Controlling Processes

Operation	Command Format and Examples	Comments

Suspending a process	`$ SET PROCESS/SUSPEND -`
	`_$ [/ID=pid] [process-name]`
	`$ SET PROCESS/SUSPEND/ID=21A`
	`$ SET PROCESS/SUSPEND JONES`
		Every process has a unique process identification (PID), a hexadecimal number. To refer to a process by PID, as shown in the example, use the **/IDENTIFICATION** qualifier. The command **SHOW SYSTEM** displays PIDs. You are not required to enter leading zeroes.

Resuming a suspended process	`$ SET PROCESS/RESUME -`
	`_$ [/ID=pid] [process-name]`
	`$ SET PROCESS/RESUME/ID=21A`
	`$ SET PROCESS/RESUME JONES`	

Changing the base priority of a process	`$ SET PROCESS/PRIORITY=n -`
	`_$ [/ID=pid] [process-name]`
	`$ SET PROCESS/PRIORITY=4 /ID=21A`
		Every process has a process name, unique within its UIC group. To refer to a process in your UIC group by process name, supply the name as the command parameter. To refer to a process not in your UIC group, you must use its PID, not its process name.

Stopping and deleting a process	`$ STOP [/ID=pid] [process-name]` `$ STOP/ID=21A`	

Restricting the Use of Disk Space

- UAF restricts use of many system resources
- No value in the UAF record restricts use of disk space
- Disk space restriction is handled through *disk quotas*
  - — Quotas are based on UICs, not individual user names
- Disk quotas are managed through use of the SYSMAN utility
  - — **DISKQUOTA** command subset
- Disk quotas are enabled on a volume–by–volume basis
  - — Default is no disk quotas enabled
- Quota files
  - — One file per enabled volume: [000000]QUOTA.SYS
  - — Contains *quota entries,* one per UIC
  - — Created and manipulated by SYSMAN

SYSMAN Utility and DISKQUOTA Functions

- SYSMAN serves many system management functions
  - — Used primarily for VAXcluster systems
- SYSMAN **DISKQUOTA** command subset
  - — Used to create, enable, disable, and delete quota files
  - — Used to add, modify, and remove quota entries
- To run SYSMAN (must have OPER privilege enabled):

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN>
```

- To get help on SYSMAN **DISKQUOTA** commands:

```
SYSMAN> HELP DISKQUOTA
```

Table 2–20: DISKQUOTA Commands Within the SYSMAN Utility

Command Format†	Description
`DISKQUOTA ADD uic`	Adds an entry to a disk quota file and initializes its usage count to zero.
`DISKQUOTA CREATE`	Creates and enables a quota file for a disk volume that does not currently contain one. It is strongly recommended that you do **NOT** create and enforce quotas on the system disk.
`DISKQUOTA DISABLE`	Suspends the maintenance and enforcement of disk quotas on a volume.
`DISKQUOTA ENABLE`	Resumes quota enforcement on a disk volume containing an existing quota file.
`DISKQUOTA MODIFY uic`	Changes an entry in a quota file, or adjusts default values for quotas and overdrafts. Default values for entries on the volume are taken from the entry [0,0].
`DISKQUOTA REBUILD`	Updates a quota file, adding new UICs and correcting usage counts for each user on the volume.
`DISKQUOTA REMOVE uic`	Deletes an entry from the quota file.
`DISKQUOTA SHOW`	Displays quotas, overdrafts, and usage counts.
`EXIT`	Exits user from the SYSMAN utility.
`HELP DISKQUOTA`	Obtains help information on the DISKQUOTA commands.

†For all of these commands, be sure to use the qualifier **/DEVICE=volume–name** to specify the proper volume; otherwise SYSMAN will use either the last volume specified with the most recent **/DEVICE=volume–name**, or the volume associated with your current default directory (if you had not yet used **/DEVICE=volume–name** in a **DISKQUOTA** command.)

Establishing Quotas on a Volume

- A quota file must be created in the volume's MFD (directory [000000])
- Exact steps for properly creating the quota file depend on whether:
  - The volume has just been created (no user files exist yet)
  - The volume has been in use for a while (user files already exist)
- One entry must be created for each UIC allowed to use the volume
- Each quota entry contains the following fields:
  - UIC
  - Usage
  - Permanent Quota
  - Overdraft
- **Quotas should not be enabled on the system disk**

Table 2–21: Fields in a Quota File Record

Field	Meaning	DISKQUOTA Qualifier
UIC	Identifies the user who is permitted to use the volume. Note that files are owned by UICs, not by user names. Therefore, if more than one user shares the same UIC, all of them have the same access to files. They also share the quota assigned to that UIC for the volume. When you log in, VMS reads your UAF record to determine your UIC.	Specify the UIC as a parameter, not as a qualifier, in DISKQUOTA commands.
Usage	Shows the number of blocks of storage this UIC owns.	None. This value is updated by VMS as files are created by the UIC. It is not assigned by the system manager.
Permanent Quota	Determines the number of blocks of storage this UIC can own before VMS refuses to create new files or extend existing files. If the UIC has an Overdraft value greater than 0, a user with this UIC can retry the file operation (create or extend).	`/PERMQUOTA`
Overdraft	Determines the number of blocks above the permanent quota this UIC can own before VMS refuses to create new files or extend existing files. Therefore, the permanent quota plus the overdraft defines the total number of blocks available to a user on a volume.	`/OVERDRAFT`

Table 2–22: Establishing Quotas on a New Volume Called DISK$DATA

Steps	Commands	Comments
1	Log in as SYSTEM	You can alternatively give your current process the OPER privilege to use the SYSMAN utility.
2	`$ RUN SYS$SYSTEM:SYSMAN`	Invoke the SYSMAN utility.
3	`SYSMAN> DISKQUOTA CREATE -` `_SYSMAN> /DEVICE=DISK$DATA`	Creates the file DISK$DATA:[000000]QUOTA.SYS and automatically enables quotas on the volume.

Figure 2–3: Adding a Quota Record to a Volume Quota File

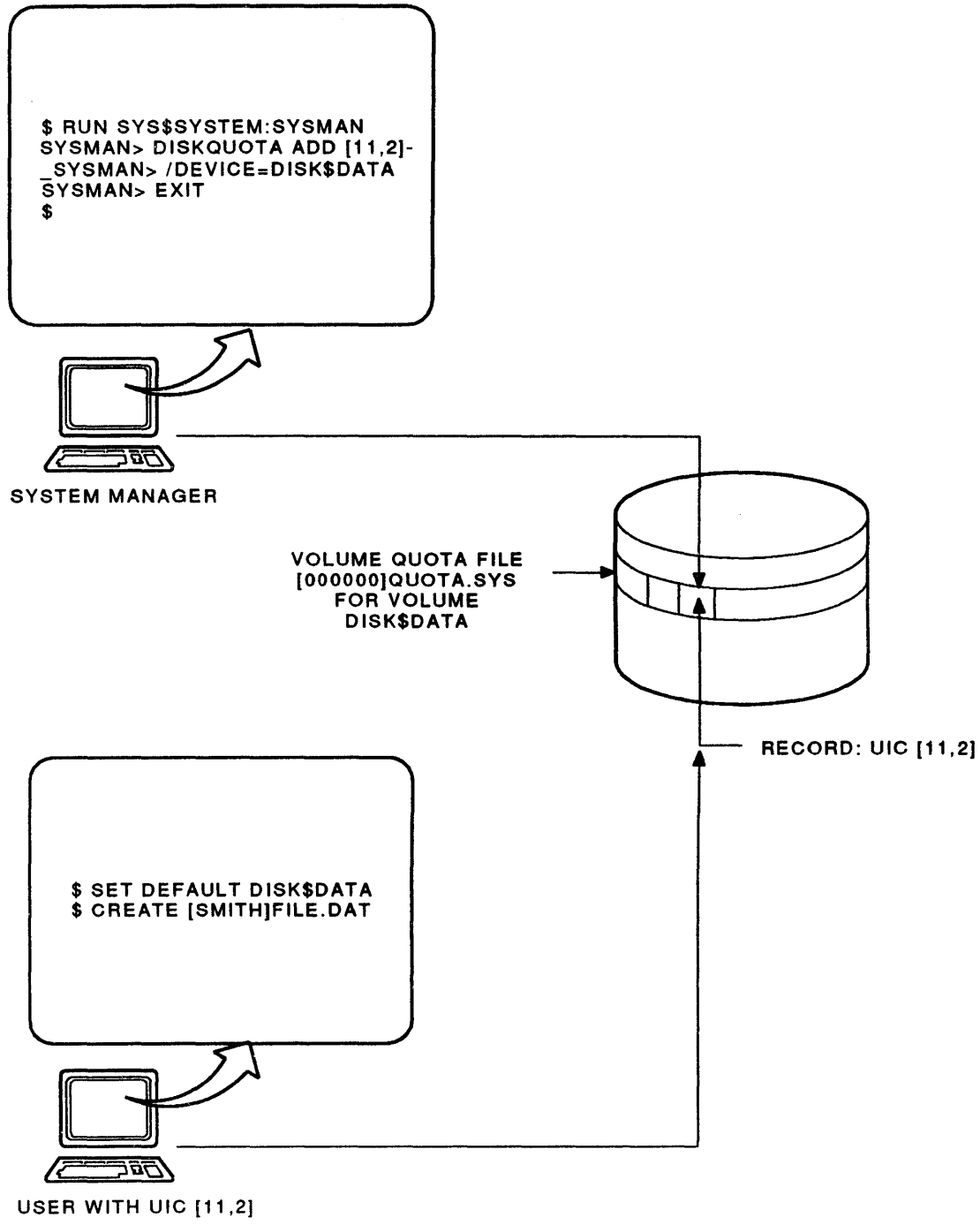


Table 2–23: Displaying the Contents of a Volume Quota File

Operation	SYSMAN Command Format
Displaying the entry of a particular user	`DISKQUOTA SHOW [uic]`
Displaying the entries of all users with UICs in a particular group	`DISKQUOTA SHOW [group-number,*]`
Displaying the entries for all users	`DISKQUOTA SHOW [*,*]`
Displaying DISKQUOTA commands	`HELP DISKQUOTA`

NOTE

The disk volume usage recorded by VMS and displayed by SYSMAN includes some overhead. Therefore, the disk usage displayed by the SYSMAN command DISKQUOTA SHOW and the DCL command SHOW QUOTA is usually different than the disk usage displayed by the DCL command DIRECTORY/SIZE=ALLOCATED.

Example 2–8: List of Volume Quota File Records

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> DISKQUOTA SHOW [*,*] /DEVICE=DISK$USER

   UIC                 Usage         Permanent Quota     Overdraft Limit
[0,0]                  0             690                 200
[SYSTEM]               12047         13000               200
[VMS,BEYER]            11685         15000               200
[11,15]                56            56                  200
[VMS,CLARK]            16233         20000               200
[VMS,DORSEY]           13510         20000               200
[VMS,HARKINS]          18221         20000               200
[VMS,HUNT]             21060         30000               200
[11,340]               22905         30000               200
[VMS,DISALVO]          9021          18000               200
[VMS,TARGONSKI]        2425          4000                200
[12,1]                 4             690                 200
[BEYER2]               142           144                 200
[GROUP21,ALBERT]       14137         20000               200
[21,10]                10            690                 200
[21,20]                2             690                 200
[GROUP21,EBERT]        5962          12000               200
[GROUP21,GALVIN]       3295          5000                200
[GROUP21,TATAR]        32            2000                200
[31,5]                 2             2                   200
[GROUP31,HARBO]        6117          10000               200
[GROUP31,CONNOR]       3261          8000                200
[PAPISON]              666           690                 200
[CHERPAS]              19            690                 200
[GROUP101,ALCOCK]      29806         30000               200
[GROUP101,LUCAS]       27257         30000               200
[GROUP101,MASORS]      125           690                 200
[GROUP101,WILSON]      20968         25000               200
[123,321]              20            690                 200
[DATA_COMM,DELLA]      12931         20000               200
[DATA_COMM,LENTZ]      6341          20000               200
[200,3]                2             690                 200
[200,200]              60            690                 200
[DECNET]               78            690                 200
[J65,DOE]              4             100                 100

SYSMAN> EXIT
$
```

Table 2–24: Managing Individual Records in the Volume Quota File

Operation†	SYSMAN Command Format
Adds a new entry, specifying values different from the default entry ([0,0])	`DISKQUOTA ADD uic -` `_SYSMAN> [/PERMQUOTA=blks1] [/OVERDRAFT=blks2]`
Modifies an existing entry	`DISKQUOTA MODIFY uic -` `_SYSMAN> [PERMQUOTA=blks1][/OVERDRAFT=blks2]`
Modifies the entry for [0,0], used to supply default values for Permquota and Overdraft ([0,0] should never own any files.)	`DISKQUOTA MODIFY [0,0] -` `_SYSMAN> [/PERMQUOTA=blks1] [/OVERDRAFT=blks2]`
Modifies all entries for UICs in a particular group	`DISKQUOTA MODIFY [group-number,*] -` `_SYSMAN> [/PERMQUOTA=blks1] [/OVERDRAFT=blks2]`
Modifies all entries, including the default entry, [0,0]	`DISKQUOTA MODIFY [*,*] -` `_SYSMAN> [/PERMQUOTA=blks1] [/OVERDRAFT=blks2]`
Removes an existing entry	`DISKQUOTA REMOVE uic`

†The SYSMAN utility performs all operations on the current QUOTA.SYS file. The current file is the one on your current default device if you did not specify one with the **/DEVICE** qualifier. Be sure to specify the proper volume for your command; otherwise, the most recently used **/DEVICE** qualifier sets the current file specification.

Table 2–25: Establishing Quotas on an Existing Volume Called DISK$USER

Steps	Commands	Comments
1	Notifies users that DISK$USER will be unavailable	There are a number of ways to notify users of the volume. See the section **Communicating with User Processes** in this module.
2	`$ RUN SYS$SYSTEM:SYSMAN`	Invoke the SYSMAN utility. Make sure your current process has the SYSPRV privilege to use the SYSMAN utility.
3	`SYSMAN> DISKQUOTA -` `_SYSMAN> CREATE /DEVICE=DISK$USER`	Creates a quota file on the DISK$USER volume (DISK$USER:[000000]QUOTA.SYS) and automatically enables quotas on that volume.

4	`SYSMAN> DISKQUOTA MODIFY [0,0] -`
	`_SYSMAN> /PERMQUOTA=10000 -`
	`_SYSMAN> /OVERDRAFT=1000`
		Sets the default entry values for the quota file on DISK$USER. Use appropriate values for **/PERMQUOTA** and **/OVERDRAFT** to reflect your management policy on the volume. Note that the **/DEVICE=DISK$USER** qualifier need not be specified, as the qualifier was properly specifed in Step 3.

5	`SYSMAN> DISKQUOTA REBUILD`	Updates the newly created quota file to add existing UICs that own files on the DISK$USER volume. Note again that **/DEVICE=DISK$USER** qualifier need not be specified in this case.
6	`SYSMAN> EXIT`	Exit from the SYSMAN utility.
7	Notifies users that DISK$USER is available for use.	See the section **Communicating with User Processes** for possible techniques of user notification.

Disabling and Enabling Quotas on a Volume

- To disable quotas, use the SYSMAN command **DISKQUOTA DISABLE**
- Use the SYSMAN command **DISKQUOTA REBUILD** to properly assess user space on a volume if:
  - — The volume previously had a quota file created
  - — Quotas were once enabled on the volume
  - — Quotas were later disabled
- An automatic **DISKQUOTA REBUILD** performed when a volume is mounted after being improperly dismounted
  - — Typical situation after a system failure (crash)

Managing Disk Space Using DCL Commands

Table 2–26: Controlling Files with DCL Commands

Operation	Command Format and Examples	Comments

Changing the ownership of a file
 	
Remember that VMS records the use of disk space on a UIC basis, so changing the owner UIC of a file changes which disk quota entry the space is recorded under.
	`$ SET FILE/OWNER_UIC=uic -`
	`_$ file-spec`
	
	`$ SET FILE/OWNER_UIC=[320,63] -`
	`_$ INVENTORY.DAT`
		This operation requires SYSPRV privilege or a system UIC. One reason for using this would be if you forget the **/OWNER** qualifier when copying a file to a user's directory.

Establishing automatic version deletion for a particular file	`$ SET FILE/VERSION_LIMIT=n -`
	`_$ file-spec`
	`$ SET FILE/VERSION_LIMIT=2 -`
	`_$ [MARTIN...]*.*`
		VMS automatically deletes the file with the lowest version number when you create a new version to keep the total number of versions equal to the version limit for the file.

Adding an ACL to a file or modifying on existing ACL on a file	`$ SET ACL/OBJECT_TYPE=FILE -`
	`_$ file-spec`
	`$ SET ACL/OBJECT_TYPE=FILE -`
	`_$ [ADMIN]PAYROLL.DAT`
		The keyword FILE specifies the file whose ACL is being edited.

Deleting unwanted versions of files	`$ PURGE filespec /KEEP=n` `$ PURGE TEMP.DAT /KEEP=1`	The **/KEEP** qualifier is used to specify how many versions to retain. If no parameter is specified, the qualifier defaults to 1.

Table 2–27: Controlling Directories and Volumes with DCL Commands

Operation	Command Format and Examples	Comments

Changing the ownership of a directory
 	
Changing the UIC changes which quota file entry the space is recorded under.
	`$ SET DIRECTORY/OWNER_UIC=uic -`
	`_$ directory-spec`
	`$ SET DIRECTORY/OWNER_UIC=[ENG] -`
	`_$ DISK$DESIGNS:[PRODUCT_X]`
		This operation requires SYSPRV privilege or a system UIC. One reason for using this would be if you forget the /**OWNER** qualifier when creating the directory for a user.

Establishing automatic version deletion for all files in a directory	`$ SET DIRECTORY/VERSION_LIMIT=n -`
	`_$ directory-spec`
	`$ SET DIRECTORY/VERSION_LIMIT=3 -`
	`_$ DISK$MFG:[INVENTORY]`
		This sets a version limit on all files subsequently created in the directory. It does not affect previously created files. You can also use the **SET FILE** command to set a different limit for a particular file.

Adding an ACL to a directory file or modifying an existing ACL on a directory file	`$ SET DIRECTORY/ACL -`
	`_$ directory-spec`
	`$ SET DIRECTORY/ACL WORK1:[SMITH]`
		Invokes the ACL editor, discussed later in this course in MODULE 8.

Disabling special protection on a directory file	`$ SET FILE/NODIRECTORY file-spec`
	`$ SET FILE/NODIRECTORY -`
	`_$ [USERS]SMITH.DIR`
		This command enables you to delete a corrupted directory file. It is not intended for use on normal (valid) directory files.

Changing the ownership of a volume
 	
Volumes have protection masks that are similar to file protection masks. A user who does not pass the volume protection mask cannot access any directories or files on the volume. If you change the owner UIC of a volume, you affect who will be able to access space on that volume.
	`$ SET VOLUME/OWNER_UIC=uic -`
	`_$ device-spec`
	`$ SET VOLUME/OWNER_UIC=[ADMIN] -`
	`_$ DISK$DATA`
		This operation requires the VOLPRO privilege.

COMMUNICATING WITH USER PROCESSES

- System managers must be fluent in user communication methods
- Communication methods on VMS:
  - — Common directory for collecting and distributing information
  - — VMS utilities and commands

Table 2–28: Communication Methods

Communication	Method
Operator/user direct dialogue	PHONE utility
System–wide distribution of messages and information	MAIL utility
User requests to the operator	REQUEST command MOUNT command
Operator responses to user requests, or short messages to users	REPLY command

Handling Requests for Operator Assistance

- The *operator communication process:* OPCOM
  - — Coordinates all requests by users for operator assistance
  - — Coordinates all operator responses to users
- Requests for operator assistance come from:
  - — **MOUNT** or **REQUEST/REPLY** commands
  - — I/O events
- Requests are sent to *operator terminals:*
  - — Based on the *category* of the request
  - — Operator terminals must be enabled for requests
- Use the **REPLY/ENABLE** command to enable operator terminals

```
$ REPLY/ENABLE=(PRINTER,TAPES) TXG6
```

- Use the **REPLY/DISABLE** command to disable operator terminals

```
$ REPLY/DISABLE=PRINTER TXG6
```

- Use the **REPLY/STATUS** command to determine request categories enabled on your terminal

Table 2–29: Operator Categories Enabled/Disabled with the REPLY Command

Category Keyword	Description
CARDS	Displays messages sent to the card readers.
CENTRAL	Displays messages sent to the central system operator.
CLUSTER	Displays messages from the connection manager pertaining to cluster state changes.
DEVICES	Displays messages pertaining to mounting disks.
DISKS	Displays messages pertaining to mounting and dismounting disk volumes.
NETWORK	Displays messages pertaining to networks; the keyword CENTRAL must also be specified to inhibit network messages when disabling network messages with qualifier /**DISABLE=NETWORK**.
OPER1 through OPER12	Displays messages sent to operators identified as OPER1 through OPER12.
PRINTER	Displays messages pertaining to print requests.
SECURITY	Allows messages pertaining to security events. Requires SECURITY privilege.
TAPES	Allows messages pertaining to mounting and dismounting tape volumes.

Table 2–30: Events Requiring Operator Assistance

Event Causing Request	Category of Operator Notified	Operator Action	Comments
MOUNT commands	DISKS or TAPES	Mount a disk or tape volume.	By default, all **MOUNT** commands request operator assistance unless the appropriate volume is already loaded on the drive.
REQUEST/REPLY commands	All operators or only the group(s) of operators you specify. Group names include DISKS, TAPES and OPER1.	Perform the operation (or not), then respond to the user.	The user is waiting for a reply from the operator, and can do no other work until a reply is received.
I/O event: any command that reads a file on a tape	TAPES	Mount a new tape reel for the file system when it is reading a multireel volume set. Mount a new tape reel for the file system when it is writing a multireel volume set.	The operator receives a message from the MOUNT utility to mount another volume. The message looks similar to the message received from the **MOUNT** command when a user is attempting to mount a tape that is not loaded.

Example 2–9: Using REQUEST/REPLY

```
$
$ REQUEST/REPLY "Please respond to this message"
%OPCOM-S-OPRNOTIF, operator has been notified, waiting...  20:18:40.03
%OPCOM-S-OPREPLY,
Hold on just a minute, please.
20:20:46.88, request 13578 is pending by operator _SUPER$TTD6:

%OPCOM-S-OPRNOTIF, operator has been notified, waiting...  20:20:46.99
*CANCEL*
%OPCOM-I-RQST_PROMPT, REQUEST - Enter message or cancel request with control/Z.
REQUEST - Message? *EXIT*
%OPCOM-S-OPREPLY,
%%%%%%%%%%  OPCOM  24-JUL-1984 20:21:03.89  %%%%%%%%%%
Request 13578 was canceled
$
$
```

Example 2–10: Operator Aborts a Request

```
$ REQUEST/REPLY "Please mount the tape labeled MYBACKUP"
%OPCOM-S-OPENOTIF, operator has been notified, waiting...   20:25:58.64
%OPCOM-S-OPREPLY,
There is no such tape
20:26:20.81, request 14602 was aborted by operator _SUPER$TTD6:
$
$
```

Example 2–11: Operator Receives Message from REQUEST/REPLY Command

```
%%%%%%%%%%  OPCOM  24-JUL-1984  19:00:15.00  %%%%%%%%%%
Request 5377, from user JONES
_TTA0:, Please connect remote sensor 3A
$
```

Example 2–12: Operator Receives Message from MOUNT Program

```
%%%%%%%%%%  OPCOM  24-JUL-1984  19:00:57.60  %%%%%%%%%%
Request 6401, from user SMITH
Please mount volume DATA15 in device_MUA0:
$
```

Example 2–13: Operator Receives Message from REQUEST Command, No Reply Required

```
%%%%%%%%%%  OPCOM  24-JUL-1984  17:01:15.00  %%%%%%%%%%
Message from user GREEN
_TTB2:. Please check the line printer
$
```

Example 2–14: Operator Receives Two Messages from AUTHORIZE Utility

```
%%%%%%%%%%  OPCOM 24-JUL-1984  19:02:16.92  %%%%%%%%%%
Message from user SYSTEM
PID=2020008D SYSTEM added SYSUAF record BROWN on 24-JUL-1984  19:02:16.91

%%%%%%%%%%  OPCOM  24-JUL-1984  19:02:35.48  %%%%%%%%%%
Message from user SYSTEM
PID=2020008D SYSTEM modified SYSUAF record SMITH on 24-JUL-1984 19:02:35.47

$
```

Table 2–31: Providing Operator Assistance

Circumstance	Command Format	Comments
User–requested function	`$ REPLY/TO=request-id -` `_$ "message-text"`	Users request various functions. Use the **REPLY** command with an appropriate message to respond.
User enters **MOUNT**command without loading a disk or tape volume in drive	No command required. Just load the volume requested on the drive.	The MOUNT utility sends a message to the operator terminal(s). The message includes the name of the volume requested.
User enters **MOUNT** command without the volume first being loaded. You attempt to load the volume on the drive, but fail. You decide to load the volume on an alternate drive (MUA2).	`$ REPLY/TO=request-id -` `_$ "Reissue mount using MUA2"`	Tell the user where you are loading the volume. The user must enter the **MOUNT** command again.
User needs to read information that spans more than one volume. You must load the next volume when the system requests it.	`$ REPLY/TO=request-id -` `_$ "volume-label"`	Respond to the system message after you have loaded the volume. Specify the volume's label in the message.

User needs to write more information than one volume can contain. You must load another volume when the system requests it.	`$ REPLY -`
	`_$ /INITIALIZE_TAPE=request-id -`
	`_$ "tape-label"`
		Use the **/INITIALIZE_TAPE** qualifier if the tape has been initialized before. VMS checks the tape's protection code. If you are not allowed access according to the code, you need the VOLPRO privilege to gain access.

	`$ REPLY -`
	`_$ /BLANK_TAPE=request-id -`
	`_$ "tape-label"`
		Use the **/BLANK_TAPE** qualifier for new tapes. You need VOLPRO privilege because it bypasses volume protection checking.

Cannot perform user–requested function; must abort request	`$ REPLY/ABORT=request-id -` `_$ "message-text"`	Typically used when you cannot locate or use a resource requested by the user, (such as a volume, a drive, paper, or ribbon). You abort the request to cancel the repeated messages sent to the operator terminal. Then find the resource or use the **REPLY** command or the MAIL utility to explain the situation more completely.
May be able to perform requested function; want to notify interactive user, but leave request pending	`$ REPLY/PENDING=request-id -` `_$ "message-text"`	Since the request is pending, the user does not regain control of his or her own terminal. The user regains control (returns to the DCL prompt) by cancelling the request or by waiting until the operator completes it.

Example 2–15: Request–Reply Interaction Between User JONES and User TAPES

```
Tapes:  $ REPLY/ENABLE=(TAPES,OPER1)
        %%%%%%%%%%  OPCOM  24-JUL-1984  19:28:18.57  %%%%%%%%%%
        Operator _SUPER$TTA1: has been enabled, username TAPEOPR

        $
        %%%%%%%%%%  OPCOM  24-JUL-1984  19:28:18.73  %%%%%%%%%%
        Operator status for operator _SUPER$TTA1:
        TAPES, OPER1

        $

Jones:  $ REQUEST/TO=TAPES/REPLY "Please mount tape 362F"
        %OPCOM-S-OPRNOTIF, operator has been notified, waiting... 19:31:18.65

Tapes:  $
        %%%%%%%%%%  OPCOM  24-JUL-1984  19:31:18.67  %%%%%%%%%%
        Request 9482, from user JONES on SUPER
        _SUPER$TTA2:, Please mount tape 362F

        $
        %%%%%%%%%%  OPCOM  24-JUL-1984  19:31:18.67  %%%%%%%%%%
        Request 9482, from user JONES on SUPER
        _SUPER$TTA2:, Please mount tape 362F

        $ REPLY/PENDING=9482 "You are second in line"

Jones:  %OPCOM-S-OPREPLY,
        You are second in line
        19:31:48.82, request 9482 is pending by operator_SUPER$TTA1:

        %OPCOM-S-OPRNOTIF, operator has been notified, waiting... 19:31:49.05

Tapes:  $ REPLY/TO=9482 "Tape 362F is now mounted"
        Tape 362F is now mounted
        19:39:18.48, request 9482 was completed by operator _SUPER$TTA1:

        $

Jones:  %OPCOM-S-OPREPLY,
        Tape 362F is now mounted
        19:39:18.48, rquest 9482 was completed by operator _SUPER$TTA1:

        $ REQUEST/TO=TAPES "Thank you"

Tapes:  $
        %%%%%%%%%%  OPCOM  24-JUL-1984  19:39:43.16  %%%%%%%%%%
        Message from user JONES on SUPER
        _SUPER$TTA2:. Thank you

        $ REPLY/DISABLE=(TAPES)
        %%%%%%%%%%  OPCOM  24-JUL-1984  19:40:02.16  %%%%%%%%%%
        Operator status for operator _SUPER$TTA1:
        OPER1

        $
```

Table 2–32: Sending Messages to Users

Operation†	Command Format and Examples
Sending a message to all terminals on the system	`$ REPLY/ALL "message-text"` `$ REPLY/ALL/BELL "DISK$USER will be unavailable at noon."`
Sending a message to all terminals where users are logged on	`$ REPLY/USERNAME "message-text"` `$ REPLY/USERNAME "Policy Meeting in 10 minutes"`
Sending a message to specified users	`$ REPLY/USERNAME=(name [,...]) "message-text"` `$ REPLY/USERNAME=(JKMARTIN,BECKER) "DISK$DATA is available"`
Sending a message to a specified terminal or terminals	`$ REPLY/TERMINAL=(term [,...]) "message-text"` `$ REPLY/TERMINAL=(TXQ2,TXQ3) "Terminal repairperson due at 3pm!"`
Sending a message about system shutdown‡	`$ REPLY/SHUTDOWN "message-text"` `$ REPLY/SHUTDOWN "System will be down for PM at 2."`
Sending an urgent message‡	`$ REPLY/URGENT "message-text"` `$ REPLY/URGENT "Everyone off the system NOW!!"`

† To ring or buzz the terminal receiving the message, use the additional **/BELL** qualifier. More bells ring for URGENT and SHUTDOWN messages than for other types of messages.

‡ URGENT and SHUTDOWN messages are NOT sent to user terminals that have the command **SET TERMINAL/NOBROADCAST.**

REPLY Command

Table 2–33: Qualifiers to the REPLY Command

Function	Qualifier
Responding to Requests	
Lists the requests that have not been sent a final message	`/STATUS`
Sends a final response to a request	`/TO=request-id`
Sends a response to a request, but not the final response	`/PENDING=request-id`
Cancels the request	`/ABORT=request-id`
Sends the label of an initialized tape as the final response	`/INITIALIZE_TAPE=request-id`
Sends the label of a blank tape as the final response	`/BLANK_TAPE=request-id`
Sending Messages and Information	
Sends a message to a specific terminal	`/TERMINAL=(term [,...])`
Sends a message to all users presently logged in to the system, or to specific users	`/USERNAME` `/USERNAME=(name [,...])`
Sends a message to all terminals whether a user is logged in or not	`/ALL`
Sends an urgent message to all users	`/URGENT`
Sends a shutdown message to all users	`/SHUTDOWN`
Causes the terminal receiving the message to buzz or ring (add this to any of the above qualifiers)	`/BELL`
Requests that the originating terminal be notified when the message is successfully received	`/NOTIFY`
Controlling Operator Terminals and the Log File	
Sets up a nonconsole terminal as an operator's terminal	`/ENABLE`
Stops a terminal from being an operator's terminal	`/DISABLE`
Closes the current OPERATOR.LOG file and opens a new file with a higher version number	`/LOG`
Designates this terminal to be an operator's terminal only for the current interactive session.	`/TEMPORARY`
Lists the categories enabled for this terminal	`/STATUS`

Operator's Log File

- SYS$MANAGER:OPERATOR.LOG
- Entries made by OPCOM process
- Logs operator requests and replies
- Controlled by the system manager

Table 2–34: Controlling the Operator's Log

Operation	Command Format	Comments
Closing the operator's log and opening a new one	`$ REPLY/LOG`	Requires OPER privilege.
Closing the operator's log but not opening a new one	`$ REPLY/NOLOG`	Requires OPER privilege.
Printing the operator's log file	`$ SET DEFAULT SYS$MANAGER` `$ PRINT OPERATOR.LOG;n`	In this command format, "`;n`" indicates the version number. You can print (or type) any version except the current one, which is locked by OPCOM.

SUMMARY

- The system manager is responsible for the management of key system databases and command procedures
  - — User authorization file (SYS$SYSTEM:SYSUAF.DAT)
  - — Quota files on disk volumes ([000000]QUOTA.SYS)
  - — Login files (for example, SYS$MANAGER:SYLOGIN.COM)
- Many process characteristics are defined by values in the corresponding UAF record
- Add a user account to the system by creating
  - — UAF record, specifying:
    - Identification and environment information
    - Access and security constraints
    - Quotas and resource limits
    - Privileges
  - — Default disk device and directory
  - — One or more disk quota entries
- To remove a user from the system
  - — Remove the UAF record
  - — Remove remaining disk files
  - — Remove quota entries (if unique to user's UIC)
- Establish and maintain disk quotas on volumes using SYSMAN utility
  - — **DISKQUOTA** subcommands
- Use DCL commands to regulate resources and processes running on the system

SET PROCESS	Set process characteristics
STOP	Stop a process
PURGE	Delete old versions of files
SET FILE	Set file characteristics
SET DIRECTORY	Set directory characteristics
SET VOLUME	Set volume characteristics

- Respond to user requests and send messages using
  - — **REPLY** command
  - — MAIL utility
  - — PHONE utility
- Define operator terminals using **REPLY/ENABLE=(class,...)**
  - — Set reception of certain operator–related messages by specifying appropriate class keywords with the /**ENABLE** qualifier
- Manage the operator log file using
  - — **REPLY/[NO]LOG**
  - — **PURGE**

MODULE 3
MANAGING QUEUES

INTRODUCTION

When you issue the **PRINT** command to print a file, all the system printers may already be in use. Because this condition occurs often, VMS maintains a list of all print requests in the order in which they occur. This ordered list is called a *print queue*, and the requests are called *print jobs*. The **PRINT** command places your job in the queue. When the system is ready to process your job, it passes it to the printer associated with the queue or the first available printer, which prints the file. Similarly, when you issue the **SUBMIT** command to execute a command procedure in a batch process, system resources may not be sufficient to support the immediate creation of a *batch process* for you. Therefore, VMS maintains a list of all batch requests in the order in which they occur. This ordered list is called a *batch queue*, and the requests are called *batch jobs*. When the system is ready to process your batch job, it creates a batch process and executes your command procedure in the context of that batch process.

System managers customize batch and print queues on their systems. They attempt to maximize the performance of the system by matching the system queue structure to the workload and resources of their system. Usually, the system manager includes queue customization commands in startup procedures, so queues are created automatically when he or she starts the system. The module **Customizing the System** contains a more detailed discussion on startup procedures.

As the queues are used, various problems may require the system manager's intervention. For example, paper can get jammed in a line printer, printers can run out of paper, or a batch job may need to be aborted. In these and other cases, the system manager must enter queue management commands to control the jobs in a queue while fixing the problem.

OBJECTIVES

To share limited printing and CPU resources among users and processes, a system manager must be able to:

- Describe how the VMS operating system handles print and batch jobs
- Assess user requirements for batch and print facilities
- Perform the following queue management tasks:
  - — Select proper queue attributes and characteristics to match user requirements
  - — Create print and batch queues
  - — Restrict access to queues
  - — Modify the attributes and characteristics of queues
  - — Control queues
  - — Handle queue problems

RESOURCES

1. *VMS System Manager's Manual*
2. *Guide to Using VMS Command Procedures*
3. *VMS DCL Dictionary*
4. *Guide to VMS System Security*
5. *VMS Access Control List Editor Manual*
6. *Guide to Maintaining a VMS System*

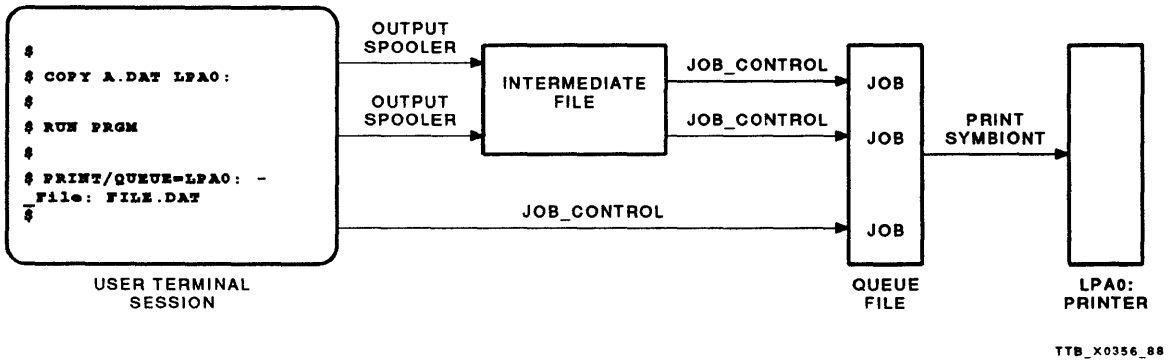
OVERVIEW OF QUEUE FACILITIES AND OPERATIONS

- The queue manager process (JOB_CONTROL)
- The system queue file (SYS$SYSTEM:JBCSYSQUE.DAT)
- Types of queues

Queue Manager and the System Queue File

- Queue information stored in SYS$SYSTEM:JBCSYSQUE.DAT
- **START/QUEUE/MANAGER**
  - — Creates SYS$SYSTEM:JBCSYSQUE.DAT (if it does not already exist)
  - — Starts the queue manager function of the JOB_CONTROL process
  - — Should be included in the startup command procedure (SYSTARTUP_V5.COM)
  - — Should use the qualifier **/RESTART** if the command must be used interactively to restart the queue manager

Figure 3–1: JOB_CONTROL Process Handles All Print Jobs



Types of Queues

- Two *classes* of queues:
  - — Execution

    Accepts either batch or print jobs for processing, depending on how the queue was initialized (created)

  - — Generic

    Holds jobs until they are transferred to an assigned execution queue
- Queue classes are further categorized into *types* of queues, depending on:
  - — The kind of job the queue accepts
  - — The type of device to which output is directed
- Three *types* of queues: execution, generic and logical

Table 3–1: Types of Queues

Queue Type	Action
Execution	Executes the job.
Generic	Moves jobs to specified execution queues when resources for executing the job are available.
Logical	Moves jobs to execution queues when the manager sets up an association between a logical queue and an execution queue. The association does not have to be permanent and it does not always have to be made with the same execution queue.

Execution Queues

- Performs the actual processing of the job
- Two types of execution queues:
  - — Batch

    Can only accept (process) batch jobs
    Executes as a *detached* process

  - — Output

    Accepts (typically) print jobs for processing by an independent process called a *symbiont*
    Three types of output execution queues:

Printer	Directs output to line printers
Terminal	Directs output to terminal printers (printers attached to terminal lines)
Server	Processes files in the queue using a specially created symbiont

    Symbionts for server execution queues are user–written

    Not necessarily used for print output operations
    Not covered in this course

Generic Queues

- Used to hold a job until an associated execution queue becomes available
- The list of associated execution queues is defined when the generic queue is initialized
- When an execution queue becomes available, the job is *requeued* from the generic queue to the execution queue
- Two types of generic queues:

Generic batch queue	Directs jobs only to batch execution queues. These are typically used in VAXclusters to distribute the workload across several systems.
Generic output queue	Directs jobs to any of the three types of output execution queues: printer, terminal, or server.

Logical Queues

- Specially defined output execution queue
- Not really an output execution queue
  - — Has its output redirected to another execution queue
  - — Holds a job until it can be transferred to the execution queue
- Appears as a sort of "holding" queue
- Logical queues and generic queues are somewhat similar:
  - — A logical queue is usually associated with a single output execution queue ("one–to–one" relationship)
  - — Multiple logical queues can be associated with a single execution queue ("many–to–one" relationship)
  - — A generic queue can be associated with one or more execution queues ("one–to–one" or "one–to–many" relationship)
- Both the logical and associated execution queues must be started for jobs to be processed
- Can only be used with output execution queues, not with batch queues

HOW VMS HANDLES PRINT JOBS

- Print queues solve waiting and scheduling problems
- **PRINT** command places a job in the queue
- *Print symbionts* execute print jobs
- JOB_CONTROL sends jobs to print symbionts

Example 3–1: JOB_CONTROL and Print Symbiont Processes

```
$ SHOW SYSTEM
VAX/VMS V5.0  on node BROWNY 19-NOV-1987 19:05:46.42   Uptime    9 02:31:19
  Pid    Process Name    State  Pri      I/O       CPU        Page flts Ph.Mem
20200021 SWAPPER         HIB     16        0    0 00:00:41.04         0      0
20200182 JK Martin       CUR      5      179    0 00:00:06.32       866    201
20200026 ERRFMT          HIB      8     2381    0 00:00:27.53        70     94
20200027 CACHE_SERVER    HIB     16        6    0 00:00:00.20        60     90
20200028 CLUSTER_SERVER  HIB     10        6    0 00:00:00.45       104    233
20200029 OPCOM           LEF      8     1585    0 00:00:25.52     10798     60
2020002A SMISERVER       HIB      9       61    0 00:00:01.96       218    612
1 2020002B JOB_CONTROL   HIB      8    13988    0 00:02:29.72       213    473
2020002C CONFIGURE       HIB     10        7    0 00:00:00.37        92    128
2 2020002D SYMBIONT_0001 HIB      6       85    0 00:00:03.92       670     46
2020002F NETACP          HIB     10     2143    0 01:20:25.01   3899776   1500
20200030 EVL             HIB      6     1003    0 00:01:19.52    189831     49  N
20200032 REMACP          HIB      8      323    0 00:00:01.68        75     45
20200198 BECKER          LEF      4      125    0 00:00:03.56       813    155
```

Notes on Example 3–1

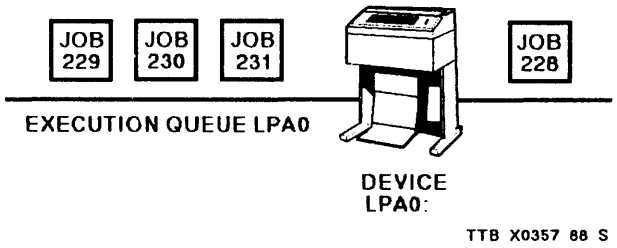
1 The JOB_CONTROL process

2 A print symbiont process

Print Job Scheduling

- Job with highest queue priority executed first
- Smaller jobs executed before larger jobs (within the same priority group)
  - — This is the default action
- Jobs executed in order of submission if they are same size and have same priority
- Priority of jobs in queues limited by two system parameters:
  - — DEFQUEPRI
  - — MAXQUEPRI
- Scheduling can be changed to "first–come–first–served" with the qualifier /**SCHEDULE=NOSIZE**

Figure 3–2: Print Queue



Example 3–2: Scheduling Print Jobs

```
$
$ SHOW QUEUE LPA0/FULL
Printer queue LPA0
    /BASE_PRIORITY=4 /DEFULAT=(FLAG) /FORM=DEFAULT Lowercase
    /OWNER=[SYSTEM] /PROTECTION=(S:E,O:D,G:R,W:W)

  Jobname          Username      Entry  Blocks  Status
  -------          --------      -----  ------  ------
1 ACTION           JONES           228       6  Printing
    Submitted 13-DEC-1987 12:02 /FORM=DEFAULT /PRIO=4
      _DRA1:[JONES]ACTION.COM;1 /COPIES=2

  NOTES            JONES           231      12  Pending
    Submitted 13-DEC-1987 12:15 /FORM=DEFAULT /PRIO=6
      _DRA1:[JONES]NOTES.TXT;1                    2
                                          3
  MEMO             JONES           230       1  Pending
    Submitted 13-DEC-1987 12:08 /FORM=DEFAULT /PRIO=4
      _DRA1:[JONES]MEMO.MEM;1

4 MATH             JONES           229       6  Pending
    Submitted 13-DEC-1987 12:04 /FORM=DEFAULT /PRIO=4
      _DRA1:[JONES]MATH.LIS;1
$
$
```

Notes on Example 3–2

1 Job 228 is currently executing. The JOB_CONTROL process examines the parameters of the pending jobs to determine which job it will print next.

2 Since Job 231 has the highest priority of the pending jobs, it will be printed next.

NOTE

The priority of a job in a queue is limited by two system parameters:

- **DEFQUEPRI – the default queue priority assigned to all print jobs**
- **MAXQUEPRI – the maximum queue priority any user can assign to a job (range is 0–255) (Regardless of the values of these parameters, users with OPER or ALTPRI privilege can submit jobs at any priority using the /PRIORITY qualifier.)**

3 Job 230 is smaller than Job 229, and they have the same priority, so Job 230 will be printed third.

4 Finally, Job 229 will be printed. However, if another job is submitted before Job 229 begins printing, the JOB_CONTROL process will examine the parameters of Job 229 and the new job to determine which job it will print first.

PRINT QUEUE OPERATIONS

Types of Print Queues

- Four types of print queues:

Printer execution queue	Feeds a physical printer
Terminal execution queue	Feeds a physical printer attached to a terminal line
Generic queue	Feeds execution queues enabled with **/ENABLE_GENERIC** attribute
Logical queue	Feeds execution queue associated with **ASSIGN/QUEUE** command

- System moves jobs from generic and logical queues to execution queues
  - — Will not move jobs from generic queues to execution queues initialized with **/NOENABLE_GENERIC** qualifier
  - — Execution queues given **/ENABLE_GENERIC** attribute by default

Creating Print Queues

- To establish a print execution queue (requires OPER privilege):
  - — Set physical attributes of device
  - — Spool device
  - — Initialize and start an execution queue for device
- Queues can be created either:
  - — Interactively
  - — Automatically
- Can create and start queues using one or more commands

Table 3–2: Initializing and Starting Queues

Command	Comments
`$ INITIALIZE/QUEUE [/qualifiers] queue-name`	Creates the queue. If the queue is already running, this command has no effect. If a queue exists but is stopped, you can use this command to modify queue parameters. Jobs listed in the queue and new jobs will execute under the new parameters.
`$ START/QUEUE [/qualifiers] queue-name`	Starts a stopped queue. If the queue is already running, the system displays an error message.
`$ INITIALIZE/QUEUE/START [/qualifiers] queue-name`	Creates and starts a queue. Include this command for each queue in the procedure SYSTARTUP_V5.COM. If the queue is already running, this command has no effect.

Table 3–3: Creating and Using Print Execution Queues

Operation	Creating a Printer Queue	Creating a Terminal Queue	Comments
Determine the device	`$ SHOW DEVICE L`	`$ SHOW DEVICE T`	List the devices and select one.

Set the device attributes	`$ SET PRINTER -`
	`_$ /UPPER LPA0`
		`$ SET TERMINAL -`
		`_$ /PERMANENT -`
		`_$ /NOTYPE_AHEAD -`
		`_$ /SPEED=2400 -`
		`_$ /NOBROADCAST TTA3`
			Set the attributes of the printer or terminal to match its physical attributes or to force the printer to produce specific output. (For example, **/UPPER** causes all jobs to be printed in uppercase.) Terminals must have certain attributes set as shown (speed should be specified to match the terminal speed)

Spool the device	`$ SET DEVICE -`
	`_$ /SPOOLED LPA0`
		`$ SET DEVICE -`
		`_$ /SPOOLED=WORK1 -`
		`_$ TTA3`
			Enables **COPY** commands and write statements for that device; you can specify an intermediate device or use the current default device (SYS$DISK).

Create and start the queue	`$ INITIALIZE/QUEUE -`
	`_$ /START/ON=LPA0 -`
	`_$ SYS$PRINT`
		`$ INITIALIZE/QUEUE -`
		`_$ /TERMINAL -`
		`_$ /START/ON=TTA3 -`
		`_$ SYS$PRINT`
			Assign it a different name than its device name if desired by using the **/ON** qualifier. (By default, the name of the queue matches the name of the printer.)

List the device queues.	`$ SHOW QUEUE/ALL -` `_$ /DEVICE`	`$ SHOW QUEUE/ALL -` `_$ /DEVICE`	Displays all execution queues.
Use the queue	`$ PRINT FILE.DAT`	`$ PRINT FILE.DAT`	Since the **PRINT** command sends files to the SYS$PRINT queue by default, and the name of the print execution queue for the LPA0 printer is SYS$PRINT, the first command prints FILE.DAT on LPA0. The second command is similar, but the SYS$PRINT queue is defined to print on TTA3.

Creating Generic Print Queues

Establish generic print queues when you have more than one printer set up in the same fashion, and want to share the processing among the printers.

Table 3–4: Creating and Using Generic Print Queues

Operation	Command	Comment

Create an execution queue for a printer	`$ SET PRINTER/UPPER LPA0`
	`$ SET DEVICE/SPOOLED LPA0`
	`$ INITIALIZE/QUEUE/START LPA0`
		For example, the printer device LPA0.

Create an execution queue for another printer	`$ SET PRINTER/UPPER LPB0`
	`$ SET DEVICE/SPOOLED LPB0`
	`$ INITIALIZE/QUEUE/START LPB0`
		For example, the printer device LPB0.

Create a generic print queue	`$ INITIALIZE/QUEUE/START -` `_$ /GENERIC SYS$PRINT`	This queue will receive default print jobs.
Use the generic print queue	`$ PRINT FILE.DAT`	SYS$PRINT is the default queue for the **PRINT** command. In this example, SYS$PRINT is a generic print queue. The file will be printed on LPA0 if it is available. If LPA0 is not available, and LPB0 is available, the file will be printed on LPB0.

Creating Logical Print Queues

- Establish logical queues to hold jobs
  - — Requiring special paper or printer attributes
  - — Requiring the presence of an operator or other authorized personnel to print
  - — When you want to control which printer is used

Table 3–5: Creating and Using Logical Queues

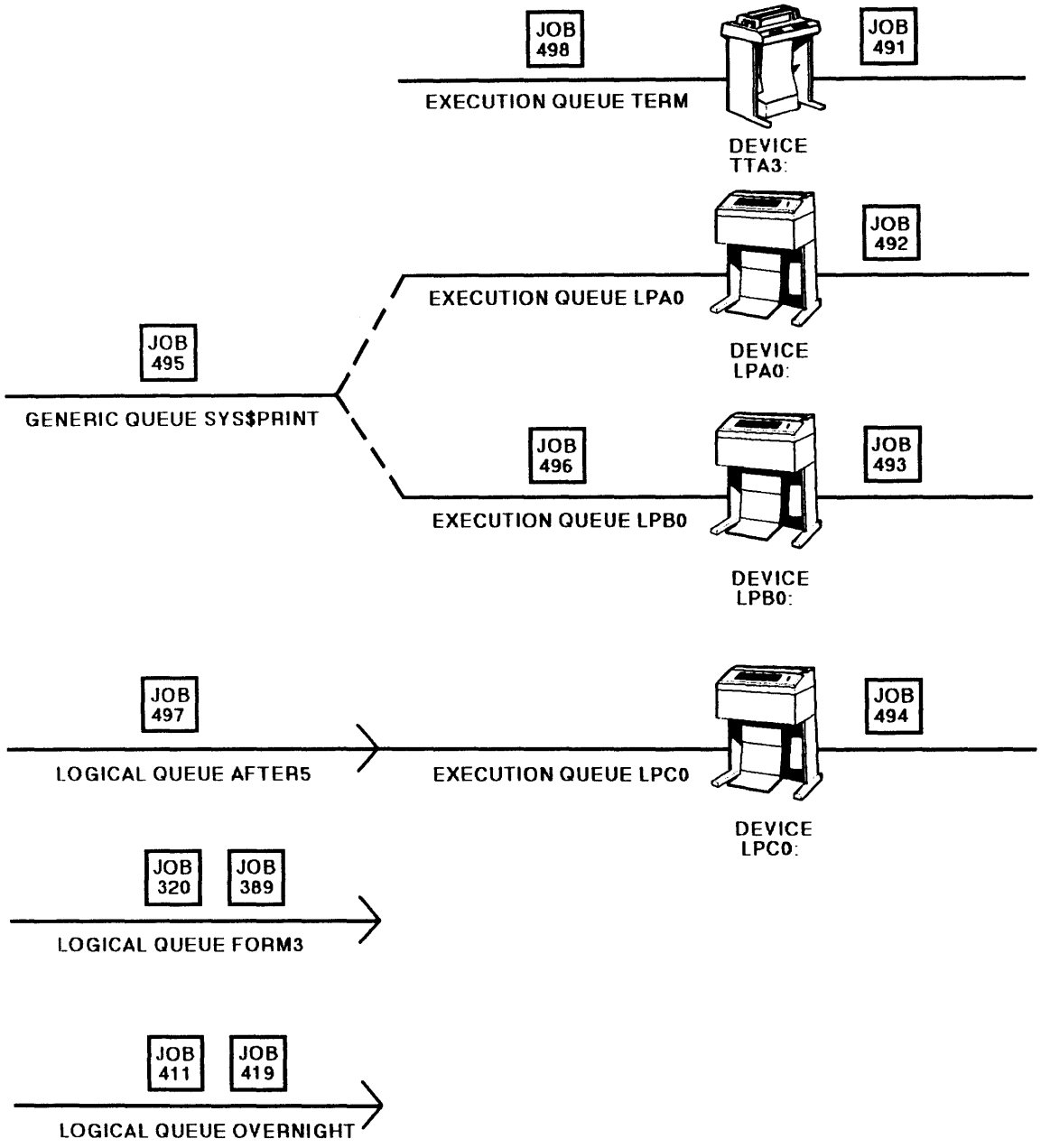
Operation	Command	Comment

Create an execution queue	`$ SET PRINTER/UPPER LPC0`
	`$ SET DEVICE/SPOOLED LPC0`
	`$ INITIALIZE/QUEUE/START LPC0`
		For example, use LPC0 as the execution queue. Set the device to be spooled and to print only in UPPERCASE.

Create a logical queue	`$ INITIALIZE/QUEUE AFTER5`	Note that the /START qualifier is not valid for logical queues.
Use the logical queue	`$ PRINT/QUEUE=AFTER5 FILE.DAT`	Jobs are listed in the queue but are not executed.
Associate the logical queue with the execution queue.†	`$ ASSIGN/QUEUE LPC0 AFTER5`	This tells JOB_CONTROL to "feed" LPC0 jobs from the AFTER5 queue once the AFTER5 queue is started.
Start the logical queue	`$ START/QUEUE AFTER5`	The listed jobs will be executed on the LPC0 printer.
Use the logical queue	`$ PRINT/QUEUE=AFTER5 TEXT.DAT`	The job is listed in the AFTER5 queue, and is moved to the LPC0 queue to be printed as soon as the printer is available. Jobs will continue to be processed in this way until the association is broken (by using the command DEASSIGN/QUEUE).

† A logical queue does not have to be associated with an execution queue. It can be used simply as a holding queue for large jobs, jobs requiring special paper, or other types of jobs. The operator moves these jobs to specific execution queues to be printed when it is appropriate (see Table 3–17).

Figure 3–3: Current, Pending, and Holding Jobs



Example 3–3: Queue Status Display of Current, Pending, and Holding Jobs

```
$
$ SHOW QUEUE/DEVICES/GENERIC
Logical queue AFTER5, assigned to LPC0

  Jobname          Username     Entry  Blocks  Status
  -------          --------     -----  ------  ------
  LATER            JONES          497       1  Pending

Printer queue FORM3, stopped

  Jobname          Username     Entry  Blocks  Status
  -------          --------     -----  ------  ------
  MATH             JONES          389       5  Pending
  TEST             JONES          320       7  Pending

Printer queue LPA0

  Jobname          Username     Entry  Blocks  Status
  -------          --------     -----  ------  ------
  MEMO             JONES          492       1  Printing

Printer queue LPB0

  Jobname          Username     Entry  Blocks  Status
  -------          --------     -----  ------  ------
  ACTION           JONES          493       1  Printing
  TABLES           JONES          496      21  Pending

Printer queue LPC0

  Jobname          Username     Entry  Blocks  Status
  -------          --------     -----  ------  ------
  FORTEST          JONES          494       1  Printing

Printer queue OVERNIGHT, stopped

  Jobname          Username     Entry  Blocks  Status
  -------          --------     -----  ------  ------
  LONG             JONES          419     400  Pending
  BIGJOB           JONES          411     478  Pending

Generic printer queue SYS$PRINT

  Jobname          Username     Entry  Blocks  Status
  -------          --------     -----  ------  ------
  MEMO             JONES          495       1  Pending

Terminal printer queue TERM, on TTA3:

  Jobname          Username     Entry  Blocks  Status
  -------          --------     -----  ------  ------
  PROG             JONES          491      10  Printing
  NOTES            JONES          498       9  Pending
$
$
```

Automatic Queue Creation

- Include queue commands in SYS$MANAGER:SYSTARTUP_V5.COM to create and start queues at system startup
  - — Even better to include a line in SYS$MANAGER:SYSTARTUP_V5.COM to invoke a separate command procedure to start up queues
- System shutdown procedure stops all queues (SYS$SYSTEM:SHUTDOWN.COM)
  - — More detail on queue shutdown found in the module **Starting Up and Shutting Down the System.**

Example 3–4: Startup Commands in SYSTARTUP_V5.COM

```
$ SET NOON
$ !
$ ! Start up the queue manager before issuing any other queue commands
$ !
$ START/QUEUE/MANAGER
$ !
$ ! Define and start up printer queues
$ !
$ SET PRINTER/LOWER  LPA0
$ SET DEVICE/SPOOLED  LPA0
$ INITIALIZE/QUEUE/START/DEFAULT=(BURST,FLAG)  LPA0
$ !
$ SET PRINTER/LOWER  LPB0
$ SET DEVICE/SPOOLED  LPB0
$ INITIALIZE/QUEUE/START  LPB0
$ !
$ SET PRINTER/LOWER  LPC0
$ SET DEVICE/SPOOLED  LPC0
$ INITIALIZE/QUEUE/NOENABLE_GENERIC  LPC0
$ !
$ ! Define and start up a generic print queue
$ !
$ INITIALIZE/QUEUE/GENERIC/START  SYS$PRINT
$ !
$ ! Define a special logical queue and assign it
$ ! to execution queue LPC0, but don't start it yet
$ !
$ INITIALIZE/QUEUE  AFTER5
$ ASSIGN/QUEUE  LPC0  AFTER5
$ !
$ Define other logical queues
$ !
$ INITIALIZE/QUEUE  OVERNIGHT
$ INITIALIZE/QUEUE  FORM3
$
```

Monitoring Print Queues

- Use the **SHOW QUEUE** command to monitor an entire queue:

```
$ SHOW QUEUE [/qualifiers] [queue-name]
$ SHOW QUEUE/SUMMARY/DEVICE=(PRINTER,TERMINAL)
```

- Default action is to display status of all queues and all jobs owned by you
- Queues are displayed in alphabetical order
- Qualifiers provide selection of the type and amount of queue information to be displayed
- Queue status codes indicate current state of the queue (see Table 3–8)

Table 3–6: SHOW QUEUE Qualifiers for Displaying Types of Queues

Qualifier	Description
/BY_JOB_STATUS=status-type	Displays queues that contain jobs of a specified type of status. If no keyword is specified, by default the jobs of all status are displayed.
/BATCH	Displays the status of batch execution queues.
/DEVICE=execution-type	Displays the status of output execution queues. If no keywords are specified, all types of output queues are displayed.
/GENERIC	Displays the status of generic queues.

Table 3–7: SHOW QUEUE Qualifiers for Displaying the Amount of Queue Information

Qualifier	Description
/ALL_JOBS	Displays information about all jobs for the selected queue.
/BRIEF	Displays a brief listing of information about job entries in the queue. The brief listing is the default when no qualifier is specified with the **SHOW QUEUE** command.
/FULL	Displays complete queue and job information, including any ACLs set for the queue.
/SUMMARY	Displays the total number of executing, pending, holding, retained, and time–released jobs.

Table 3–8: Queue Status Codes

Status Code	Description
`aligning`	The queue manager is processing a **START/QUEUE/ALIGN** command.
`device unavailable`	Device to which the print symbiont is assigned is not available.
`pausing`	The queue manager is processing a **STOP/QUEUE** command.
`paused`	A **STOP/QUEUE** command has been executed.
`resuming`	The queue manager is processing a **START/QUEUE** command on a **paused** queue.
`resetting`	The queue manager is processing a **STOP/QUEUE/RESET** command.
`operator service`	A **PRINT/OPERATOR** command has been executed.
`stalled`	Print symbiont processing temporarily halted due to a device related problem.
`stopping`	The queue manager is processing a **STOP/QUEUE** command specified with either a **/NEXT**, **REQUEUE**, or **RESET** qualifier.
`stop pending`	Queue will be **stopped** when current jobs have finished executing.
`stopped`	A **STOP/QUEUE** command specified with either a **/NEXT**, **REQUEUE**, or **RESET** qualifier has been executed.
`starting`	Queue has been started, but the print symbiont process is not yet active.

Monitoring Print Jobs

- Use the **SHOW ENTRY** command to monitor individual jobs:

```
$ SHOW ENTRY [/qualifiers] [entry-number]
$ SHOW ENTRY 228
```

- Most **SHOW QUEUE** qualifiers can be used to select the type and amount of queue information to be monitored
- Queue status codes indicate current state of the job

Table 3–9: Job Status Codes

Status Code	Description
`aborting`	Executing job is terminating.
`executing`	Job is executing from a batch queue.
`holding`	Job is being held until explicitly released.
`holding until`	Job is being held until a specified time.
`pending`	Job is in a wait state, typically waiting to be processed.
`printing`	Job is executing from a printer or terminal execution queue.
`processing`	Job is executing from a server queue.
`retained on completion`	Job remains in the queue upon completion.
`retained on error`	Job remains in the queue upon encountering an error.
`waiting`	Symbiont refuses the job.

Setting Print Queue Attributes

- Some attributes can be modified
  - — Owner (defaults to the user of the process creating the queue)
  - — Base priority
  - — Printer form definition
  - — Protection code
  - — Number of separation pages for print jobs
  - — Maximum and minimum allowed sizes of print jobs
  - — Printer characteristics

Table 3–10: Commands to Modify Queue Attributes at Certain Times

Command	When to Use
`INITIALIZE/QUEUE`	When the queue is being created (does not currently exist)

`SET QUEUE`
`START/QUEUE`
`INITIALIZE/QUEUE`
	After the queue has been created, but is currently **stopped**

`SET QUEUE`	When the queue exists and is currently **running.** Note that not all parameters can be changed while the queue is running.

Example 3–5: Modifying a Running Queue

```
$
$ SHOW QUEUE/FULL LPA0
Printer queue LPA0
     /BASE_PRIORITY=4 /FORM=DEFAULT Lowercase /OWNER=[SYSTEM]
     /PROTECTION=(S:E,O:D,G:R,W:W)
$
$ SET QUEUE/SEPARATE=(BURST,TRAILER) LPA0
$
$ SHOW QUEUE/FULL LPA0
Printer queue LPA0
     /BASE_PRIORITY=4 /FORM=DEFAULT Lowercase /OWNER=[SYSTEM]
     /PROTECTION=(S:E,O:D,G:R,W:W) /SEPARATION=(BURST,TRAILER)
$
$ PRINT/HEADER MEMO.TXT
Job MEMO (queue SYS$PRINT, entry 349) started on SYS$PRINT
$
```

Specifying Separation Pages

- *Separation pages* are used to delineate between individual jobs and individual files within the jobs
  - — Job separation pages
  - — File separation pages
- Defaults can be set for separation pages on a queue (system default is **no separation pages**)
  - — Use **/SEPARATE=option** for job separation pages
  - — Use **/DEFAULT=option** for file separation pages
  - — Users can override defaults set for file separation pages
  - — Users **cannot** override defaults set for job separation pages
- Order of printed pages when all possible file and job separation page defaults are set:
  1. Job burst page (/SEPARATE=BURST)
  2. Job flag page (/SEPARATE=FLAG)
  3. File burst page (/DEFAULT=BURST)
  4. File flag page (/DEFAULT=FLAG)
  5. File contents are printed
  6. File trailer page (/DEFAULT=TRAILER)
  7. System repeats previous four steps until all files in job are printed
  8. Job trailer page (/SEPARATE=TRAILER)
- Separation page attributes can be viewed with **SHOW QUEUE/FULL**

Example 3–6: SHOW QUEUE—Job and File Separation Page Defaults

```
$
$ SHOW QUEUE/FULL LPC0
Printer queue LPC0
     /BASE_PRIORITY=4 /DEFAULT=(FLAG)
     /FORM=DEFAULT Lowercase /OWNER=[SYSTEM] /PROTECTION=(S:E,O:D,G:R,W:W)
     /SEPARATE=(BURST,FLAG,TRAILER)
$
$
```

Table 3–11: Job Separation Page Options for the /SEPARATE Qualifier

Option	Description
[NO]BURST	Specifies a copy of the flag page printed in such a way as to overprint the perforation between the preceding flag page. This makes it possible to determine job breaks in a stack of paper when viewed from the edge side of the paper. Note that if you specify a burst separation page, you do not need to specify a flag page, as it is printed automatically with the burst page.
[NO]FLAG	Specifies a page is printed preceding the job with the name of the user printed in large letters.
[NO]TRAILER	Specifies that a single summary sheet is printed following a job, with the name of the user printed in large letters.

Table 3–12: File Separation Page Options for the /DEFAULT Qualifier

Option	Description
[NO]BURST[=keyword]	Specifies whether file burst pages will be printed. If the keyword is ALL (the default), a burst page is placed before each file in the print job. If the keyword is ONE, a burst page is placed before the first copy of the first file in the job. Note that if you specify a burst separation page, you do not need to specify a flag page, as it is printed automatically with the burst page.
[NO]FLAG[=keyword]	Specifies whether file flag pages will be printed. If the keyword is ALL (the default), a flag page is placed before each file in the print job. If the keyword is ONE, a flag page is placed before the first copy of the first file in the job.
[NO]TRAILER[=keyword]	Specifies whether file trailer pages will be printed. If the keyword is ALL (the default), a trailer page is placed at the end of each file in the print job. If the keyword is ONE, a trailer page is placed after the last copy of the last file in the job.

Figure 3–4: File Separation Burst and Flag Pages

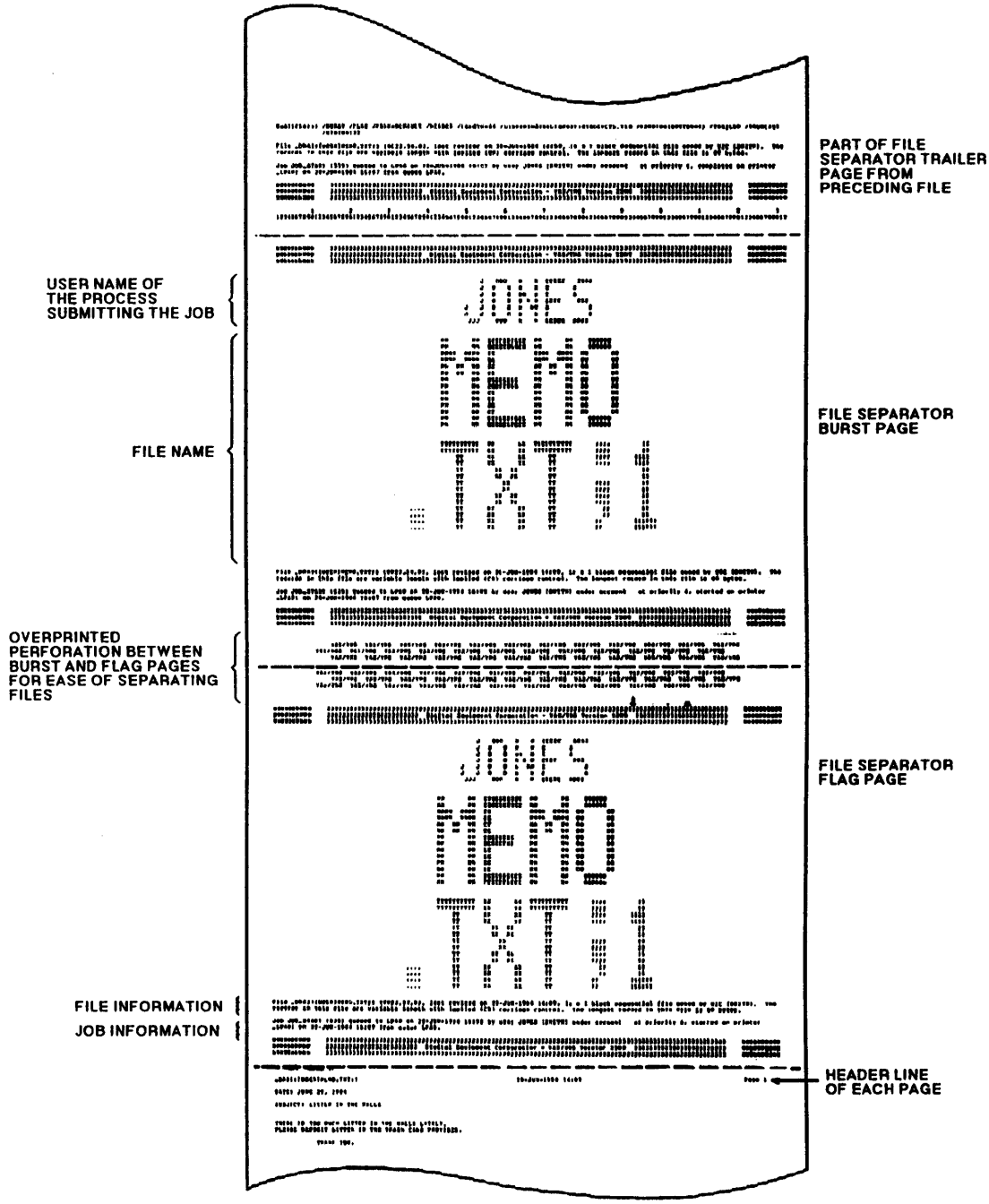


Figure 3–5: File Separation Trailer Page

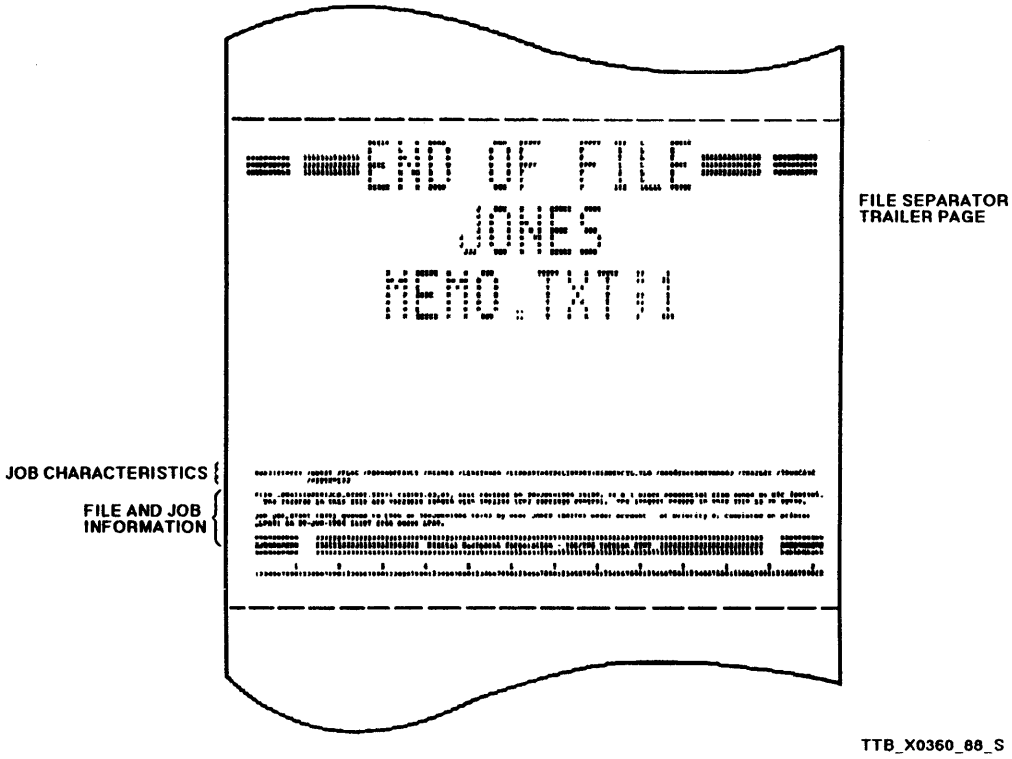


Figure 3–6: Job Separation Burst and Flag Pages

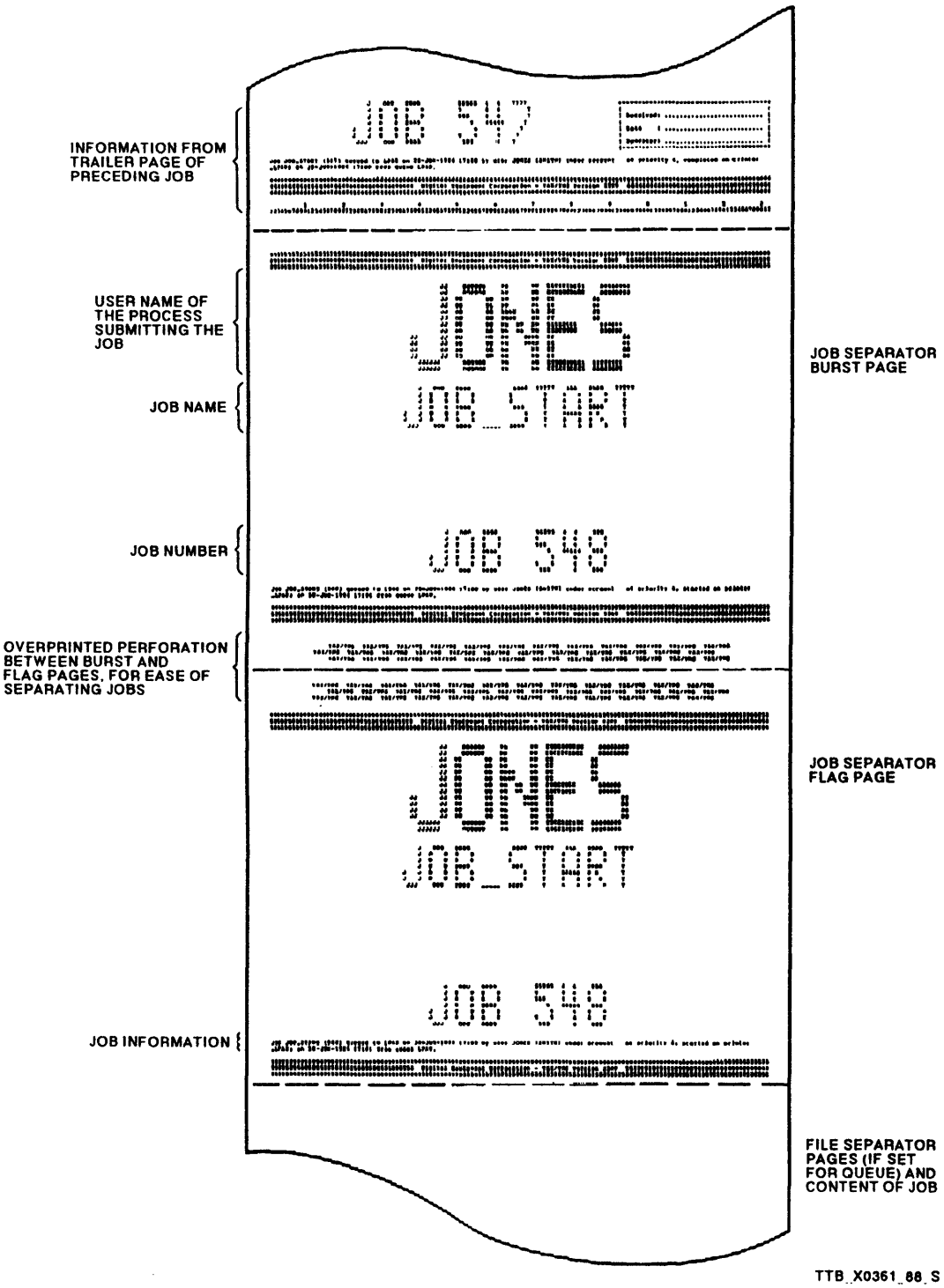
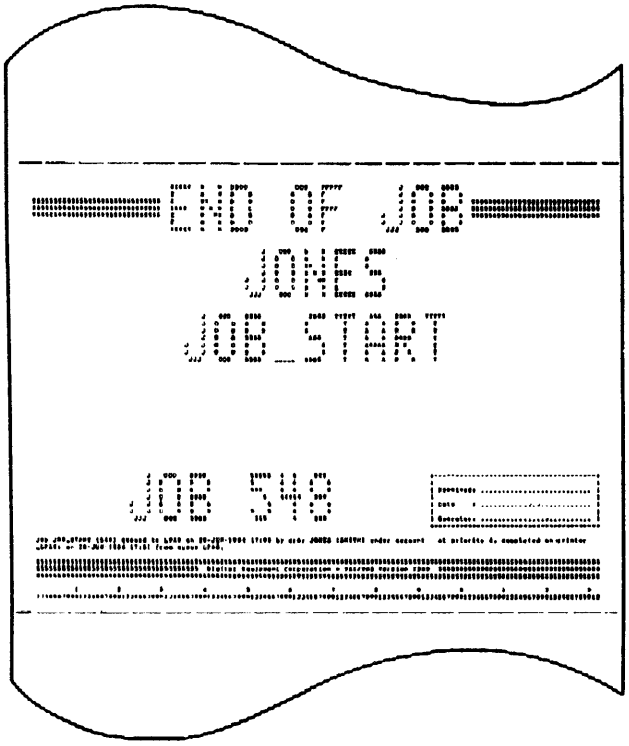


Figure 3–7: Job Separation Trailer Page



Limiting Print Job Sizes

Limiting the size of print jobs can establish different queues for different job sizes.

Table 3–13: Setting Block Limits on Print Queues

Qualifier/Example	Comments
`/BLOCK_LIMIT=(minimum,maximum)`	Size of jobs submitted to queue must be within specified range.
`$ INITIAL/QUEUE/ON=LPA0 -` `_$ /BLOCK=(100,500) MEDIUM`	Jobs smaller than 100 blocks or larger than 500 blocks will not execute from the queue.
`$ INITIALIZE/QUEUE/ON=LPB0 -` `_$ /BLOCK=(500,"") LARGE`	Jobs smaller than 500 blocks will not execute from this queue. It is reserved for jobs larger than 500 blocks.
`$ INITIALIZE/QUEUE/ON=LPC0 -` `_$ /BLOCK=100 SMALL`	Jobs larger than 100 blocks will not execute from this queue. It is reserved for jobs smaller than or equal to 100 blocks. Notice that you do not need to include parentheses when you only specify an upper limit.

Controlling Print Queues

- Can stop or requeue currently executing job without adversely affecting other jobs in queue
- Sometimes you need to stop a queue to fix problems or make modifications

Table 3–14: Aborting and Requeuing Jobs

Possible Reason for Abort/Requeue	Command	Comments
Job contains unprintable characters and is using excessive amounts of paper.	`$ STOP/ABORT queue-name`	Aborts currently executing print job. Next job begins printing.
Current job is not as important as job recently queued. Requeue it to execute later.	`$ STOP/REQUEUE queue-name`	Current print job is aborted and moved to the end of the same queue. Next job begins printing.
Job is too large for this queue or its attributes are wrong. Move it to a more appropriate queue.	`$ STOP/REQUEUE=new-queue-name -` `_$ queue-name`	Current job is aborted and moved to the end of the new queue. Next job begins executing on the original queue.
Current job is not as important as more recently submitted job. Requeue current job to be executed later.	`$ STOP/ENTRY=job-number -` `_$ /REQUEUE queue-name`	Current job is stopped and placed at the end of the queue. The job with the next highest priority will begin executing. (The more important job should have been submitted with sufficiently higher priority to be the next job to execute.)

Table 3–15: Stopping Queues

Possible Reasons for Stopping a Queue	Command	Comments
Temporarily Suspend Queue Execution		
Paper is jammed in printer. Printer needs paper. Printer needs new ribbon.	`$ STOP/QUEUE -` `_$ queue-name`	Stops queue. Suspends execution of current job. No new jobs can be initiated until queue is restarted with the **START/QUEUE** command.
Orderly Shutdown of a Queue		
Queue must be deleted or reassigned, or the jobs in it must be moved to another queue.	`$ STOP/QUEUE/NEXT -` `_$ queue-name`	Stops queue after the currently executing jobs are completed. Queue must be stopped before being deleted or reassigned, or before jobs can be moved.
Immediate Shutdown of a Queue		
You cannot stop or delete the current job. A previous queue control command did not produce the desired effect.	`$ STOP/QUEUE/RESET -` `_$ queue-name`	Abruptly stops queue and stops all currently executing jobs. After this command, you have complete control over the queue and the jobs in it.
Close a Queue		
No more jobs should be entered into a queue.	`$ SET QUEUE/CLOSE -` `_$ queue-name`	Prevents jobs from being entered in the queue through the commands **PRINT** or **SUBMIT**, or as a result of requeue operations. To allow jobs to be entered, use the command **SET QUEUE/OPEN.**
Orderly Shutdown of All Queues		
All queue processing and execution must stop.	`$ STOP/QUEUE/MANAGER`	Performs an orderly shutdown of all queues. Use **START/QUEUE/MANAGER** to restart the queue manager, then restart all queues, either manually or through a command procedure.

- Change the assignment of a logical queue
  - — When the queue's associated execution queue fails and you can process the jobs on another printer
  - — When a different printer is desired for processing existing jobs

Table 3–16: Assigning and Deassigning Logical Queues

Operation	Command	Example

Change the assignment of the queue†	`$ STOP/QUEUE -`
	`_$ logical-queue-name`
	`$ ASSIGN/QUEUE -`
	`_$ execution-queue-name -`
	`_$ logical-queue-name`
		`$ STOP/QUEUE MEMOS`
		`$ ASSIGN/QUEUE LASER2 MEMOS`

Remove the assignment of the queue	`$ STOP/QUEUE logical-queue-name` `$ DEASSIGN/QUEUE logical-queue-name`	`$ STOP/QUEUE SPECIAL_PRINTS` `$ DEASSIGN/QUEUE SPECIAL_PRINTS`

†You can assign a logical queue to only one execution queue at any time, but you can assign it to a different execution queue at any time.

- Keep jobs from being entered in a queue: **$ SET QUEUE/CLOSE**
  - — Use **SET QUEUE/OPEN** to allow jobs to be entered in queue
- Moving jobs from one queue to another
  - — Stop queue with **STOP/QUEUE/NEXT** before moving jobs so that no more jobs will be executed
  - — Requeue current job if queue is an execution queue
  - — Use **SET QUEUE/CLOSE** to keep additional jobs from being entered in the queue, if desired

Table 3–17: Moving Jobs from One Queue to Another

Operation	Command Format	Example: LPA0 needs repair, LPB0 is still operational

Move all jobs from one queue to another.	`$ ASSIGN/MERGE -`
	`_$ destination-queue-name`
	`_$ source-queue-name`
		`$ STOP/QUEUE LPA0`
		`$ STOP/QUEUE/REQUEUE=LPB0 LPA0`
		`$ ASSIGN/MERGE LPB0 LPA0`

- Deleting a queue
  - — Issue **STOP/QUEUE/NEXT** before deleting the queue

Table 3–18: Deleting a Queue

Command Format	Example
`$ DELETE/QUEUE queue-name`	`$ STOP/QUEUE/NEXT LPA0` `$ DELETE/QUEUE LPA0`

- Deleting jobs in queues
  - — Can delete jobs when queue is running or when queue is stopped
  - — Can delete jobs when restarting queue

Table 3–19: Deleting a Job in a Queue

Operation	Command Format	Example
Delete a job in a queue.	`$ DELETE/ENTRY=job-number`	`$ DELETE/ENTRY=715`
Delete the current job when restarting a print execution queue.	`$ START/QUEUE/NEXT queue-name`	`$ START/QUEUE/NEXT LPA0`

Managing Printer Forms and Characteristics

- A print queue has a print *form* associated with it describing aspects of the paper (called *stock*) and the print image area:
  - — Stock name
  - — Left, right, top, and bottom margins
  - — Width of each line (in printable character columns)
  - — Length of the print image area (in lines)
- A print queue can also have one or more *characteristics*
  - — Further specifies the intended use of the queue, or other special attributes

Creating Print Forms and Characteristics

- Use the **DEFINE** command:

```
$ DEFINE/FORM  form-name  form-number  [/qualifiers]
$ DEFINE/FORM CHECK 43 /DESCRIPTION="Company Bank Check" /LENGTH=24 /WIDTH=95

$ DEFINE/CHARACTERISTIC  characteristic-name  characteristic-number
$ DEFINE/CHAR RED 3
```

- Form and characteristic names
  - — Arbitrary strings of between 1 and 31 characters
  - — Characters include both uppercase and lowercase letters, digits, dollar sign, underscore
  - — String must contain at least one nonnumeric character
  - — Name must be unique on the system
- Form and characteristic numbers
  - — Arbitrary, unique numbers
  - — Form numbers can range from 0 to 999
  - — Characteristic numbers can range from 0 to 127
- Use the **SHOW QUEUE/FORM** and **SHOW QUEUE/CHARACTERISTIC** commands to monitor existing form and characteristic definitions

Example 3–7: Displaying Queue Forms and Characteristics

```
$
$ SHOW QUEUE/FORM
Form name                          Number   Description
---------                          ------   -----------
ALLIN1 (stock=DEFAULT)                 99   ALLIN1 Memos
BORDER (stock=DEFAULT)                  4   Border Outline
DEFAULT                                 0   System-defined default
FRAME (stock=DEFAULT)                   3   Overhead Frame
TEMP                                    6   Special temporary
WPSPLUS (stock=DEFAULT)              1101   WPSPLUS Documentation
$
$
$ SHOW QUEUE/CHARACTERISTICS
Characteristic name                Number
-------------------                ------
PEN_010                                 5
PEN_020                                 6
PEN_050                                 7
RED_INK                                74
$
```

Table 3–20: Qualifiers for the DEFINE/FORM Command

Qualifier	Description
/DESCRIPTION=string	Assigns a description to the form. Can be as long as 255 characters, with the default being the name of the form.
/LENGTH=n	The physical form length in lines. The default is 66 lines.
/MARGIN=(option[,...])	Specifies one or more of the following four margin options: LEFT=n — The number of columns to be left blank between the left–most printing position and the actual print image area. RIGHT=n — The number of columns to be left blank between the /**WIDTH** setting and the actual print image area. TOP=n — The number of blank lines to leave between the top of the physical page of paper and the start of the print image. BOTTOM=n — The number of blank lines to leave between the end of the print image on a page and the end of the physical page of paper.
/PAGE_SETUP=(module[,...])	Specifies one or more modules that set up the printer before every page.
/SETUP=(module[,...])	Specifies one or more modules that set up the printer before a file is printed.
/SHEET_FEED	Tells the print symbiont associated with the queue to pause at the end of every physical page so that a new piece of paper can be inserted.
/STOCK=string	The name of the paper stock associated with the form. The string can be from 1 to 31 characters, including all letters, digits, the dollar sign, and the underscore. The default for string is the form name.
/TRUNCATE	Tells the printer to discard all characters that would exceed either the /**WIDTH** value or the /**MARGIN=RIGHT** value. You cannot specify both /**TRUNCATE** and /**WRAP** for the same form. Specify /**NOTRUNCATE** and /**NOWRAP** for special graphics output devices.
/WRAP	Tells the printer to print all characters exceeding either the /**WIDTH** value or the /**MARGIN=RIGHT** value on the next line.
/WIDTH=n	Specifies the physical width of the paper in terms of columns or character positions.

Using Forms and Characteristics with Printer Queues

Once a form or characteristic is defined, you can use the /**FORM_MOUNTED** qualifier with the following commands to associate it with a queue:

INITIALIZE/QUEUE
START/QUEUE
SET QUEUE

Table 3–21: Defining Printer Forms and Characteristics

Command†	Comments
`$ INITIALIZE/QUEUE -` `_$ /CHARACTERISTICS=(number or name,[,...])†`	Characteristics of jobs must match or be a subset of queue characteristics for the job to execute. The command **DEFINE/CHARACTERISTICS** sets up a correspondence between the characteristic name and number. Note that a characteristic must already be defined before it can be used with a queue.

`$ DEFINE/CHAR REDINK 2`
`$ INITIALIZE/QUEUE/START/CHAR=REDINK PLOTTER`
`$ PRINT/CHAR=REDINK/QUEUE=PLOTTER FILE.DAT`
	Set up the characteristic name REDINK to correspond to the number 2. Specify the REDINK characteristic as a characteristic of the PLOTTER queue. Only jobs sent to this queue that specify the REDINK characteristic by name (or number) will be executed from this queue. Other jobs will be placed on hold.

`$ INITIALIZE/QUEUE/FORM=(number or name,[,...])`	Limits jobs to those whose form name (or number) matches that of the queue. Note that a form must already be defined before it can be used with a queue.
`$ DEFINE/FORM WIDE 2` `$ INITIALIZE/QUEUE/START/FORM=WIDE LPA0`	Define a new form called WIDE and initialize (and start) the LPA0 queue with this new form.
`$ PRINT/FORM=WIDE/QUEUE=LPA0 FILE.DAT`	Submit a request to print FILE.DAT on the LPA0 queue with the WIDE form.

†It is generally considered a bad practice to specify forms or characteristics by their numbers. You should strive to use the form or characteristic name to better document the intended action and to keep from inadvertently specifying the wrong form or characteristic.

Handling Print Queue Problems

- Problems can arise with printers:
  - — Paper jams
  - — Paper ran out
  - — Ribbon tear or poor print density
- You might have to reprint part or all of the job
- The commands needed to reprint problem jobs can also be used to set up a printer for special print situations
  - — Bank check printing
  - — Invoices
- Use positioning and alignment qualifiers to handle these situations

Table 3–22: Positioning a Print Job

Qualifier to START/QUEUE	Comments
`/BACKWARD=n`	File is backspaced n pages before printing is resumed.
`/FORWARD=n`	File is forward spaced n pages before printing is resumed.
`/SEARCH=string`	Resumes printing with page containing string. (Search direction is forward. Other qualifiers processed first.)
`/TOP_OF_FILE`	Printing begins at top of interrupted file (not top of job).

Table 3–23: Aligning Printer Paper

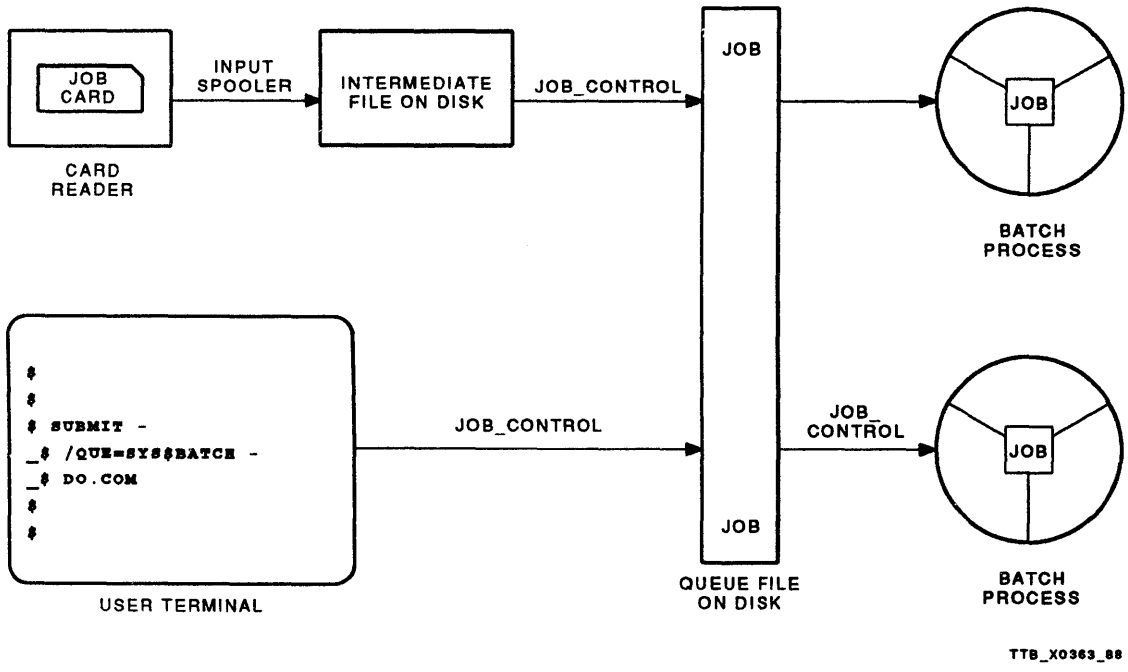
Command Format/Examples	Comments
`$ START/QUEUE/ALIGN -` `_$ queue-name`	One page of the job is printed. The queue stops. Adjust the paper and restart the queue.

`$ START/QUEUE/BACKWARD=2 -`
`_$ /ALIGN=2 LPA0`
`$ START/QUEUE LPA0`
	You can back up several pages before beginning the reprint. In this example, the symbiont backs up two pages in the job, then prints two alignment pages and stops. The user adjusts the paper and restarts the queue. The system begins printing the next page in the job.

HOW VMS HANDLES BATCH JOBS

- Batch queues solve waiting problems and make best use of terminal and card reader resources
- **SUBMIT** command places batch job in queue
- Batch queues must exist for VMS to execute batch jobs
- Batch queue information stored in SYS$SYSTEM:JBCSYSQUE.DAT

Figure 3–8: JOB_CONTROL Process Handles All Batch Jobs



- When the JOB_CONTROL process schedules a batch job to execute, it creates a batch process in which to execute the job
  - — Name of process composed of word BATCH and job's queue entry number
  - — Jobs listed in queue as currently executing
  - — Batch process appears in **SHOW SYSTEM** display with "B" in right hand column

Example 3–8: JOB_CONTROL, Input Symbiont, and Batch Job Processes

```
$
$ SHOW SYSTEM
VAX/VMS V5.0  on node BROWNY 13-DEC-1987 12:12:24.68   Uptime   12 22:30:54
  Pid     Process Name    State  Pri      I/O       CPU        Page flts Ph.Mem
20200021 SWAPPER         HIB     16        0   0 00:00:15.79         0      0
202001A2 Ed Bernstein    LEF      6    17322   0 00:05:08.44      7888    997
20200026 ERRFMT          HIB      9     3206   0 00:00:41.04        71     95
20200027 CACHE_SERVER    HIB     16        6   0 00:00:00.17        61     91
20200028 CLUSTER_SERVER  HIB      8        6   0 00:00:00.41       102    231
20200029 OPCOM           LEF      8    40150   0 00:09:32.53     26113    132
1 202001B8 BATCH_103     CUR      3       80   0 00:00:02.93       588    259  B  2
2020002A SMISERVER       HIB      9       68   0 00:00:02.09       220    614
3 2020002B JOB_CONTROL   HIB      9     1875   0 00:00:14.57       207    412
2020002C CONFIGURE       HIB     10        7   0 00:00:00.52        93    128
2020002D SYMBIONT_0001   HIB      6      132   0 00:00:05.58      1129     46
20200196 JK Martin       LEF      4      985   0 00:00:26.96      5140    160
20200039 NETACP          HIB     10    25671   0 05:56:18.63  11846645   1500
4 202001A3 _CRA0:        LEF      4     2501   0 00:00:58.87        50     41
2020003A EVL             HIB      6    43834   0 00:10:08.40    544317     50  N
2020003B REMACP          HIB      9      122   0 00:00:00.94        76     45
$
$
```

Notes on Example 3–8

1 Batch process.

2 The letter "B" indicates that this process is executing a batch job.

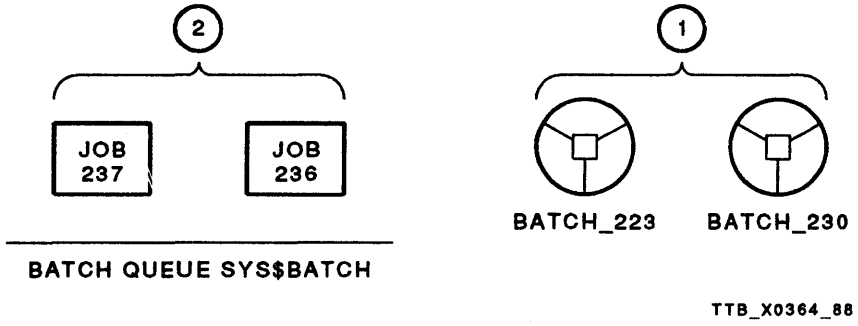
3 The JOB_CONTROL process.

4 The input symbiont process name is the same as the device name of the card reader from which the job is read.

Batch Job Scheduling

- System manager limits number of jobs that can run concurrently from batch queue
- JOB_CONTROL schedules batch jobs according to priority and submission time:
  - — Jobs with highest queue priority execute first
  - — Jobs execute in order of submission if they have the same queue priority
- Queue priority of job affects when job is scheduled to execute
- Base priority of batch process defined either by:
  - — BASE_PRIORITY queue attribute
  - — DEFPRI system parameter (if BASE_PRIORITY not specified)

Figure 3–9: Jobs on a Batch Queue



Example 3–9: Current and Pending Jobs on a Batch Queue

```
$
$ SHOW QUEUE/FULL SYS$BATCH
Batch queue SYS$BATCH
    /BASE_PRIORITY=3 /JOB_LIMIT=2 /OWNER=[SYSTEM] /PROTECTION=(S:E,O:D,G:R,W:W)

  Jobname          Username     Entry  Status
  -------          --------     -----  ------
  ACTION           JONES          230  Executing
1   Submitted 13-DEC-1987 12:44 /PRIORITY=4
      _DJA0:[JONES]ACTION.COM;2 (executing)

  MATH             JONES          233  Executing
    Submitted 13-DEC-1987 12:57 /PRIORITY=4
      _DJA0:[JONES]MATH.COM;1 (executing)

2 COMPUTE          JONES          237  Pending
    Submitted 13-DEC-1987 13:41 /PRIORITY=6
      _DJA0:[JONES]COMPUTE.COM;7 (pending)

3 ACTION           JONES          236  Pending
    Submitted 13-DEC-1987 13:10 /PRIORITY=4
      _DJA0:[JONES]ACTION.COM;3 (pending)

$
$
```

Notes on Figure 3–9 and Example 3–9

1 Jobs 230 and 233 are currently executing. The JOB_LIMIT attribute of the queue has a value of two, limiting the number of concurrent batch processes from this queue to two. Because the number of concurrent batch processes running from SYS$BATCH equals its job limit, Job 236 and Job 237 must wait to execute. Their status is pending. The JOB_CONTROL process examines the parameters of the pending jobs to determine which job it will execute next.

2 Since Job 237 has a higher queue priority than Job 236, (see note 2 of Example 3–2) it will be executed next.

NOTE

Although the queue priority of a batch job helps to determine when it will be scheduled, the queue priority does not affect the base priority of the batch process. The base priority of batch processes is defined by:

- **The value of the BASE_PRIORITY queue attribute. In this case, two.**
- **The value of the system parameter DEFPRI. The system uses this value if you do not set the BASE_PRIORITY attribute for a queue.**

3 Finally, Job 236 executes. However, if a user submits another job before Job 236 begins executing, the JOB_CONTROL process will compare the parameters of the new job with those of Job 236 to determine which job it will execute next.

BATCH QUEUE OPERATIONS

- Very similar to print queue operations
- Differ only slightly in creating and stopping queues

Types of Batch Queues

- Two types of batch queues:

Execution	Executes batch job
Generic	Feeds execution queues

Creating Batch Queues

- Default system batch queue called SYS$BATCH
  - — Default queue for **SUBMIT** command
  - — Default queue for spooled jobs from card readers
- You must create all batch queues (none are automatically created)
- The /**BATCH** qualifier is **required** to create a batch queue
- Different kinds of batch queues can be created for different kinds of environments

Table 3–24: Batch Queue Names and Parameter Values

Suggested Name and Purpose of the Queue	Parameters and Comments
GROUP360 To be used by those whose group UIC is 360 because they are working on a high–priority project.	Set the owner UIC of the queue to [360,000]. Give GROUP users READ and WRITE access only. Give WORLD users no access. Set larger working set extent, CPU limits, and priority. Possibly increase the job limit.
FASTQUE To be used by all who need a job done quickly.	Set the base priority at 5. Use default queue protection or possibly restrict use to a certain group, UIC, or ACL. Set job limit at 2, but limit the maximum CPU to a low value to keep the queue from taking over the system. Optionally use higher working set limits.
SLOWQUE To be used by all who want to run a batch job that affects system performance as little as possible.	Set the priority at 3. DO NOT set to 0 or 1, as jobs may get very little CPU time and finish too slowly. Use default queue protection. Set job limit to 1. Use default working set limits.
ZOOMQUE Very fast queue that you only start after hours or during lunch.	Set the base priority at 6 or higher, but definitely set a low CPU limit. Use default queue protection. Set job limit to 2. Set high working set limits. You should note, however, that setting a batch queue's default BASE_PRIORITY to a value higher than the normal interactive value is generally considered somewhat dangerous to system response time. You should carefully monitor any batch queue running in this fashion to avoid system degradation.
CADCAM To be used by large, compute–intensive applications, such as engineering, manufacturing, or modeling programs.	Set up as shown with SLOWQUE, but increase working set limits to large values. Set CPU limit to very large value, or infinite. Optionally protect the queue for access by only specific user groups.

Table 3–25: Qualifiers to INITIALIZE/QUEUE for Batch Queues

Qualifiers	Examples	Comments
/JOB_LIMIT	`$ INITIALIZE/QUEUE/BATCH -` `_$ /JOB_LIM=2 FASTBAT`	Sets a limit on the number of batch processes that can run concurrently from one batch queue. The default is 1.
/BASE_PRIORITY	`$ INITIALIZE/QUEUE/BATCH -` `_$ /BASE=5 FASTBAT`	Defines the base priority of a batch process. The system parameter DEFPRI sets the default. The higher the priority, the sooner it will be scheduled to run. If many batch processes have priorities as high as, or higher than, interactive process priorities (default 4), they can degrade the performance for interactive users and the whole system.

/PROTECTION
/OWNER_UIC
	`$ INITIALIZE/QUEUE/BATCH -`
	`_$ /OWN=[ENG,PROJ5] -`
	`-$ /PROT=(S:RWED)`
		Limits access to a queue.

/WSDEFAULT
/WSQUOTA
/WSEXTENT
/DISABLE_SWAPPING
	`$ INITIALIZE/QUEUE/BATCH -`
	`_$ /WSDEF=500 -`
	`_$ /WSQUOT=800 -`
	`_$ /WSEXT=2000 -`
	`_$ /DISABLE_SWAP -`
	`_$ FASTBAT`
		Defines memory management parameters for the batch process (limits working set size and adjustment allowed; sets swap or noswap). UAF values are the default. If you set high values for these and disable swapping of processes for the queue, you can use so much memory that system performance is degraded (because paging and swapping increases for interactive jobs).

/CPUMAXIMUM	`$ INITIALIZE/QUEUE/BATCH -`
	`_$ /CPUMAX=INFINITE -`
	`_$ BIGJOBBAT`
		Sets the maximum CPU limit to be assigned to a batch process from the queue. The default maximum is the CPU limit in the owner's UAF record.

/CPUDEFAULT	`$ INITIALIZE/QUEUE/BATCH -`
	`_$ /CPUDEF=00:03:00 -`
	`_$ 3MINBAT`
		Sets default CPU limit assigned to batch processes from this queue. Otherwise, the default is the CPU limit in a users' UAF record or the value of **/CPUMAXIMUM** for the queue.

Stopping Batch Queues

- Can stop or requeue currently executing job without adversely affecting other jobs in queue
  - — Need entry number for batch jobs:

```
$ STOP/ENTRY=job-number queue-name
$ STOP/ENTRY=205 SYS$BATCH
```

- Suspend current jobs when stopping queue: **$ STOP/QUEUE**
- Complete current jobs before stopping queue: **$ STOP/QUEUE/NEXT**

Example 3–10: Stopping Batch Queues

```
$
$ SUBMIT ACTION.COM
Job ACTION (queue SYS$BATCH, entry 911) started on SYS$BATCH
$
$ STOP/QUEUE SYS$BATCH
$
$ SHOW QUEUE/ALL SYS$BATCH
Batch queue SYS$BATCH, paused

  Jobname          Username     Entry  Status
  -------          --------     -----  ------
  ACTION           JONES          911  Executing
$
$ SHOW SYSTEM/BATCH
VAX/VMS V5.0  on node BROWNY 13-DEC-1987 13:30:24.38   Uptime   12 22:30:54
  Pid    Process Name    State  Pri      I/O       CPU        Page flts Ph.Mem
202003C5 BATCH_911       SUSP     4       25   0 00:00:00.71       143    171  B
$
$ START/QUEUE SYS$BATCH
$
$ STOP/QUEUE/NEXT SYS$BATCH
$
$ SHOW QUEUE/ALL SYS$BATCH
Batch queue SYS$BATCH, stop pending

  Jobname          Username     Entry  Status
  -------          --------     -----  ------
  ACTION           JONES          911  Executing
$
$ SHOW SYSTEM/BATCH
VAX/VMS V5.0  on node BROWNY 13-DEC-1987 13:31:46.62   Uptime   12 22:31:33
  Pid    Process Name    State  Pri      I/O       CPU        Page flts Ph.Mem
202003C5 BATCH_911       LEF      4       25   0 00:00:00.71       143    171  B
$
$ SHOW QUEUE/ALL SYS$BATCH
Batch queue SYS$BATCH, stopped
$
$ SHOW SYSTEM/BATCH
$
```

Restricting Access and Control of Queues

- Two methods to specify access and control of queues:
  - — UIC–based protection
  - — ACL–based protection
- By default, UIC–based protection is used
- Default protection code for queues: `(S:E,O:D,G:R,W:W)`

UIC–Based Queue Protection

- System uses queue protection mask and owner UIC of queue to restrict access and control
- Users placed in categories according to their UIC
- Each user category has all privileges assigned to lower categories

Table 3–26: Categories of User

Category	Comments
System	User has a system UIC
Owner	User UIC matches queue's owner UIC
Group	Group UIC number of user matches group UIC number of queue owner UIC
World	User UIC does not match queue's owner UIC

Table 3–27: Access to Queues

Access Codes	Access Allowed
Read	Can list the attributes of a job
Delete	Can modify or delete any job, or delete the queue
Write	Can submit jobs to the queue
Execute	Can affect the queue and/or all jobs in the queue (users with OPER privilege have E access to all queues)

ACL-Based Protection

- In addition to UIC-based protection, a queue can be assigned ACLs
  - — Provides finer specification of access and control
  - — Allows access to specific users who cross UIC groups
- Works in much the same manner as ACLs for files
- Offers additional capability through the **Execute** right
  - — Provides for "local" operator capabilities

Table 3–28: Preparing a Privately Controlled Printer Queue

Operation	Command	Comments

Create identifiers	`$ SET DEFAULT SYS$SYSTEM`
	`$ RUN AUTHORIZE`
	`UAF> ADD/ID MFG_PRT`
	`UAF> ADD/ID MFG_PRT_OPER`
		Create the MFG_PRT identifier for users allowed to submit jobs to the print queue, and the MFG_PRT_OPER for users responsible for management of the queue.

Grant identifiers to proper users	`UAF> GRANT/ID MFG_PRT MFG_USER1`
	`UAF> GRANT/ID MFG_PRT MFG_USER2`
	`UAF> GRANT/ID MFG_PRT MFG_USER3`
	`UAF> GRANT/ID MFG_PRT ENG_USER1`
	`UAF> GRANT/ID MFG_PRT ENG_USER2`
	`UAF> GRANT/ID MFG_PRT_OPER MFG_USER1`
	`UAF> EXIT`
		Associate the MFG_PRT identifier with the five users, and give MFG_USER1 the additional MFG_PRT_OPER identifier.

Set queue protection†	`$ SET QUEUE/PROTECTION=(S,O,G,W) MFG_LN03`
	
	`$ SET ACL/ACL=(IDENTIFIER=MFG_PRT_OPER,`
	`-`
	`_$ ACCESS=READ+WRITE+DELETE+EXECUTE) -`
	`_$ /ACL=(IDENTIFIER=MFG_PRT, -`
	`_$ ACCESS=READ+WRITE+DELETE) -`
	`_$ /OBJECT=QUEUE MFG_LN03`
		First, set the printer queue for NO ACCESS by any user by specifying a UIC mask with no access parameters. Next, associate the two previously created identifiers with the MFG_LN03 printer queue. Note that the EXECUTE access type is what gives the MFG_USER1 special queue control capabilities, such as starting and stopping the queue, modifying or deleting jobs, and so on.

†This example assumes the MFG_LN03 printer queue has already been created using one of the previously described queue creation procedures.

OVERVIEW OF QUEUE COMMANDS

Most queue commands require either OPER privilege or Execute (E) access to the target queue.

Table 3–29: Summary of Queue–Related DCL Commands

DCL Command	Command Description
Creating/Controlling/Deleting Queues	
INITIALIZE/QUEUE	Creates and initialize a queue
ASSIGN/QUEUE	Assigns a queue to a device
ASSIGN/MERGE	Moves jobs from one queue to another
START/QUEUE	Starts or restart a queue
STOP/QUEUE	Controls queue or current entry in it
DEASSIGN/QUEUE	Deassigns a queue from a device
DELETE/QUEUE	Deletes a queue and all its entries
Setting Queue Attributes	
SET QUEUE	Sets various queue parameters
SET ACL/OBJECT=QUEUE	Sets the access rights of a queue
EDIT/ACL/OBJECT=QUEUE	Edits the access rights of a queue
DEFINE/FORM	Sets the characteristics of a form
DEFINE/CHARACTERISTC	Defines the characteristic name and number in the system table
Setting Job Attributes	
PRINT	Places an entry in a print queue
SUBMIT	Places an entry in a batch queue
SET ENTRY	Changes the status of a pending entry in a queue
DELETE/ENTRY	Deletes a pending entry from a queue
Monitoring Queue and Entry Status	
SHOW QUEUE	Displays status of entries in a queue
SHOW ENTRY	Displays status of an individual job entry

SUMMARY

- Queue facilities managed by JOB_CONTROL process
  - — Started with **START/QUEUE/MANAGER**
- Queue information is kept in SYS$SYSTEM:JBCSYSQUE.DAT
- Two classes of queues
  - — Execution
  - — Generic
- Three types of queues
  - — Execution
  - — Generic
  - — Logical
- Two types of execution queues
  - — Output

    Printer
    Terminal
    Server
  - — Batch
- Two types of generic queues
  - — Output
  - — Batch
- Logical queues
  - — A type of output execution queue
  - — Has its output redirected to another execution queue
- Queues are created (and optionally started) with **INITIALIZE/QUEUE**
- Queues are started with **START/QUEUE**
  - — Typically done in system startup command procedure

- Monitor queues and jobs with **SHOW QUEUE** and **SHOW ENTRY**
- Set queue attributes with **INITIALIZE/QUEUE, SET/QUEUE,** and **START/QUEUE**
- Control queues with **STOP/QUEUE** and **SET QUEUE**
- Use separation pages to better distinguish between print jobs and their owners
  - — Job separation pages are controlled by the system manager
  - — File separation pages are specified by the system manager, controlled by the user
- Limit print job sizes to better control printer usage
- Use printer forms and characteristics to better manage printers and control output format
- Use several batch queues with varying parameters to improve turnaround time and minimize system degradation
- Use UIC and ACL protection to improve queue security and management

MODULE 4
MANAGING DISK AND TAPE VOLUMES

INTRODUCTION

System managers must decide how to allocate the disk and tape devices on their systems. They normally allocate some for public use and some for private use.

Public volumes are disk volumes that all users on the system can access. They contain system code, system data, and files useful to all users. Private volumes are disk and tape volumes that only a single user or group of users can access. Private volumes typically contain user–specific or group–specific code or data.

The system manager coordinates disk and tape management with the system's users and operators. The manager decides whether a volume will be public or private. The manager loads, initializes, mounts, unloads, and maintains public volumes. Users (possibly with the help of the manager or an operator) do the same for private volumes.

Maintenance of public volumes includes:

- Allocating disk space to users
- Tracking the use of disk space (with the SYSMAN utility)
- Scheduling backups
- Installing system software and layered product software on the volume
- Monitoring device errors, (MODULE 8, discusses monitoring device errors in more detail.)

The manager also helps maintain private volumes by:

- Scheduling a disk or tape device for private use
- Training users to use the equipment properly (loading volumes, starting the drive, using the volume, and unloading it when the job is done)
- Responding to **REQUEST/REPLY** commands in system environments where users are not allowed access to the equipment

OBJECTIVES

To share limited disk resources among users and processes, a system manager should be able to:

- Describe the uses of public disk and tape volumes
- Manage volumes, including:
  - — Prepare volumes for use
  - — Obtain and modify volume information
  - — Use public volumes properly
  - — Control the allocation of public volumes
  - — Create a volume set
  - — Maintain public and private volumes
  - — Use the BACKUP utility to back up and restore information on volumes
- Use the VERIFY utility to check the validity of the file structure on a disk.
- Transfer files between the VMS system and PDP–11 systems.

RESOURCES

To complete this module, you must have access to the following documents:

1. *Guide to Maintaining a VMS System*
2. *VMS DCL Dictionary*
3. *Guide to Setting Up a VMS System*
4. *VMS Exchange Utility Manual*
5. *VMS Backup Utility Manual*
6. *VMS Bad Block Locator Utility Manual*
7. *VMS Analyze/Disk_Structure Utility Manual*
8. *VMS Mount Utility Manual*

To complete the Laboratory Exercises in this module, you must have access to at least one of the following:

- A blank tape volume and a tape device
- A blank disk volume and a corresponding disk device

REFERENCES

The following documents contain information related to the material presented in this module:

1. *VMS Installation and Operation Guide* for your specific VAX processor
2. *Guide to VMS Files and Devices*
3. *Guide to Using VMS Command Procedures*
4. *VMS I/O User's Reference Manual: Part I*
5. *VMS I/O User's Reference Manual: Part II*
6. *RMS–11 User's Guide*

PRIVATE DISK AND TAPE VOLUMES

- Owned by a single user or group of users
- Used to:
  - — Preserve files
  - — Transfer files
  - — Provide a private storage environment to work in

PUBLIC VOLUMES

- Normally owned by SYSTEM
- Provides storage space for system files and user files
- Accessible to all users on system

DCL COMMANDS FOR USING DISK AND TAPE VOLUMES

- Manage disk and tape volumes using DCL commands
  - **ALLOCATE**
  - **DEALLOCATE**
  - **ANALYZE/MEDIA**
  - **ANALYZE/DISK_STRUCTURE**
  - **INITIALIZE**
  - **MOUNT**
  - **DISMOUNT**
  - **SET VOLUME**
  - **BACKUP**

Figure 4–1: Volume Manipulation Commands

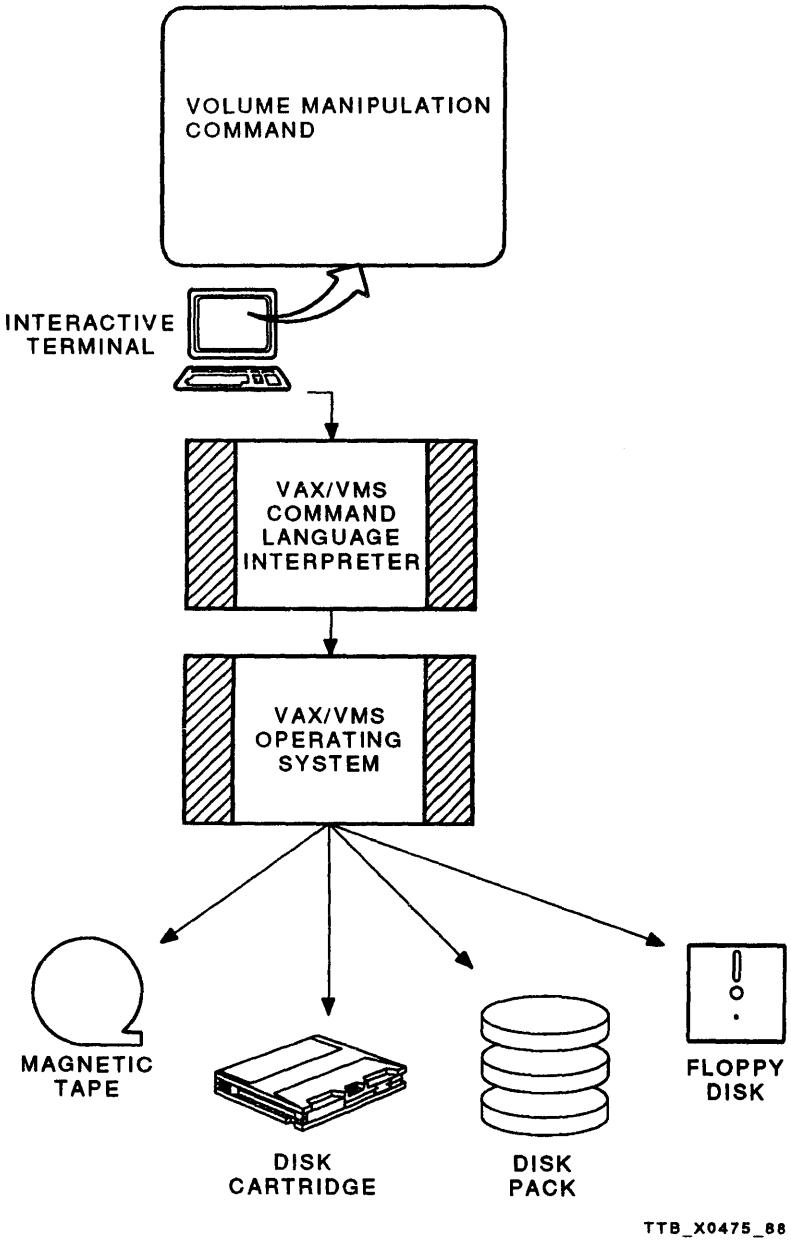
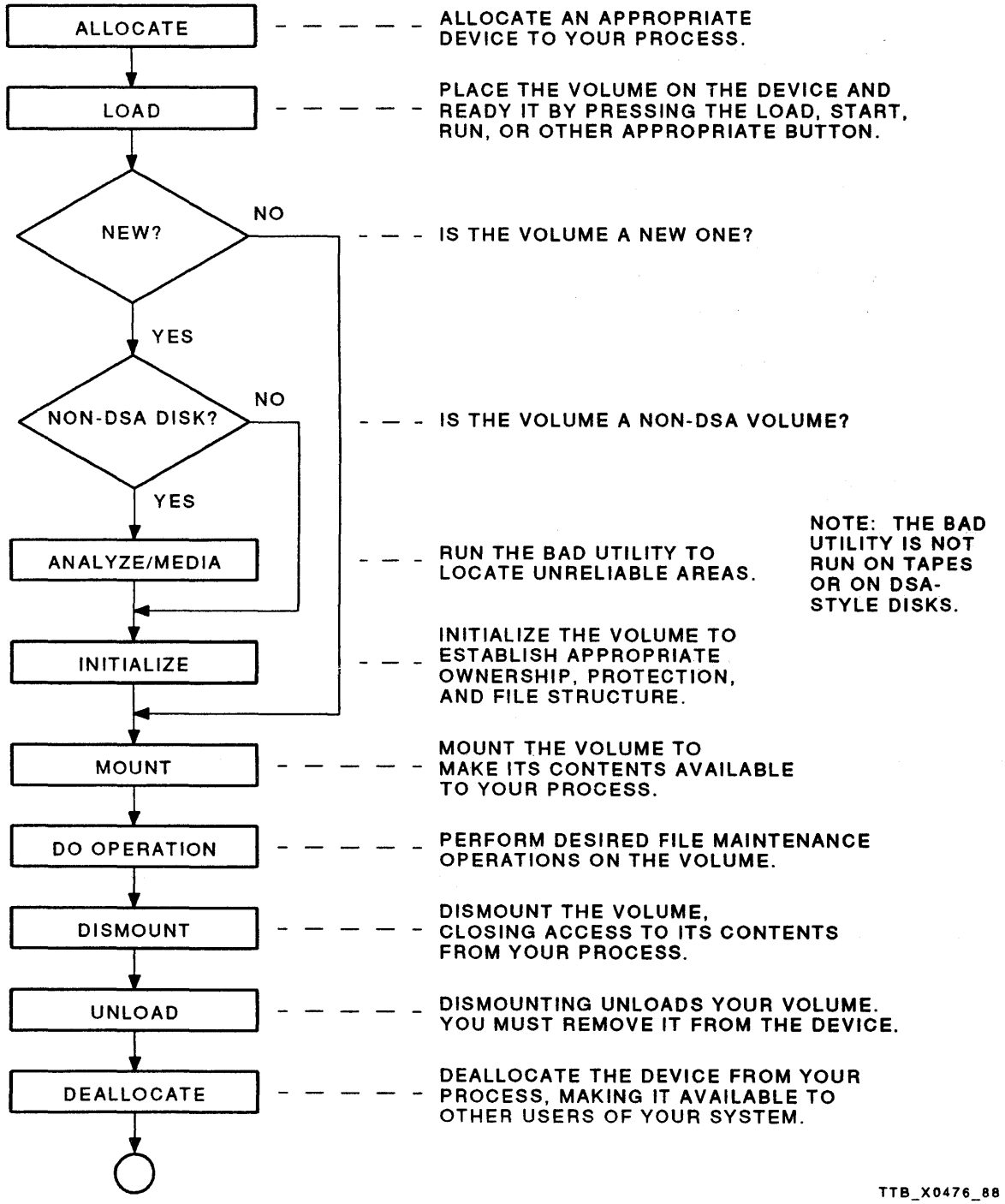


Figure 4–2: Preparing and Using a Disk or Tape Volume



PREPARING VOLUMES FOR USE

Locating Bad Blocks

- Bad Block Locator utility (BAD)
  - — Run on each new disk (ANALYZE/MEDIA)

    Locates unreliable disk areas (bad blocks)
    Records locations of bad blocks
    Allocates bad blocks to [000000]BADBLK.SYS
  - — Not run on disks with revectoring capability (DSA disks)
  - — Not run on tapes

Table 4–1: The BAD Utility

Command	Comment
`$ MOUNT/FOREIGN DBA1:`	The volume must be mounted with the /**FOREIGN** qualifier.
`$ ANALYZE/MEDIA/OUTPUT=BAD.LOG DBA1:`	You can produce an output listing of the bad blocks on the volume without affecting the information stored on the volume.
`$ ANALYZE/MEDIA/EXERCISE/LOG DBA1:`	The /**EXERCISE** qualifier causes read/write checks to be performed on the volume. All previously stored data is destroyed. As it detects each bad block, the utility sends a message to the user because the /LOG qualifier was included.

Initializing and Mounting the Volume

- **INITIALIZE** defines volume parameters
  - — Owner
  - — Label
  - — Protection code
  - — Cluster size
- **MOUNT** identifies volumes to system
  - — Makes volumes available for processing
  - — Enables disk and tape operations

Defining User Access to Volumes

- Operations limited by volume protection code
  - — Default for disk volumes: `(S:RWED, O:RWED, G:RWED, W:RWED)`
  - — No default protection for tape volumes
  - — Specify UIC and protection code for volume when initializing volume
- Users with VOLPRO privilege can gain access to any volume regardless of its owner UIC and protection code.

Figure 4–3: User Access to Files on Disk and Tape Volumes

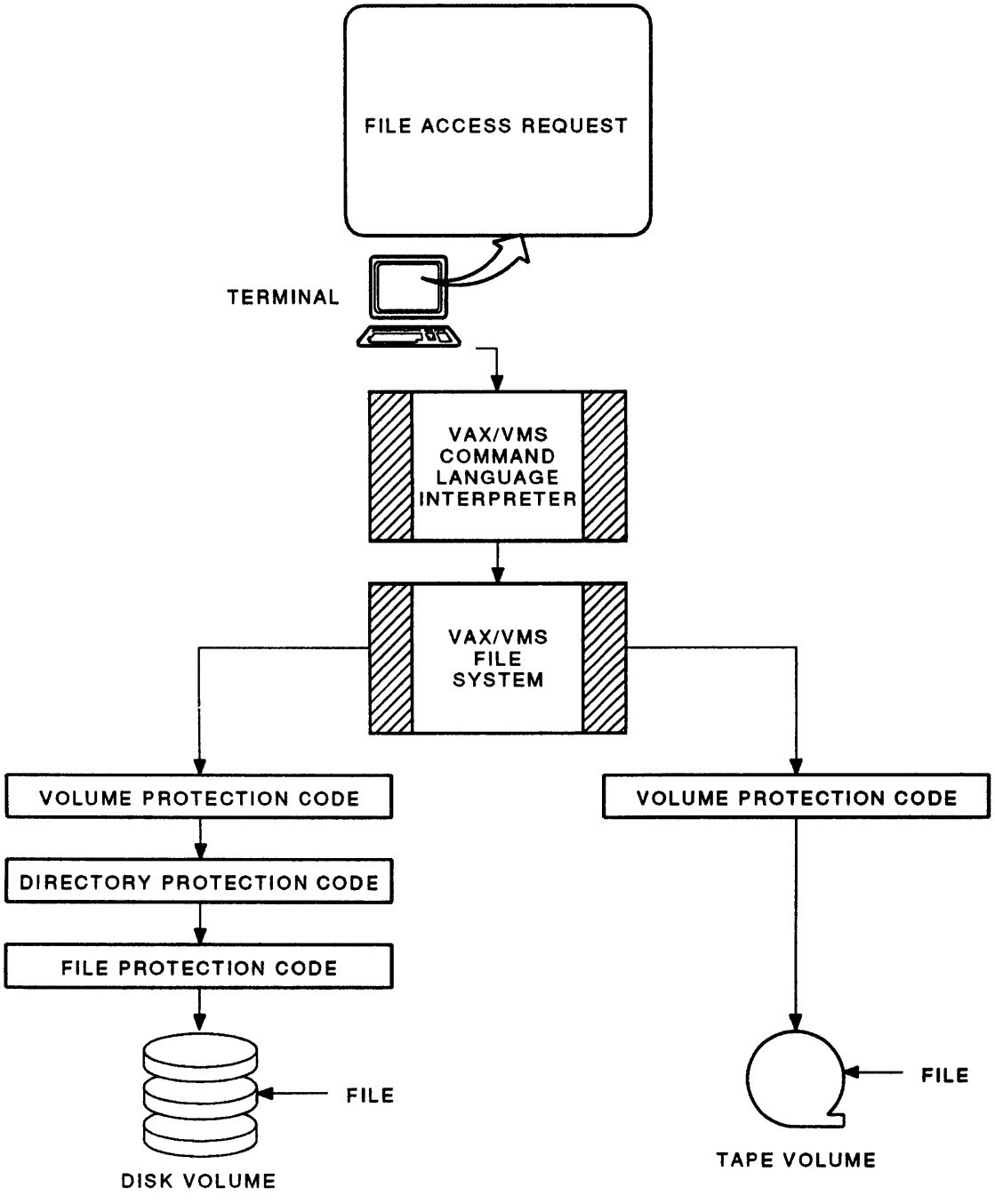


Table 4–2: Summary of Effects of Access Rights to Files

(R)ead	(W)rite	(E)xecute	(D)elete	(C)ontrol†
Disk Volumes				
Can read files on the volume	Can modify files on the volume	Can create files on the volume	Can delete files from volume	Does not apply
Disk Directories				
Can list files in directory with or without using wildcards	Can write to the directory file (Requires READ access to create or remove files or do other operations that would change the directory)	Can list files by name only (Can not use wildcards to list files)	Can delete the directory	Can change the characteristics of the directory (**SET DIRECTORY**)
Disk Files				
Can read contents of file (Read access implies execute access)	Can modify the contents of a file	Can execute executable files	Can delete or rename the file	Can change the characteristics of the file. (**SET FILE**)
Tape Volumes				
Can read list of files on tape	Can add files on the volume (Write access implies read access)	Does not apply	Does not apply	Does not apply

†Control access does not appear in the protection mask for a file. It is never granted to the GROUP or WORLD categories of users; it is always granted to SYSTEM and OWNER users.

- To establish protection during initialization process, one of the following must apply:
  - — Volume must be blank (new)
  - — You must own the volume
  - — You have VOLPRO privilege
- Define volume protection with the **INITIALIZE** command:

```
$ INITIALIZE [/qualifier,...] device-spec label
```

Table 4–3: Defining Specific Volume Protection Codes During Initialization

Qualifier	Effects	Comments
None	Owner UIC for disk: none Owner UIC for tape: UIC of current user	On disks, the owner UIC field is empty.
	Protection on disk: (S:RWED,O:RWED,G:RWED,W:RWED)	The default protection code allows access to all users.
	Protection on tape: none	System does not record a protection code on tape unless you specify one.
/PROTECTION=code	Owner UIC: UIC of current user Protection: specified in code	The code is in the same format as file protection codes. For tapes, only Read and Write access are meaningful. Also, System and Owner always have R and W access to a tape regardless of what is specified. For disks, E access allows create access. Unless you specify protection for tapes, all users have access.
/OWNER_UIC=uic	Owner UIC: UIC specified Protection: no effect on code	VMS compares the UIC of the process attempting access to the owner UIC and protection code of the volume.

Table 4–4: Establishing Predefined Volume Protection Codes During Initialization

Qualifier†	Effects	Comments
/NOSHARE	Owner UIC: UIC of current user Protection: (S:RWED,O:RWED,G,W)	Only system and owner have access
/GROUP/NOSHARE	Owner UIC: [g,0]‡ Protection: (S:RWED,O:RWED,G:RWED,W)	Expands access to group members as well
/GROUP	Owner UIC: [g,0]‡ Protection: (S:RWED,O:RWED,G:RWED,W:RWED)	Expands access to all users
/SYSTEM	Owner UIC: [1,1] Protection: (S:RWED,O:RWED,G:RWED,W:RWED)	Gives all users access, but only system users can create top–level directories. Public volumes are typically mounted using the /SYSTEM qualifier so the manager can govern the use of disk space.

†Can not combine qualifiers except as shown

‡g = group number of current user

- To specify protection during mount process, one of the following must apply:
  - — You own the volume
  - — You have VOLPRO privilege
- Override established volume protection with the **MOUNT** command:

```
$ MOUNT [/qualifier,...] device-spec label logical-name
```

Table 4–5: Overriding Volume Protection Codes Established at Initialization

Purpose	MOUNT Command Qualifier	Effects/Required Privilege
To override volume parameters as volume is mounted:		
Override owner UIC of tape or disk	`/OWNER_UIC=uic`	Specified UIC is owner while volume is mounted (does not modify UIC written on volume) Volume ownership or VOLPRO privilege required
Override disk protection code	`/PROTECTION=code`	Specified code is protection code while volume is mounted (does not modify protection code written on volume) Volume ownership or VOLPRO privilege required

Table 4–6: Qualifiers to the MOUNT Command Specifying User Access to a Volume

Qualifier	Comments
Mount a disk for private use	
none	The user must own the volume or have VOLPRO privilege. If the user has VOLPRO privilege but does not own the volume or know the label, he or she must include the **/OVERRIDE=IDENTIFICATION** qualifier with the **MOUNT** command to access the volume. If a qualifier is not included in the **MOUNT** command to allow access to other users, only the owner can access the volume while it is mounted regardless of the protection code.
Make a disk accessible to your group	
/GROUP	If the volume was initialized as a GROUP volume to set the protection code, but is not mounted **/GROUP**, group users can not access it. Assigns logical name in group table (GRPNAM privilege required). Group members can gain access to the volume without mounting it because name is in group logical name table. Group members must still pass the volume protection code.
Make disk accessible to all users on the system	
/SYSTEM	If the volume was initialized as a SYSTEM volume to set the protection code, but is not mounted **/SYSTEM**, users can not access it. Assigns logical name in system table (SYSNAM privilege required). All users can gain access to the volume without mounting it. Users must still pass the volume protection code. Example 4–2 uses the **/SYSTEM** qualifier.
Make disk available to all users on condition	
/SHARE	Other users must issue a **MOUNT/SHARE** command. Other users must still pass the volume protection code.

OBTAINING AND MODIFYING VOLUME INFORMATION

- List characteristics of a volume

```
$ SHOW DEVICE/FULL
```

- Modify characteristics of a volume

```
$ SET VOLUME
```

Table 4–7: Obtaining and Modifying Volume Information

Operation	Command/Example	Comments
Displaying the characteristics of the volume	`$ SHOW DEVICE/FULL device` `$ SHOW DEVICE/FULL MYDISK`	Displays the characteristics of the volume currently mounted on the device.
Modifying the characteristics of the volume	`$ SET VOLUME/qualifier volume-name` `$ SET VOLUME/LABEL=MAY10BCK MYDISK`	Requires VOLPRO privilege or volume ownership to change the characteristics

Example 4–1: The SHOW DEVICE/FULL Command

```
$ SHOW DEVICE/FULL DRA2

Disk DRA2:, device type RM03, is online, allocated, deallocate on dismount,
        mounted, error logging enabled.

    Error count                    0    Operations completed              250
    Owner process            "BIERLY"   Owner UIC                     [11,340]
    Owner process ID         00000094   Dev Prot  S:RWED,O:RWED,G:RWED,W:RWED
    Reference count                2    Default buffer size               512

    Volume label          "USER_DISK"   Relative volume no.                 0
    Cluster size                   3    Transaction  count                  1
    Free blocks               131589    Maximum files allowed           16460
    Extend quantity                5    Mount count                         1
    Mount status             Process    Cache name           "_DRA0:XQPCACHE"
    File ID cache size            64    Extent cache size                  64
    Quota cache size               0
    Write-thru caching enabled

Volume is subject to mount verification, file high-water marking.

$
```

CREATING A PUBLIC VOLUME

- Initializing a disk with the /**SYSTEM** qualifier
  - — Sets owner UIC to [1,1]
  - — Sets volume protection to: `(S:RWED, O:RWED, G:RWED, W:RWED)`
- Mounting a disk with the /**SYSTEM** qualifier
  - — Deallocates the device
  - — Places logical name specified in system logical name table
  - — Allows all users to access volume

Example 4–2: Initializing and Mounting a Public Disk

```
$ SHOW DEVICE DRA2

Device                  Device       Error     Volume      Free Trans Mnt
 Name                   Status       Count     Label      Blocks Count Cnt
DRA2:                   Online           0
$
$ ALLOCATE DRA2 PUBDISK
%DCL-I-ALLOC, _DRA2:  allocated
$ MOUNT/FOREIGN PUBDISK
%MOUNT-I-MOUNTED, USER_DISK     mounted on _DRA2:
$ ANALYZE/MEDIA/EXERCISE PUBDISK
$ DISMOUNT/NOUNLOAD PUBDISK
$ INITIALIZE/SYSTEM PUBDISK MYVOL
$ MOUNT/SYSTEM PUBDISK MYVOL
%MOUNT-I-MOUNTED, MYVOL         mounted on _DRA2:
$ SHOW LOGICAL/SYSTEM D*

(LNM$SYSTEM_TABLE)

   "DBG$INPUT" = "SYS$INPUT:"
   "DBG$OUTPUT" = "SYS$OUTPUT:"
   "DDP$DIS" = "SYS$MANAGER:DDP.DIS"
   "DISK$MYVOL" = "DRA2:"
   "DISK$VAXVMSRL3" = "DRA0:"
   "DTR$LIBRARY" = "SYS$SYSROOT:[DTR]"
$
```

Table 4–8: Commands for Creating and Accessing Private Disk and Tape Volumes

Command/Example	Comments
Allocating a device	
`$ ALLOCATE device [logical-name]` `$ ALLOCATE DU DISK`	Finds the first available disk drive of type DU and allocates it to your process. The system places the logical name DISK in your process logical name table, equating it to the name of the allocated device. Other users can not gain access to this device. Since you used the **ALLOCATE** command, you must use the **DEALLOCATE** command when you are done to make the device available to other users.
Finding unreliable areas on an unused disk	
`$ MOUNT/FOREIGN device` `$ ANALYZE/MEDIA/EXERCISE device`	Finds unreliable areas of the disk and records their locations in a special block. The process destroys all previous data on the disk. If the disk is not your own, VOLPRO privilege is required.
`$ DISMOUNT/NOUNLOAD device`	Since the volume is mounted as a foreign volume, you must dismount it so you can initialize it as a Files–11 volume.
Creating a file structure on a tape or disk	
`$ INITIALIZE device label` `$ INITIALIZE DUA2: TESTDISK`	Builds Files–11 On–Disk Structure on the volume loaded on the DUA2: device. The volume is labeled TESTDISK. You are declared owner of the disk; all user groups are allowed all types of access (RWED). You must own the disk, or possess VOLPRO privilege to initialize it.
`$ INITIALIZE MUA0: TSTTAP`	Builds current ANSI level tape structure on the volume located on device MUA0. The volume receives the label TSTTAP. By default, you are declared the owner; all user groups are allowed all types of access (RWED).
Creating a link between the volume and your process	
`$ MOUNT device label [logical-name]`	The system allocates the device to your process until you issue the **DISMOUNT** command.
`$ MOUNT DUA2: TESTDISK DISK`	Mounts TESTDISK on the device DUA2. The logical name DISK is assigned the equivalence name DUA2.

Table 4–9: Commands for Removing Private Disk and Tape Volumes

Command/Example	Comments
Breaking the link between the volume and your process	
`$ DISMOUNT [/NOUNLOAD] device`	If you want VMS to keep the volume on–line when you dismount it, use the qualifier /**NOUNLOAD**. The **DISMOUNT** command does not deallocate a device that you have allocated with the **ALLOCATE** command.
`$ DISMOUNT DUA2:`	Dismounts and automatically unloads the volume on DUA2. Deletes the logical name DISK assigned by the **MOUNT** command.
Deallocating a device	
`$ DEALLOCATE device`	
`$ DEALLOCATE DUA2:`	Deallocates device DUA2. Frees the device for use by other users. Does not delete a logical name assigned by the **ALLOCATE** command.

Defining Space Allocation on Public Volumes

Table 4–10: INITIALIZE Command Qualifiers that Affect Space Allocation

Qualifier	Comment
/CLUSTERSIZE=n	The minimum number of contiguous blocks to be allocated to each file.
/EXTENSION=n	The number of blocks to give to a file when it is extended.
/MAXIMUMFILES	The maximum number of files the disk volume can contain (once set, can only be changed by reinitializing the volume).
/DENSITY	The density (800, 1600 or 6250 bits/in) at which a tape is to be written. If writing RX02 floppies, density is SINGLE or DOUBLE.

ALLOCATION OF DEVICES

- System manager decides which devices hold private volumes and which hold public volumes

Private Volumes

- One or more drives set aside for private volumes
- When using a drive for more than one private volume:
  - — Run BAD utility on each volume while volume is in drive before initializing it.
  - — Periodically align and check drive for malfunctions
  - — Load/unload all volumes properly
  - — Back up volume before removing it from the drive (when practical)

Public Volumes

- Can contain system and user software
- Can provide space for user files
- Allocate space on volume using the SYSMAN utility
  - — Strive to keep system disk less than three–quarters full for maximum performance
  - — When obtaining optional software, check to see how many blocks of disk space it requires (Some configurations, due to lack of disk space, can not support all optional products desired.)
- Maintain security of system files by using protection codes and access control lists judiciously

Creating Volume Sets

- Create when database or user directories are too large for one volume
  - — Consists of one or more disk volumes
  - — System treats as one large volume
  - — VMS stores files on any volume in the set with available space, attempting to use space evenly over all volumes in the set
  - — Create from new or existing volumes
- To create volume sets
  - — Allocate the necessary devices
  - — Physically load the volumes
  - — Initialize each new volume
  - — Use the **MOUNT/BIND** command to create volume set from initialized volumes

Example 4–3: Creating a Volume Set from an Existing Volume

```
$ ALLOCATE DRA2 DEV1
%DCL-I-ALLOC, _DRA2: allocated
$ ALLOCATE DRA3 DEV2
%DCL-I-ALLOC, _DRA3: allocated
$ INITIALIZE/SYSTEM DEV2 USER2
$ MOUNT/SYSTEM/BIND=USER_SET DEV1,DEV2 USER1,USER2
%MOUNT-I-MOUNTED, USER1        mounted on _DRA2:
%MOUNT-I-MOUNTED, USER2        mounted on _DRA3:
$ SHOW LOGICAL/SYSTEM D*

(LNM$SYSTEM_TABLE)

   "DBG$INPUT" = "SYS$INPUT:"
   "DBG$OUTPUT" = "SYS$OUTPUT:"
   "DDP$DIS" = "SYS$MANAGER:DDP.DIS"
   "DISK$USER1" = "DRA2:"
   "DISK$USER2" = "DRA3:"
   "DISK$USER_SET" = "DRA2:"
   "DISK$VAXVMSRL3" = "DRA0:"
   "DTR$LIBRARY" = "SYS$SYSROOT:[DTR]"
$
$ COPY WORK1:[BROWN]EXAMP5.COM
_To: DISK$USER_SET:[SMITH]EXAMP5.COM
$ DIRECTORY DISK$USER_SET:[SMITH]

Directory DISK$USER_SET:[SMITH]

EXAMP5.COM;1

Total of 1 file.
$ DISMOUNT DEV1
$ SHOW LOGICAL/SYSTEM D*

(LNM$SYSTEM_TABLE)

   "DBG$INPUT" = "SYS$INPUT:"
   "DBG$OUTPUT" = "SYS$OUTPUT:"
   "DDP$DIS" = "SYS$MANAGER:DDP.DIS"
   "DISK$VAXVMSRL3" = "DRA0:"
   "DTR$LIBRARY" = "SYS$SYSROOT:[DTR]"
$
```

MAINTAINING PRIVATE AND PUBLIC VOLUMES

- Regular maintenance improves performance
  - — Back up public disks
  - — Look for lost files
- Use **MOUNT/OVERRIDE** command to mount volume with unknown label (need VOLPRO privilege unless you own the volume)

Example 4–4: Mounting a Disk with an Unknown Label

```
$ MOUNT/OVERRIDE=IDENTIFICATION DRA2 UNKNOWN MYDISK
%MOUNT-I-MOUNTED, PROGRAM_DISK mounted on _DRA2:
$ SHOW DEVICE/FULL MYDISK

Disk DRA2:, device type RM03, is online, allocated, deallocate on dismount,
        mounted, error logging enabled.

    Error count                    6    Operations completed            2643
    Owner process          "BIERLY"    Owner UIC                    [11,340]
    Owner process ID       00000094    Dev Prot S:RWED,O:RWED,G:RWED,W:RWED
    Reference count                2    Default buffer size              512

    Volume label    "PROGRAM_DISK"     Relative volume no.                0
    Cluster size                   3    Transaction count                  1
    Free blocks               131589    Maximum files allowed          16460
    Extend quantity                5    Mount count                        1
    Mount status             Process    Cache name          "_DRA0:XQPCACHE"
    File ID cache size            64    Extent cache size                 64
    Quota cache size               0
    Write-thru caching enabled

Volume is subject to mount verification, file high-water marking.

$ DISMOUNT MYDISK
$
```

On–Line BACKUP

- Use on–line BACKUP from any terminal where you are logged in
- Use on–line BACKUP commands to save:
  - — Individual files and their locations in a directory structure
  - — A directory structure and the files in it
  - — All directory structures and files on a volume
- To back up an entire volume, first mount it privately
- Normally, do not use on–line BACKUP to back up system volume
  - — Use standalone BACKUP to back up system volume
- **BACKUP** command line structure:

```
$ BACKUP [/qualifier...] input-specifier output-specifier
```

Table 4–11: BACKUP Terms

Term	Definition	Comments/Examples
Save set	A file in BACKUP format, created by the BACKUP utility.	The utility can create save sets on tape or disk volumes.†
Save–set–name	Any legal VMS filename, file type‡, and version number.	`SAVE.BCK;2`
Save–set–specifier	A device name and a save– set–name. The format is device:save–set–name.	`MUA0:SAVE.BCK;2`

Input–specifier	The input file specification (what you are saving) or save–set–specifier (what you are restoring from) in the **BACKUP** command.	`$ BACKUP -`
		`_$ [BROWN]FILE.DAT -`
		`_$ output-specifier`
		
		`$ BACKUP -`
		`_$ MUA0:SAVE.BCK;2 -`
		`$ output-specifier`

Output–specifier	The output save–set–specifier (where you are saving to) or file specification (where you are restoring to) in the **BACKUP** command.	`$ BACKUP -`
		`_$ input-specifier -`
		`_$ MUA0:SAVE.BCK`
		
		`$ BACKUP -`
		`_$ input-specifier -`
		`_$ DBA1:[BROWN]`

† If you are creating or using a save set on a disk volume, you must follow the name of the save set with the **/SAVE_SET** qualifier.

‡ There is no default file type. However, the .BCK file type is used by convention. The examples in this module use the file type .BCK.

BACKUP Qualifiers

- Five types of qualifiers
  - — Command qualifiers
  - — Input file–selection qualifiers
  - — Input save–set qualifiers
  - — Output file qualifiers
  - — Output save–set qualifiers
- Some qualifiers exist in more than one qualifier–type category

Table 4–12: BACKUP Qualifier Types

Qualifier Type	Description
Command qualifier	Modifies the default action of a BACKUP command. You can place this type of qualifier anywhere on the command line. The qualifier acts upon every file in the input or output specifier. (See Table 4–13)
Input file–selection qualifier	Selects files from the input specifier. Place immediately after the input specifier. (See Table 4–15)
Input save–set qualifier	Affects the way BACKUP handles an input save set during a restore operation. Place immediately after the input specifier. (See Table 4–16)
Ouput file qualifier	Changes the way output files are restored. Place them immediately after the ouput specifier. (See Table 4–17)
Output save–set qualifier	Affects the way BACKUP processes an output save set during a save operation. Place immediately after the output specifier. (See Table 4–18)

Table 4–13: BACKUP Command Qualifiers

Qualifier	Description
/[NO]ASSIST	Allows operator or user intervention if a request to mount a magnetic tape fails during a BACKUP process.
/BRIEF	Used with **/LIST** to produce an abbreviated listing of files in the save set.
/BUFFER_COUNT=n	Specifies the number of I/O buffers to be used in the BACKUP operation.
/COMPARE	Compares the save set, device, file, or files specified by the input specifier with the save set, device, file, or files specified by the output specifier and displays an error message if it finds a difference. Can be used with the qualifiers **/IMAGE** and **/PHYSICAL.**
/DELETE	Specifies that a BACKUP save or copy operation is to delete the selected input files from the input volume after all files have been processed.
/FAST	Processes the input specifier using a fast file scan to reduce processing time. The input specifier must be a Files–11 disk.
/FULL	Lists the file information produced by the command qualifier **/LIST** in the format provided by the DCL command **DIRECTORY/FULL.**
/IGNORE=option	Specifies that a BACKUP save or copy operation is to override certain restrictions. See Table 4–14 for list of options.
/IMAGE	Directs BACKUP to process an entire volume or volume set (all files).
/INCREMENTAL	Allows you to restore an incremental save set.
/[NO]INITIALIZE	Initializes an output disk volume, destroying all previous contents.
/INTERCHANGE	Directs BACKUP to process files in a manner suitable for data interchange with utilities and systems that are incompatible with the standard BACKUP format.
/JOURNAL[=file-spec]	Specifies that a BACKUP save operation is to create a BACKUP journal file, or append information to an existing BACKUP journal file.
/LIST[=file-spec]	Lists information about a BACKUP save set and its contents.
/[NO]LOG	Directs BACKUP to display the file spec of each file processed.
/PHYSICAL	Directs BACKUP to ignore any file structure on the input volume and to process the volume in terms of logical blocks.
/RECORD	Directs BACKUP to write the current date and time in the BACKUP date field of each file header record once a file is successfully saved or copied.
/[NO]TRUNCATE	Controls whether a copy or restore operation truncates a sequential output file at the end–of–file (EOF) when restoring it.
/VERIFY	Specifies that the contents of the output specifier be compared with the contents of the input specifier after a save, restore, or copy operation is completed.
/VOLUME=n	Indicates that a specific disk volume in a disk volume set is to be processed. (Only valid with the **/IMAGE** qualifier.)

Table 4–14: BACKUP /IGNORE Qualifier Options

Option	Description
INTERLOCK	Processes files that otherwise could not be processed because of file access conflicts. This option can be used to save or copy files currently open for writing, such as when doing a backup save operation on a disk actively being used. Note that no synchronization is made with the process writing the file, so the file data that is copied to the output specifier might be inconsistent with the input file, depending on the circumstances. When a file open for writing is processed, BACKUP issues the message: `%BACKUP-W-ACCONFLICT, "filename" is open for write by another user` The INTERLOCK option is especially useful if you have files that are open so much of the time that they might not otherwise be saved. The use of this option requires the SYSPRV privilege, a system UIC, or ownership of the volume.
LABEL_PROCESSING	Saves or copies the contents of files to the specified magnetic tape volume regardless of the information contained in the volume header record. BACKUP will not verify the volume label or expiration date before writing information to the tape volume.
NOBACKUP	Causes BACKUP to save files marked with the NOBACKUP flag by the DCL command **SET FILE/NOBACKUP.** If you do not specify this option, BACKUP saves only the file header record of files marked with the NOBACKUP flag.

Table 4–15: BACKUP Input File–Selection Qualifiers

Qualifier	Description
/BACKUP	Selects files according to the BACKUP date written in the file header record by the **BACKUP/RECORD** command.
/BEFORE=date	Selects files dated earlier than the specified date and time.
/BY_OWNER=uic	Selects files for processing according to the specified UIC.
/CONFIRM	Prompts for confirmation to process the input file.
/CREATED†	Selects files according to the value of the creation date field in each file header record.
/EXCLUDE=(file-spec,...)	Do not process files that otherwise meet the selection criteria.
/EXPIRED†	Selects files according to the value of the expiration date field in each file header record.
/MODIFIED†	Selects files according to the value of the modified date field in each file header record.
/SINCE=date	Selects files dated equal to, or later than, the specified date and time.

†Must also specify either the **/BEFORE** or **/SINCE** qualifiers.

Table 4–16: BACKUP Input Save Set Qualifiers

Qualifier	Description
/[NO]CRC	Specifies that the software Cyclic Redundancy Check (CRC) is to be performed.
/[NO]REWIND	Rewinds the input tape reel to the beginning–of–tape (BOT) marker before reading the input volume.
/SAVE_SET	Directs BACKUP to treat the input file as a BACKUP save set and not as an input file to be saved.
/SELECT=(file-spec,...)	Selects the specified files for processing.

Table 4–17: BACKUP Output File Qualifiers

Qualifier	Description
/BY_OWNER[=option]	Redefines the owner UIC for restored files. Options include: • Default — Sets the owner UIC to the user's current default UIC. • Original — Retains the owner UIC of the file being restored. • Parent — Sets the owner UIC to the owner UIC of the directory to which the file is being restored or copied. • [uic] — Sets the owner UIC to the UIC specified.
/NEW_VERSION†	Creates a new version of a file if a file with an identical specification already exists at the output location.
/OVERLAY†	Writes the input file over a file with an identical specification at the output location.
/REPLACE†	Replaces a file on the output specifier with an identically named file from the input specifier.

†If you do not specify **/NEW_VERSION, /OVERLAY,** or **/REPLACE,** and the version number of the file being restored is identical to that of an existing file, BACKUP reports an error and does not restore the file.

Table 4–18: BACKUP Output Save Set Qualifiers

Qualifier	Description
/BLOCK_SIZE=n	Specifies the output block size in bytes for data records in a BACKUP save set.
/BY_OWNER=uic	Specifies the owner UIC of the save set.
/COMMENT=string	Places a comment of up to 1024 characters in an output save set.
/[NO]CRC	Specifies that the software Cyclic Redundancy Check (CRC) is to be computed and stored in the data blocks of the output save set.
/DENSITY=n	Specifies the recording density of the output magnetic tape. Note that /**REWIND** is required with this qualifier.
/GROUP_SIZE=n	Defines the number of blocks BACKUP places in each redundancy group.
/LABEL=(string[,...])	Specifies the one– to six–character volume labels for magnetic tapes to which the save set is written.
/OWNER_UIC=uic	Specifies the owner UIC of the save set.
/PROTECTION=code	Specifies the protection to be applied to the save set.
/[NO]REWIND	Rewinds the output tape to the beginning–of–tape (BOT) marker and initializes the output tape. Specifying /**NOREWIND** causes the tape to wind forward to the logical end–of–tape (EOT) and to begin writing the save set there.
/SAVE_SET	Directs BACKUP to treat the output file as a BACKUP save set (on disk).
/TAPE_EXPIRATION[=date]	Writes the date on which the tape will expire to the volume header record. /**REWIND** must also be specified with this qualifier.

Table 4–19: Saving Files and Directories with On–Line BACKUP

Example	Comments
`$ BACKUP [BROWN]FILE.DAT -` `_$ MUA0:SAVE.BCK`	You must include the name of a save set as either the input–specifier or the output–specifier. To create a save set on a tape volume, mount the volume with the **MOUNT/FOREIGN** command first, so the BACKUP utility can format the tape.

`$ BACKUP [BROWN]*.*;*, -`
`_$ [SMITH]PRGM.FOR -`
`_$ MUA0:SAVEALL.BCK`
	You can save several files by including wildcards and/or listing the names (separated by commas). However, you can only specify one save set in the output–specifier.

`$ BACKUP [BROWN]FILE.DAT -` `_$ DRA1:[ARCHIVE]BROWN.BCK/SAVE_SET`	Include the **/SAVE_SET** qualifier after the save–set–name if the save set is on a disk volume. You can create save sets on any disk volume where you can create other files. You do not have to mount the volume as a foreign volume first.

List contents of a save set using the BACKUP utility.

```
$ BACKUP/LIST device-spec:save-set-name
```

Example 4–5: Listing the Contents of a Save Set

```
$ ALLOCATE MUA0
%DCL-I-ALLOC, _MUA0:  allocated
$ MOUNT/OVERRIDE=ID MUA0
%MOUNT-I-MOUNTED, SAVE mounted on _MUA0:
$ DIRECTORY MUA0:

Directory MUA0:[]

SAVE.BCK;1

Total of 1 file.
$DISMOUNT/NOUNLOAD MUA0
$ MOUNT/FOREIGN MUA0
%MOUNT-I-MOUNTED, SAVE mounted on _MUA0:
$ BACKUP/LIST MUA0:SAVE.BCK
Listing of save set

Save set:          SAVE.BCK
Written by:        BROWN
UIC:               [011,340]
Date:              4-JUL-1984 11:18:59.32
Command:           BACKUP WORK1:[BROWN]*.*;* MUA0:SAVE.BCK
Operating System:  VAX/VMS version X2M9
BACKUP version:    V4.0
CPU ID register:   0138700D
Node Name:         _SUPER::
Written on:        _MUA0:
Block size:        8192
Group size:        10
Buffer count:      3

[BROWN]A.LOG;6                  3 15-NOV-1983  11:43
[BROWN]FILE.DAT;1               1 17-OCT-1983  13:02
[BROWN]FILE.DAT;2               1 17-OCT-1983  14:00
[BROWN]JUNK.DAT;9               1 28-JUL-1983  09:00
[BROWN]MEMO.DAT;2               2  4-FEB-1984  10:49

Total of 5 files, 8 blocks
End of save set

$ DISMOUNT MUA0
$ DEALLOCATE MUA0:
```

When restoring save sets, enter the directory name (in acceptable format) as the output-specifier.

Table 4–20: Output-Specifier Formats Used to Restore Save Sets with On-Line BACKUP

Output-Specifier Format/Example†	Comments
`[*...]`	Restores the directory structure and files in the save set to the output volume in their original form.
`$ BACKUP MUA0:SAVE.BCK -` `_$ DRA1:[*...]`	Restores the files in the save set to DRA1:[BROWN] and DRA1:[BROWN.FILES], respectively. If these directories do not exist on the volume in DRA1, the BACKUP utility creates them.
`[...]`	Restores the files in the save set to the current default directory and subdirectories of the current default directory.

`$ SHOW DEFAULT`
`DRA1:[SMITH]`
`$ BACKUP MUA0:SAVE.BCK [...]`
	Restores the files from the [BROWN] directory to DRA1:[SMITH]. Restores the files from [BROWN.FILES] to [SMITH.FILES]. The utility creates subdirectories as needed.

`[directory...]`	Restores the files in the save set to the named directory and subdirectories of the named directory.
`$ BACKUP MUA0:SAVE.BCK -` `_$ DRA1:[JONES...]`	Restores the files from the [BROWN] directory to DRA1:[JONES]. Restores the files from [BROWN.FILES] directory to [JONES.FILES]. The utility creates the [JONES] and [JONES.FILES] directories if they do not exist.
`[directory]`	Restores the files in the save set to the named directory. If the save set contains files within subdirectories, the subdirectories are not created; instead, all files in all subdirectories will be restored to the single named directory.
`$ BACKUP MUA0:SAVE.BCK -` `_$ DRA1:[JONES]`	Restores the files from the save set to the [JONES] directory. All files in the directory and its subdirectories in the save set are restored as well. The utility does not create any subdirectories.
`[ ]` or no directory	Restores the files in the save set to the current default directory.

`$ SHOW DEFAULT`
`DRA1:[SMITH]`
`$ BACKUP MUA0:SAVE.BCK [ ]`
`$ BACKUP MUA0:SAVE.BCK *.*`
	Either command restores all the files in the save set to the DRA1:[SMITH] directory. The utility does not create subdirectories.

†You are restricted to these formats when you restore save sets. For all examples in this table, the save set contains the directory [BROWN], the subdirectory [BROWN.FILES], and several files in each.

Example 4–6: Restoring Specific Files from a Save Set

```
$ ALLOCATE MUA0
%DCL-I-ALLOC, _MUA0:  allocated
$ INITIALIZE MUA0 BROWN
$ MOUNT/FOREIGN MUA0
%MOUNT-I-MOUNTED, BROWN mounted on _MUA0:
$ SET DEFAULT DRA1:[BROWN]
$ BACKUP *.*;* MUA0:BROWN.BCK
$ BACKUP/REWIND/LIST MUA0:BROWN.BCK
Listing of save set

Save set:            BROWN.BCK
Written by:          BIERLY
UIC:                 [011,340]
Date:                 4-JUL-1984 11:08:51.55
Command:             BACKUP *.*;* MUA0:BROWN.BCK
Operating system:    VAX/VMS version X2M9
BACKUP version:      V4.0
CPU ID register:     0138700D
Node name:           _SUPER::
Written on:          _MUA0:
Block size:          8192
Group size:          10
Buffer count:        3

[BROWN]EXAMP5.COM;2                          1 17-NOV-1983  14:19
[BROWN]EXAMP5.COM;1                          1 17-NOV-1983  14:18
[BROWN]EXAMP5.DAT;4                          4 17-NOV-1983  14:24
[BROWN]EXAMP5.DAT;3                          4 17-NOV-1983  14:23
[BROWN]EXAMP5.DAT;2                          4 17-NOV-1983  14:19
[BROWN]EXAMP5.DAT;1                          4 17-NOV-1983  14:19
[BROWN]MAIL.MAI;1                           29 28-MAY-1984  17:06

Total of 7 files, 47 blocks
End of save set

$ DIRECTORY DRA2:[SMITH]
%DIRECT-W-NOFILES, no files found
$ BACKUP/REWIND MUA0:BROWN.BCK/SELECT=EXAMP5.COM DRA2:[SMITH]
$ DIR DRA2:[SMITH]

Directory DRA2:[SMITH]

EXAMP5.COM;2        EXAMP5.COM;1

Total of 2 files.
$
```

Improve performance of volume with image backup.

Table 4–21: Image Backups

Command/Example	Comments
Save a disk volume to a save set on tape	
`$ BACKUP/IMAGE device-name -` `_$ save-set-specifier`	Create a save set on tape so you can restore data if failures occur, or so you can temporarily store data on a less expensive medium. (Tape volumes must be mounted with the /**FOREIGN** qualifier first.)
`$ BACKUP/IMAGE DBA1: -` `_$ MUA0:MAR24.BCK`	
`$ BACKUP/IMAGE/RECORD DBA1: -` `_$ MUA0:FULL.BCK`	The /**RECORD** qualifier causes the utility to record the current date and time in the backup field of the file header of every file on DBA1 as it saves them. The BACKUP utility checks this field when it does an incremental backup (see Table 4–22).
Restore a disk volume from an image save set	
`$ BACKUP/IMAGE save-set-specifier -` `_$ device-name`	Restore a save set after a failure occurs and the data on the original disk volume is destroyed, or move data from temporary storage back to a disk volume.
`$ BACKUP/IMAGE MUA0:MAR24.BCK -` `_$ DBA1:`	Copy the contents of the MUA0:MAR24.BCK save set to the disk volume, DBA1. Since you include the /**IMAGE** qualifier, BACKUP creates the directories and subdirectories named in the save set to duplicate the original volume on the output volume.
Copy a disk volume to another disk and create a more efficient volume	
`$ BACKUP/IMAGE input-device -` `_$ output-device`	Make a copy of the volume loaded in the input device to the volume loaded in the output device.
`$ BACKUP/IMAGE DBA1: DBA2:`	Copy the contents of the volume in device DBA1 to the volume in device DBA2. In the new volume, the files are made contiguous, making I/O more efficient.

Perform incremental backups in between full image backups.

Table 4–22: Incremental Backup and Restore

Command/Example	Comments
Incremental Backup	

`$ BACKUP/RECORD -`
`_$ device-name/SINCE=BACKUP -`
`_$ save-set-specifier`
	Back up those files that were created or modified since the last save operation. BACKUP finds out whether a file was created or modified since the last backup by comparing the backup field in the file header with the modified field. The **/RECORD** qualifier causes the utility to write the current date and time to the file header of each file it saves.

`$ BACKUP/RECORD -`
`_$ DRA1:[*...]/SINCE=BACKUP -`
`_$ MUA0:JUN19F`
	Copies all files in all directories on the DRA1 volume which were created or done with the **/RECORD** qualifier. Files are copied to the save set MUA0:JUN19F. Note that you must mount the volume with the **/FOREIGN** qualifier before entering this command.

Incremental Restore	

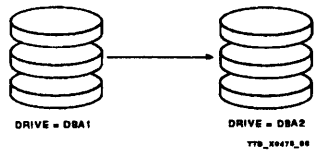
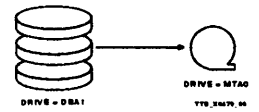
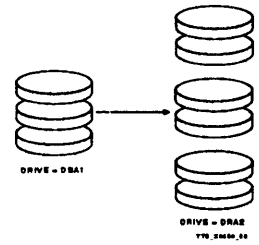
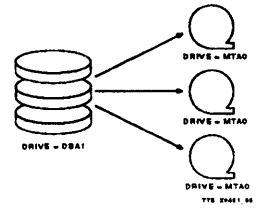
`$ BACKUP/INCREMENTAL -`
`_$ save-set-specifier`
`_$ output-device/BY_OWNER=ORIGINAL`
	Restore incremental backup save sets in reverse order (that is, restore the save set made on Friday, then Thursday's, then Wednesday's, etc.) after you restore the most recent full image save set. Include the **/INCREMENTAL** qualifier when restoring incremental save sets to enable the utility to restore the files correctly. Include the **/BY_OWNER=ORIGINAL** qualifier to ensure that each file is given the UIC of its original owner, rather than the UIC of the process entering the BACKUP command.

`$ BACKUP/IMAGE -`
`_$ MUA0:WEDFULL.BCK -`
`_$ DRA1:`
 	
`$ BACKUP/INCREMENTAL -`
`_$ MUA0:FRIINC.BCK -`
`_$ DRA1:[*...]/BY_OWNER=ORIGINAL`
 	
`$BACKUP/INCREMENTAL -`
`_$ MUA0:THUINC.BCK -`
`_$ DRA1:[*...]/BY_OWNER=ORIGINAL`
	Restores the full image backup done on Wednesday to the device DRA1. The incremental save sets update the volume until it is in the same state it was in at the time of the last incremental backup (done on Friday).

Operator Action During Multivolume Disk or Tape Set Backups

- VMS automatically intervenes to create multivolume set
- Operator receiving request to load next volume in set should:
  - — Unload current volume
  - — Load next volume
  - — Respond to request with **REPLY/INITIALIZE** or **REPLY/BLANK_TAPE** command

Figure 4–4: Output Media Used for Backups

OPERATION/COMMAND	COMMENTS
Backup Disk to Disk - $ BACKUP/IMAGE DBA1 DBA2  	Copy all of the information from the input disk to the output disk. If the output disk is large enough to hold all of the information from the input disk, you can store the input disk and use the output disk. Performance improves because the information on the output disk is compressed and is faster to access. Note that if the input disk was bootable, the output disk will also be bootable.
Backup Disk to Tape - BACKUP/IMAGE DBA1 MUA0:FULL.BCK/ SAVE_SET  	Copy all of the information from the disk to a save set on the tape. To improve performance, load a scratch disk and restore the information to it from the tape. Then use the newly created disk and store the previous one.
Backup Disk to Multiple Disks - $ BACKUP/IMAGE DBA1: - _$ DRA2:SAVE.BCK  	Create a save set on the output volumes. When the first output volume is full, the system notifies the operator by sending a message to the operator console. The operator removes the volume and loads the next one. The operator is prompted to load each successive volume until the backup is complete. Improve performance with the same method used in disk to tape backups (see previous section of figure). These output disks are called a multivolume disk set. To restore them, you must begin with the first volume.
Backup Disk to Multiple Tapes - $ BACKUP/IMAGE DBA1: - _$ MUA0:SAVE.BCK  	Create a save set. When the first volume is full, the system notifies the operator to load the next volume. This continues until the backup is complete. Improve performance using the same method described for disk to tape backups (See previous section of figure). The output tapes are called a multivolume tape set. To restore them, you must begin with the first volume.

TTB_X0971_88A

Example 4–7: Creating a Multivolume Tape Set on One Drive

```
$ ALLOCATE MT: TAPE
%DCL-I-ALLOC, _MUA0: allocated
$ INITIALIZE TAPE BACKUP1
$ MOUNT TAPE BACKUP1
%MOUNT-I-MOUNTED, BACKUP1 mounted on _MUA0:
$ SHOW DEVICE/FULL TAPE

Magtape MUA0:, device type TK50, is online, allocated, mounted,
        error logging enabled.

    Error count                    0     Operations completed              31
    Owner process          "_TTA1:"      Owner UIC                    [11,340]
    Owner process ID       0000009D      Dev Prot S:RWED,O:RWED,G:RWED.W:RWED
    Reference count                1     Default buffer size             2048

    Volume label           "BACKUP1"     Relative volume no.                0
    Record size                    0     Transactin count                   1
    Mount status             Process     Mount count                        1
    ACP process name       "MUA0CACP"
    Density                     1600     Format                      Normal-11

  Volume status:  beginning-of-tape, odd parity.

$ COPY SYS$SYSTEM:SYSDUMP.DMP TAPE
$ COPY/EXCLUDE=*.DAT   SYS$LIBRARY:*.*  TAPE

$
%%%%%%%%%%  OPCOM  4-JUL-1984  14:35:20.47  %%%%%%%%%%
Request 1, from user BIERLY
MOUNT new relative volume 2 (BACKUP2) on_MUA0:

$ SET PROCESS/PRIVILEGE=OPER
$ REPLY/INITIALIZE=1 "BACKUP2"
$
BACKUP2
14:41:16.86, request 1 was completed by operator _OPA0:

$

$ SHOW DEVICE/FULL TAPE

Magtape MUA0:, device type TK50, is online, allocated, mounted,
        error logging enabled.

    Error count                    3  Operations completed           14110
    Owner process          "_TTA1:"  Owner UIC                    [11,340]
    Owner process ID       0000009D  Dev Prot  S:RWED,O:RWED,G:RWED.W:RWED
    Reference count                1  Default buffer size             2048

    Volume label           "BACKUP2"  Relative volume no.                2
    Record size                    0  Transaction count                  1
    Mount status             Process  Mount count                        1
    ACP process name       "MUA0CACP"
    Density                     1600  Format                      Normal-11

  Volume status:  end-of-file, odd parity.

$ DISMOUNT TAPE
$ DEALLOCATE TAPE
$ DEASSIGN TAPE
$
```

Creating Multivolume Tape Sets on More than One Drive

- Initialize the first volume, assigning it some volume label
- List names of drives in **MOUNT** command and include **/INITIALIZATION=CONTINUATION** qualifier
- System will then:
  - — Initialize and mount subsequent volumes on drives you specified in the **MOUNT** command if you have loaded the volumes on the drives
  - — Generate label names automatically based on user–specified label and relative volume number
  - — Subsequent labels are created by using the first four characters of the specified label

    Underscore ("_") used to pad labels specified with fewer than four characters (see Table 4–23)

Table 4–23: Generating Labels Automatically

Specified Label (First Volume)	Corresponding Generated Label (Second and Subsequent Volumes)
MAIN	MAIN02, MAIN03,... MAIN99
T15	T15_02, T15_03,... T15_99
DAN	DAN_02, DAN_03,... DAN_99
BACKUP	BACK02, BACK03,... BACK99

BACKUP Tape Label Processing

- BACKUP checks the tape label for expiration before attempting to write to it
  - — BACKUP will not allow writing to a tape that has not yet expired
  - — Use the /**IGNORE=LABEL_PROCESSING** qualifier to override this situation
- Always set an appropriate expiration date when creating a BACKUP save set on tape
- Be sure to specify the /**REWIND** qualifier when setting the expiration date

```
$ BACKUP /RECORD/SINCE=YESTERDAY  DISK$USER:[*...] -
_$ MUA0:MAY0688.BCK /REWIND/TAPE_EXPIRATION=13-MAY-1988
```

BACKUP Journal Files

- BACKUP can create a *journal file* when creating a save set
  - — Contains BACKUP operations and file specifications
  - — Provides an efficient way to retain and list "directories" of multivolume tape save sets
  - — Useful for quickly determining the location of files that must be restored
- Use the **/JOURNAL=file-spec** qualifier to create a journal file

```
$ BACKUP /RECORD/JOURNAL=MAY0688.BJL -
_$ DISK$USER:[PAYROLL...] -
_$ MUA1:MAY0688.BCK /REWIND/TAPE_EXPIRATION=13-MAY-1988
```

- Default file specification for journal files: SYS$DISK:[]BACKUP.BJL
- Journal files are written in a binary format
  - — Use **/LIST/JOURNAL=file-spec** to list contents
  - — Do not specify an input- or output-specifier when listing a journal file

```
$ BACKUP /LIST/JOURNAL=MAY0688.BJL
```

Table 4–24: Examples of Common BACKUP Operations

Operation	Save Command	Restore Command
Full backup operations		
Physical backup	`$ BACKUP/PHYSICAL DUA0: DUA1:`	

Image, disk to tape	`$ BACKUP/IMAGE/RECORD -`
	`_$ DUA0: -`
	`_$ MUA0:01JAN1990.BCK-`
	`_$ /INITIALIZE/REWIND/BUFFER=5 -`
	`_$ /TAPE_EXPIRATION=8-JAN-1990 -`
	`_$ /DENSITY=1600/BLOCK=32768`
		`$ BACKUP/IMAGE -`
		`_$ MUA0:01JAN1990.BCK/BUFFER=5 -`
		`_$ DUA0:`

Image, disk to disk	`$ BACKUP/IMAGE/RECORD -`
	`_$ DUA0: -`
	`_$ DUA1:`	

Image, disk to save set in disk directory	`$ BACKUP/IMAGE/RECORD -`
	`_$ DUA0: -`
	`_$ DUA1:[BACKUP]01JAN1990.BCK -`
	`_$ /SAVE_SET`
		`$ BACKUP/IMAGE -`
		`_$ DUA1:[BACKUP]01JAN1990.BCK -`
		`_$ /SAVE_SET -`
		`_$ DUA0:`

Image, disk to save set on multiple disks	`$ BACKUP/IMAGE/RECORD -`
	`_$ DUA0: -`
	`_$ DUA1:01JAN1990.BCK/SAVE_SET`
		`$ BACKUP/IMAGE -`
		`_$ DUA1:01JAN1990.BCK/SAVE_SET -`
		`_$ DUA0:`

Incremental backup operations		

Disk to tape	`$ BACKUP/SINCE=BACKUP/RECORD -`
	`_$ DUA0:[000000...]*.*.* -`
	`_$ MUA0:01JAN1990.BCK -`
	`_$ /INITIALIZE/REWIND/BUFFER=5 -`
	`_$ /TAPE_EXPIRATION=8-JAN-1990 -`
	`_$ /BLOCK=32768/DENSITY=1600`
		`$ BACKUP/INCREMENTAL -`
		`_$ MUA0:01JAN1990.BCK/BUFFER=5 -`
		`_$ DUA0:`

Disk to save set in disk directory	`$ BACKUP/SINCE=BACKUP/RECORD -`
	`_$ DUA0:[000000...]*.*.* -`
	`_$ DUA1:[BACKUP]01JAN1990.BCK -`
	`_$ /SAVE_SET`
		`$ BACKUP/INCREMENTAL -`
		`_$ DUA1:[BACKUP]01JAN1990.BCK -`
		`_$ /SAVE_SET -`
		`_$ DUA0:`

Disk to save set on multiple disks	`$ BACKUP/SINCE=BACKUP/RECORD -`
	`_$ DUA0:[000000...]*.*.* -`
	`_$ DUA1:01JAN1990.BCK/SAVE_SET`
		`$ BACKUP/INCREMENTAL -`
		`_$ DUA1:01JAN1990.BCK/SAVE_SET -`
		`_$ DUA0:`

Table 4–24: Examples of Common BACKUP Operations Cont.

Operation	Save Command	Restore Command
Partial backup operations		

Disk to tape	`$ BACKUP/RECORD -`
	`_$ DUA0:[SMITH...]*.*.* -`
	`_$ MUA0:SMITH.BCK -`
	`_$ /INITIALIZE/REWIND/BUFFER=5 -`
	`_$ /TAPE_EXPIRATION=1-JAN-1993 -`
	`_$ /DENSITY=1600/BLOCK=32768`
		`$ BACKUP -`
		`_$ MUA0:SMITH.BCK/BUFFER=5 -`
		`_$ DUA0:`

Disk to disk	`$ BACKUP -`
	`_$ DUA0:[SMITH...]*.*.* -`
	`_$ DUA1:`	

Disk to save set in disk directory	`$ BACKUP/RECORD -`
	`_$ DUA0:[SMITH...]*.*.* -`
	`_$ DUA1:[BACKUP]SMITH.BCK -`
	`_$ /SAVE_SET`
		`$ BACKUP -`
		`_$ DUA1:[BACKUP]SMITH.BCK -`
		`_$ /SAVE_SET -`
		`_$ DUA0:`

Disk to save set on multiple disk	`$ BACKUP/RECORD -`
	`_$ DUA0:[SMITH...]*.*.* -`
	`_$ DUA1:SMITH.BCK/SAVE_SET`
		`$ BACKUP -`
		`_$ DUA1:SMITH.BCK/SAVE_SET -`
		`_$ DUA0:`

Standalone BACKUP

- Use standalone BACKUP to back up system disk, since system disk is constantly being used while VMS is running
- To back up system disk
  - — Shut down VMS
  - — Boot standalone BACKUP kit to copy system disk
  - — Enter **BACKUP/IMAGE** command
  - — Reload console media and boot the system when backup is complete

VERIFY UTILITY

- Checks readability and validity of file structure
- Modes of operation
  - — Error reporting with no repairs
  - — Error reporting with repairs
  - — Error reporting with user–controlled selective repairs
- To run on data disk, mount it privately
- To run on system disk, first force all users to log out
- DCL command format:

```
$ ANALYZE/DISK_STRUCTURE device-name [/qualifier...]
```

Table 4–25: Using the VERIFY Utility

Function	Command/Example	Comments
Running VERIFY to find errors in the file structure	`$ ANALYZE/DISK_STRUCTURE device-name` `$ ANALYZE/DISK_STRUCTURE MYDISK`	Using no parameter qualifier invokes VERIFY in the error–reporting mode.

Running VERIFY to report and repair errors	`$ ANALYZE/DISK_STRUCTURE -`
	`_$ device-name/REPAIR`
	`$ ANALYZE/DISK_STRUCTURE -`
	`_$ MYDISK/REPAIR`
		The **/REPAIR** qualifier causes automatic repair of all errors located in the file structure.

Running VERIFY to report errors and allow the user to decide what gets repaired	`$ ANALYZE/DISK_STRUCTURE -`
	`_$ device-name/REPAIR/CONFIRM`
	`$ ANALYZE/DISK_STRUCTURE -`
	`_$ MYDISK/REPAIR/CONFIRM`
		The qualifiers **/REPAIR** and **/CONFIRM** cause the utility to display errors, ask the user if the error should be corrected, and carry out the user's response.

Example 4–8: Using the VERIFY Utility

```
$ MOUNT DRA2 PROGRAM_DISK MYDISK
%MOUNT-I-MOUNTED, PROGRAM_DISK mounted on _DRA2:
$ ANALYZE/DISK_STRUCTURE MYDISK
%VERIFY-I-OPENQUOTA, error opening QUOTA.SYS
-SYSTEM-W-NOSUCHFILE, no such file
%VERIFY-I-LOSTHEADER, file (11,2,1) UTLIST.DAT;1
        not found in a directory
%VERIFY-I-LOSTHEADER, file (12,2,1) UTLIST.COM;1
        not found in a directory
$
$ ANALYZE/DISK_STRUCTURE/REPAIR/CONFIRM MYDISK
%VERIFY-I-OPENQUOTA, error opening QUOTA.SYS
-SYSTEM-W-NOSUCHFILE, no such file
%VERIFY-I-LOSTHEADER, file (11,2,1) UTLIST.DAT;1
        not found in a directory
Repair this error (D to delete)? (D, Y or N):  Y
%VERIFY-I-LOSTHEADER, file (12,2,1) UTLIST.COM;1
        not found in a directory
Repair this error (D to delete)? (D, Y or N):  Y
$
$ DIRECTORY MYDISK:[SYSLOST]

Directory MYDISK:[SYSLOST]

UTLIST.COM;1        UTLTST.DAT;1

Total of 2 files.
$
```

TRANSFERRING FILES BETWEEN VAX AND PDP-11 SYSTEMS

- Utilities that can be used
  - — EXCHANGE
  - — RMSBCK
  - — RMSRST

Table 4–26: Transferring Files Between VAX and PDP-11 Systems

System	Operation	Comments
RSX	Preparing a volume on VMS for use on RSX	The only special action needed to prepare a volume for use on RSX is to initialize the volume using the **/STRUCTURE_LEVEL=1** qualifier. See the description of the **INITIALIZE** command in the *VMS DCL Dictionary*.
	Preparing a volume on RSX for use on VMS	VMS understands the RSX file structure. You can create RSX volumes and use them on a VMS operating system without taking special action.
RSTS	Transferring RMS files between systems	Use RMSBCK to create a tape volume containing the files you want to transfer. Transfer the files from the tape to the target system using RMSRST. See the the *RMS–11 User's Guide* for more information.
	Transferring non–RMS files between systems	Use the EXCHANGE utility to transfer the files from the transportable medium to the target system. See the *VMS Exchange Utility Manual* for more information.
RT–11	Transferring files between RT–11 and VMS	Use the EXCHANGE utility. See the *VMS Exchange Utility Manual*.

SUMMARY

- A system manager should be able to prepare and maintain volumes
- Volume preparation includes
  - — Locating bad blocks (**ANALYZE/MEDIA**)
  - — Initializing the volume (**INITIALIZE**)
  - — Mounting the volume (**MOUNT**)
  - — Defining access to the volume (**INITIALIZE** and **MOUNT**)
- Obtain volume information with **SHOW DEVICE**
- Modify volume information with **SET VOLUME**
- Create volume sets when files or directories are too large to fit on a single volume
- Use BACKUP to save
  - — An entire volume
  - — Only modified files since last backup
  - — Specific files
- Use BACKUP to restore
  - — An entire saveset
  - — Specific files
- Use standalone BACKUP to save/restore the system volume
- Use the VERIFY utility (**ANALYZE/DISK_STRUCTURE**) to check integrity of disk volumes
  - — Can choose to locate errors and optionally repair them
- Utilities to transfer files between VAX and PDP–11 systems
  - — EXCHANGE
  - — RMSBCK
  - — RMSRST

MODULE 5
CUSTOMIZING THE SYSTEM

INTRODUCTION

Essentially, every VMS system is a customized system. Not only does the choice of hardware vary between installations, this hardware can be configured many ways. **MODULE 1**, discussed many of these hardware options, along with a number of software choices for VMS systems.

From these choices, you should configure your system and install software to meet the needs of your users. However, standard hardware and software may not entirely satisfy users' needs. Typically, you must also customize the software on your system. You can customize the software by:

- Creating user accounts
- Establishing disk quotas as needed
- Creating user directories
- Initializing queues
- Mounting disk volumes
- Modifying system parameters
- Changing passwords
- Setting up files and directories for standalone BACKUP and for copying the console volume

Previous modules covered many of the skills you use to customize the VMS environment, such as creating user accounts and queues. This module discusses some other customizing skills, such as creating site-specific command procedures and modifying the DCL environment for a user. As you customize the system using all of the skills discussed in this course, you can improve its performance and efficiency. **MODULE 9**, discusses performance improvement in greater detail.

OBJECTIVES

To customize a system, the system manager should be able to modify the site–specific startup files and/or enter DCL commands interactively to perform tasks discussed in earlier modules of this course (such as creating batch and print queues) as well as:

- Identify the functions of the different system startup files
- Set device characteristics
- Mount site–specific volumes
- Create system logical names
- Install images
- Define and start queues
- Create reports about the last system failure
- Start local DECnet
- Announce system availability

RESOURCES

1. *VMS System Generation Utility Manual*
2. *VMS Install Utility Manual*
3. *Guide to Setting Up a VMS System*
4. *VMS Installation and Operations* for your particular VAX system

USER ENVIRONMENT

- Principal methods of defining default process environment
  - — Setting system parameters
  - — Issuing DCL commands
- Process default environment is established in two stages
  - — At system startup

    System-wide default is established
  - — When a process is created

    Individual process environment is customized
- Process environment can also be customized interactively

Table 5–1: Files Controlling the VMS Environment

File	Information in File Used to Control VMS Environment	When Information in File is Automatically Used
SYS$SYSTEM:VAXVMSSYS.PAR	System parameters	System startup
SYS$SYSTEM:STARTUP.COM	DCL commands	System startup
SYS$MANAGER:SYPAGSWPFILES.COM	DCL commands and possibly SYS-GEN commands	System startup
SYS$MANAGER:SYCONFIG.COM	DCL commands	System startup
SYS$MANAGER:SYLOGICALS.COM	DCL commands	System startup
SYS$MANAGER:SYSTARTUP_V5.COM	DCL commands and possibly INSTALL commands	System startup
SYS$MANAGER:SYLOGIN.COM	DCL commands	Interactive or batch process creation
SYS$LOGIN:LOGIN.COM	DCL commands	Interactive or batch process creation

SYSTEM STARTUP FILES

There are five startup command procedures distributed as part of VMS:

1. SYS$SYSTEM:STARTUP.COM

   Site-independent
   Commands supplied by DIGITAL
   Do not modify

2. SYS$MANAGER:SYPAGSWPFILES.COM

   Site-specific
   Initially empty
   Used to install page and swap files on any disk

3. SYS$MANAGER:SYCONFIG.COM

   Site-specific
   Initially empty
   Connects various devices to system and loads their I/O drivers
   If left empty, system automatically configures all devices

4. SYS$MANAGER:SYLOGICALS.COM

   Site-specific
   Template file supplied by DIGITAL
   Used to define system-wide logical names

5. SYS$MANAGER:SYSTARTUP_V5.COM

   Site-specific
   Template file supplied by DIGITAL
   General location for site-specific customization commands not addressed by other site-specific startup command files
   Overrides commands in STARTUP.COM

SYS$SYSTEM:STARTUP.COM

- Executes immediately after VMS is booted
- Uses a series of component files that perform:
  - — Defines system-wide logical names for standard system software
  - — Starts up system processes, such as:

    JOB_CONTROL
    OPCOM
    ERRFMT

  - — Connects all standard devices
  - — Installs known images (standard and layered software products)
  - — Calls site-specific startup command procedures in this order:

    1. SYS$MANAGER:SYPAGSWPFILES.COM
    2. SYS$MANAGER:SYCONFIG.COM
    3. SYS$MANAGER:SYLOGICALS.COM
    4. SYS$MANAGER:SYSTARTUP_V5.COM

- Component files are located according to the logical name SYS$STARTUP
  - — Actually a search list that includes:

    SYS$SYSROOT:[SYSMGR] (SYS$MANAGER)
    SYS$SYSROOT:[SYS$STARTUP]

- STARTUP.COM starts up the system in *phases*
  - — Four basic phases
  - — Three data files involved in the phased startup (all located in SYS$STARTUP):

    VMS$PHASES.DAT (**do not modify**)
    VMS$VMS.DAT (**do not modify**)
    VMS$LAYERED.DAT (managed through use of SYSMAN utility)

SYS$MANAGER:SYPAGSWPFILES.COM

- Invoked by SYS$SYSTEM:STARTUP.COM
- Used to install page and swap files on disks other than the system disk
- Before invoking SYPAGSWPFILES.COM, the system activates the following files if they exist in SYS$SYSTEM:
  - PAGEFILE.SYS
  - SWAPFILE.SYS
  - SYSDUMP.DMP
- STARTUP.COM then invokes SYPAGSWPFILES.COM
- Place whatever commands are needed to install page and swap files
  - **INITIALIZE**

    Needed to initialize a new disk volume (rare situation)
  - **MOUNT**

    Disks containing secondary page and swap files are not yet mounted
  - SYSGEN Commands

    To create and/or install secondary page and swap files

SYS$MANAGER:SYCONFIG.COM

- Invoked by SYS$SYSTEM:STARTUP.COM
- Used to connect special devices and load their I/O drivers
  - — Only necessary for nonstandard devices or unusual device settings
- SYSGEN commands are typically placed in this file
- You can optionally place **MOUNT** commands in this file
  - — Most sites mount remaining disks in SYSTARTUP_V5.COM (described later)
- When SYCONFIG.COM completes, control is returned to STARTUP.COM
  - — STARTUP.COM automatically connects all remaining devices and loads their I/O drivers
  - — Connecting and loading is accomplished by SYSGEN **AUTOCONFIGURE ALL** command

SYS$MANAGER:SYLOGICALS.COM

- Invoked by SYS$SYSTEM:STARTUP.COM
- Used to create system–wide logical names
- Define system components as executive mode logical names
- To create system logical names:

```
$ ASSIGN/SYSTEM
$ DEFINE/SYSTEM
```

- To delete system logical names:

```
$ DEASSIGN/SYSTEM
```

- Typically use the /**NOLOG** qualifier when defining logical names during system startup

Executive Mode Logical Name Requirements

- Logical names for system components and files must be executive mode
- Examples of components and files requiring executive mode logical names:
  - — Public disks and directories
  - — SYSUAF.DAT
  - — RIGHTSLIST.DAT
  - — VMSMAIL.DAT
  - — NETPROXY.DAT
- To define an executive mode logical name:

```
$ DEFINE/SYSTEM/EXECUTIVE  logical-name  equivalence-name
```

Table 5–2: Assigning System Logical Names

Operation	Command Format/Example (Requires SYSNAM privilege)

Create or replace a system logical name	`$ ASSIGN/SYSTEM eqv-name log-name`
	`$ DEFINE/SYSTEM log-name eqv-name`
	`$ ASSIGN/SYSTEM SYS$SYSTEM:NOTICE.TXT NOTICE`

Delete a system logical name	`$ DEASSIGN/SYSTEM log-name` `$ DEASSIGN NOTICE`

Table 5–3: Some Standard Logical Names to Define in SYLOGICALS.COM

Name	Definition	Function
SYS$SYLOGIN	Name of system manager's login command procedure	The system executes this procedure when it creates a process.
SYS$ANNOUNCE	Line of text or name of file containing text	The system displays this line or the contents of the file when the user presses the RETURN key to log in.
SYS$WELCOME	Line of text or name of file containing text	The system displays this line or the contents of the file after a user successfully logs in (by default, "`Welcome to VMS V5.0`")

Example 5–1: Assigning Site–Specific System Logical Names (SYLOGICALS.COM)

```
$!
$!  Define site-specific logical names
$!
$ DEFINE /SYSTEM /EXEC  SYS$SYLOGIN       SYS$MANAGER:SYLOGIN
$ DEFINE /SYSTEM /EXEC  SYS$TOOLS         DISK$USER:[TOOLS]
$ DEFINE /SYSTEM /EXEC  SYS$ANNOUNCE      "This is the MENTOR system"
$ DEFINE /SYSTEM /EXEC  SYS$WELCOME       "@SYS$MANAGER:WELCOME.TXT"
$!
```

SYS$MANAGER:SYSTARTUP_V5.COM

- The last site-specific startup command procedure invoked by STARTUP.COM
- Functions performed by SYSTARTUP_V5.COM
  - — Mounts all volumes except system disk
  - — Sets device characteristics
  - — Initializes and starts batch and print queues
  - — Installs known images
  - — Starts up DECnet software (if it exists)
  - — Analyzes most recent system failure
  - — Purges old operator log files
  - — Starts up the LAT network (if it exists)
  - — Defines the maximum number of interactive users
  - — Announces system is up
  - — Allows users to log in
- Use separate command procedures for major functions
  - — Keeps SYSTARTUP_V5.COM small and manageable
  - — Allows clean, separate execution of functions after system has been started

    Sometimes needed when certain failures occur and the function needs to be reactivated (example: print and batch queues)

Example 5–2: SYSTARTUP_V5.COM Command Procedure

```
$ SET NOON
$ SET NOCONTROL_Y
$ ! ++
$ ! SYS$MANAGER:SYSTARTUP_V5.COM
$ !
$ ! This is a sample site-specific system startup command file
$ !--
$ !
$ ! Create logical name for supporting command procedures
$ DEFINE/NOLOG  STARTUP_PROCS  SYS$SYSROOT:[SYSMGR.STARTUP]
$ !
$ ! Mount site-specific volumes
$ @STARTUP_PROCS:MOUNTDSK.COM
$ !
$ ! Set device characteristics
$ @STARTUP_PROCS:DEVICES.COM
$ !
$ ! Start the queue manager
$ START/QUEUE/MANAGER
$ !
$ ! Define and start print queues
$ @STARTUP_PROCS:START_PRNT_QUEUE.COM
$ !
$ ! Define and start batch queues
$ @STARTUP_PROCS:START_BATCH_QUEUE.COM
$ !
$ ! Install known images
$ @STARTUP_PROCS:INSTALL.COM
$ !
$ ! Start DECnet
$ @SYS$MANAGER:STARTNET.COM
$ !
$ ! Create reports about the last system failure
$ @STARTUP_PROCS:REPORT_FAILURE.COM
$ !
$ ! Purge old versions of system log files
$ PURGE SYS$MANAGER:*.LOG/KEEP=3
$ !
$ ! Set the maximum number of interactive users
$ STARTUP$INTERACTIVE_LOGINS == 40
$ !
$ ! Announce availability of the system to all terminals
$ SUBMIT STARTUP_PROCS:START_ANNOUNCE.COM
$ !
$ ! End of SYS$MANAGER:STARTUP_V5.COM
$ !
$ EXIT
```

Mounting Site-Specific Volumes

- Mount volumes in site-specific startup command procedure

```
$ MOUNT/SYSTEM  device-spec  volume-label  logical-name
```

- Do not mount volumes in site-independent startup command procedure (not even the system disk)
- By default in SYSTARTUP_V5.COM, **MOUNT** implies **MOUNT/NOASSIST**
- **/SYSTEM** qualifier:
  - — Makes volume public (available to all users)
  - — Stores specified logical name in system logical name table
  - — Assigns concealed and terminal attributes to logical name
- Specifying a logical name with each mounted disk improves the user interface for:
  - — Issuing commands
  - — Displaying file specifications in listings
- If no logical name is specified, a default logical name is created
  - — Format: DISK$volume-label
- **WAIT** statement needed prior to mounting the first DSA disk
  - — Wait time is controller dependent
  - — Refer to *VMS I/O User's Reference Manual: Part I* for details on time value

Example 5–3: Mounting Site–Specific Volumes (MOUNTDSK.COM)

```
$!
$! Mount public disk volumes
$!
$  MOUNT /SYSTEM  DBA2:  DATA13   DATADISK
$  MOUNT /SYSTEM  DBA3:  USER
```

Setting Device Characteristics

- Most common situation is setting terminal characteristics
- SYSGEN **AUTOCONFIGURE ALL** command in STARTUP.COM determines the number of terminal lines the system can support
- Other commands in SYSTARTUP_V5.COM set physical characteristics of attached terminals
  - — Best done with a separate command procedure called by SYSTARTUP_V5.COM
- Can modify permanent characteristics interactively, but effective only until next startup
- Can also set physical characteristics for such devices as printers, tape drives, and card readers
  - — Many sites prefer to set printer and card reader characteristics in command procedure called to establish print and batch queues

    Makes it easier to match queues with physical devices

Table 5–4: Setting Permanent Characteristics of Terminals

Characteristic	SET TERMINAL/PERMANENT Command Qualifiers and Examples†	Comments

Line speed, for direct lines	`/SPEED/NOMODEM`
	
	`$ SET TERMINAL/PERMANENT`
	`-`
	`_$ /SPEED=9600/NOMODEM TTF5`
		By default, all lines have the speed specified by system parameter TTY_SPEED.

Line speed, for lines connected through modems	`/AUTOBAUD/MODEM`
	
	`$ SET TERMINAL/PERMANENT`
	`-`
	`_$ /AUTOBAUD/MODEM TTA1`
		VMS does not send broadcast messages to autobaud lines unless the terminal is logged in.

Terminal type, for direct lines	`/DEVICE_TYPE=type`
	
	`$ SET TERMINAL/PERMANENT`
	`-`
	`$_ /DEVICE_TYPE=VT100 TTC2`
		Among the recognized device types are: LA36, LA120, VT52, and VT100. This one qualifier is a substitute for many individual qualifiers, each specifying only one characteristic. To list the available device types, enter the command **HELP SET TERMINAL/DEVICE_TYPE**. Many characteristics of the terminal are modified when you specify a device type, saving you from having to enter each one separately.

†Requires LOG_IO or PHY_IO privilege.

Table 5–5: Establishing Ownership and Protection of Terminals and Other Nonshareable Devices

Operation	Command Examples (Requires OPER Privilege)	Comments

Establishing protection ownership	`$ SET PROTECTION=code/DEVICE -`
	`_$ /OWNER_UIC=uic device`
	
	`$ SET PROTECTION=(S:R,O:R,G,W)`
	`_$ /DEVICE/OWNER=[JONES] TTA3`
		If you give Read access to a category of user, users in that category can allocate the terminal from a program. Other users can log in to the terminal.

Allowing all users access to a device	`$ SET PROTECTION/DEVICE device` `$ SET PROTECTION/DEVICE TTA3`	If you omit a user category, users in that category are allowed all types of access to the device.

- Control device ownership and protection

```
$ SET PROTECTION/DEVICE
```

- Commonly used protection on terminals
  - — Owner: [1,4]; protection: (S:R,O:R)
  - — Prevents nonprivileged users from allocating terminals
  - — Does not prevent nonprivileged users from logging in
  - — Default ownership and protection established by system parameters

Example 5–4: Setting Device Characteristics (TERMINALS.COM)

```
$ SET TERMINAL/PERMANENT/DEVICE=LA120/NOLOWER              TTC0:   !HARDCOPY
$ SET TERMINAL/PERMANENT/NOMODEM/SPEED=9600/DEVICE=VT100   TTA0:   !M.BROWN
$ SET TERMINAL/PERMANENT/NOMODEM/SPEED=1200/DEVICE=LA120   TTA1:   !R. JONES
$ SET TERMINAL/PERMANENT/AUTOBAUD/MODEM                    TTA2:   !DIAL-UP
```

Initializing and Starting Queues

- Establish all queues (print and batch) in one or more separate command procedures
  - — Invoked by SYSTARTUP_V5.COM
  - — Use commands and techniques described in MODULE 3.

Installing Known Images

- Reasons for installing
  - — Reduces time required for image activation
  - — Enables image to execute with privileges that the process running the image lacks
  - — Promotes sharing of physical memory
  - — Can add user-written system service
- Use the INSTALL utility
- Can use INSTALL to "undo" what STARTUP.COM has installed
  - — Remove some images
  - — Reinstall some images with different privileges or attributes

NOTE

Do not confuse the two different VMS uses of the word "install."

1. **Install known images (discussed here)**
2. **Install VMS operating system or optional software (discussed in MODULE 7)**

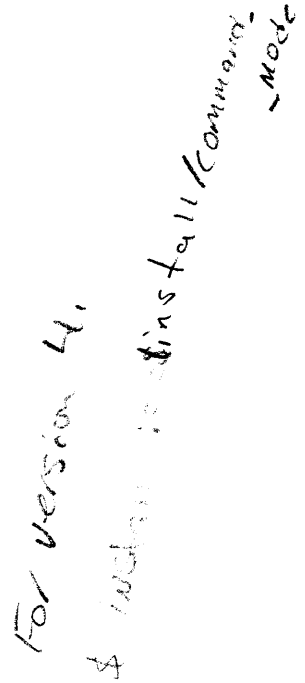


Table 5–6: Functions of the INSTALL Utility

Operation	Command Format and Example(s)	Comments
Installing a known image and specifying optional characteristics	`INSTALL> CREATE[/qualifier(s)] file-spec` `INSTALL> CREATE/OPEN/SHARED EDT`	Qualifiers used to specify optional characteristics are shown in Table 5–7.
Installing a new version of a known image with the same characteristics as the existing version	`INSTALL> REPLACE file-spec` `INSTALL> REPLACE EDT`	Note that INSTALL uses the following file specifier defaults: Device and directory: SYS$SYSTEM File type: .EXE Version number: highest Specifying a version number will produce unpredictable results when you use the known image. Therefore, never specify a version number in the file specification.
Installing a new version of a known image with different characteristics than the existing version	`INSTALL> REPLACE[/qualifier(s)] file-spec` `INSTALL> REPLACE/OPEN/SHARED/HEADER EDT`	
Removing a known image	`INSTALL> DELETE file-spec` `INSTALL> DELETE EDT`	

Displaying information about a known image or about INSTALL	`INSTALL> LIST[/qualifier(s)] [file-spec]`
	`INSTALL> LIST/FULL`
	`INSTALL> LIST/GLOBAL`
	`INSTALL> HELP LIST`
	`INSTALL> HELP ADD`
	`INSTALL> HELP REPLACE/OPEN`
		If you omit the file specification, INSTALL displays all known file entries. Qualifiers used to display information include:
		`/FULL`
		`/GLOBAL`
		`/STRUCTURE`

Table 5–7: INSTALL Command Qualifiers

Qualifier	Description
/ACCOUNTING	Activate image level accounting for the installed image. (Discussed in MODULE 9).
/EXECUTE_ONLY	Install the image with execution–only access permitted.
/HEADER_RES	The header of the image file (native images only) remains permanently resident in memory, saving one disk I/O operation per file access. The image is implicitly declared permanently open (/**OPEN**).
/LOG	List the newly created known file entry along with any associated global sections created by the installation.
/NOPURGE	Specifies that this image will not be deleted by the INSTALL command **PURGE**.
/OPEN	Directory information on the image file remains permanently resident in memory, eliminating the usual directory search required to locate a file.
/PRIVILEGED=(priv,...)	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.
/PROTECTED	A shareable image contains protected code (code that runs in kernal or executive mode) but that can be called by a user–level image. Protected images must be declared /**SHARED**.
/SHARED	More than one user can access the read–only and noncopy on–reference read/write sections of the image concurrently. Therefore, only one copy of those sections ever need be in physical memory. The image is implicitly declared permanently open (/**OPEN**).
/WRITEABLE	Shared noncopy, noncopy_on–reference writeable sections are written back to the image file when removed from physical memory (for paging reasons or because no processes are referencing it). The image must also be declared /**SHARED**.

Example 5–5: Installing Known Images (INSTALL.COM)

```
$!
$! Install known images, most frequently activated LAST
$!
$ INSTALL
CREATE SYS$SYSTEM:REPLY                 /PRIV=OPER
CREATE SYS$SYSTEM:BLISS32               /OPEN /SHARED
CREATE SYS$SYSTEM:MACRO32               /OPEN /SHARED
CREATE SYS$SYSTEM:NCP                   /OPEN /SHARED
!
! Reinstall SHOW with WORLD privilege
!
REPLACE SYS$SYSTEM:SHOW                  /OPEN /SHARED /HEADER_RES -
                                         /PRIV=(CMKRNL,WORLD,NETMBX)
!
! Reinstall the linker to be shared in memory
!
REPLACE SYS$SYSTEM:LINK                  /OPEN /SHARED /HEADER_RES
!
```

Starting Local DECnet

- Optional product required to communicate with other systems
- Single–node DECnet included with VMS
- File: STARTNET.COM
  - — VMS operating system installation procedure copies STARTNET.COM into SYS$MANAGER directory
  - — Invoke STARTNET.COM from SYSTARTUP_V5.COM or submit it as a batch job

Creating Reports About the Last System Failure

- System failure may be due to:
  - — Power loss
  - — Hardware errors
  - — Software errors
  - — Intentional actions by system manager
- When the system fails, contents of memory are written to the crash dump file
  - — To analyze the crash dump file:
    1. Invoke System Dump Analyzer (SDA) utility

       ```
       $ ANALYZE/CRASH_DUMP
       ```

    2. Create listing of information in the dump file using SDA commands
  - — Should include commands in SYSTARTUP_V5.COM to create SDA listing file

Example 5–6: System Failure Report Procedure (REPORT_FAILURE.COM)

```
$ !
$ ! Analyze the most recent crash
$ !
$ 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
```

Setting the Maximum Number of Interactive Logins

- VMS default number is 64
- Changeable by setting the symbol STARTUP$INTERACTIVE_LOGINS

```
$ STARTUP$INTERACTIVE_LOGINS == 40
```

- Cannot set the maximum number above the number specified in your VAX processor license

Announcing System Availability

Use the **REPLY** command to announce system availability.

Example 5–7: Announcing System Availability (START_ANNOUNCE.COM)

```
$!
$!  Announce that the system is available
$!
$ WAIT 00:00:45.00     ! Let things settle down a bit...
$!
$ REPLY/ALL/BELL "ISHAM is now available for use."
```

SUMMARY

- Customize the system by:
  - — Establishing paging and swap files
  - — Modifying device characteristics

    SET DEVICE
    SET TERMINAL
    SYSGEN parameters
  - — Defining logical names
  - — Installing images (INSTALL utility)
  - — Establishing queues
  - — Creating system-wide login procedure (SYLOGIN.COM)
- VMS is distributed with several system startup command procedures
  - — Never modify these files
  - — SYS$SYSTEM:STARTUP.COM invokes site-specific command procedures
- System startup site-specific command procedures
  - — SYS$MANAGER:SYPAGSWPFILES.COM
  - — SYS$MANAGER:SYCONFIG.COM
  - — SYS$MANAGER:SYLOGICALS.COM
  - — SYS$MANAGER:SYSTARTUP_V5.COM
- Most terminal characteristics must be set each time the system is rebooted
  - — Use SYCONFIG.COM to set characteristics
  - — Use /**PERMANENT** qualifier on **SET TERMINAL** command
  - — Typical characteristics set at startup

    Speed
    Protection
    Local/remote
    Device type (LA120, VT200, etc.)

- Use the INSTALL utility to:
  - — Install an image for

    Performance improvement
    Privilege requirements
  - — Replace an existing installed image to change installation characteristics
  - — Remove an installed image
- Create reports about system failures using SDA **ANALYZE/CRASH_DUMP**
  - — SDA commands typically executed at system startup to report on last failure
  - — System dump information placed in SYS$SYSTEM:SYSDUMP.DMP
- Define STARTUP$INTERACTIVE_LOGINS to maximum number of interactive users
- Use **REPLY** command to announce system availability

MODULE 6
STARTING UP AND SHUTTING DOWN THE SYSTEM

INTRODUCTION

A VMS system usually remains running for long periods of time. Under certain circumstances, however, you must shut the system down. Such circumstances include:

- Performing hardware or software maintenance
- Backing up the system disk
- Moving or reconfiguring system hardware
- Upgrading or reconfiguring system software
- Losing electrical power or air conditioning
- Losing operating system response

VMS provides a command procedure that shuts the system down at a time you specify, and notifies system users as that time approaches. Other shutdown methods are also available. You can use them when the usual procedure is not effective.

There are many methods of starting up a VMS system. Because you can record your standard system configuration, starting up your system can be as simple as pressing a button. Using other startup methods, you can specify alternate system configurations. Also, if you want to change to a different standard system configuration, you can modify the configuration information VMS reads on startup.

This module assumes that VMS software has already been configured to run on your system, by following the procedures outlined in MODULE 7 and MODULE 5.

OBJECTIVES

To use a VMS system, a system manager should be able to do one or more of the following:

- Describe the phases of system startup
- Describe the functions of the various components of the console subsystem
- Boot the system from the default system device or from an alternate device
- Run standalone utilities or diagnostics
- Use console commands at system startup and while the VMS system is running
- Use system parameters to build data structures, configure devices, or customize the user interface
- Customize the system configuration, using system parameters, boot command procedures, and startup command procedures

RESOURCES

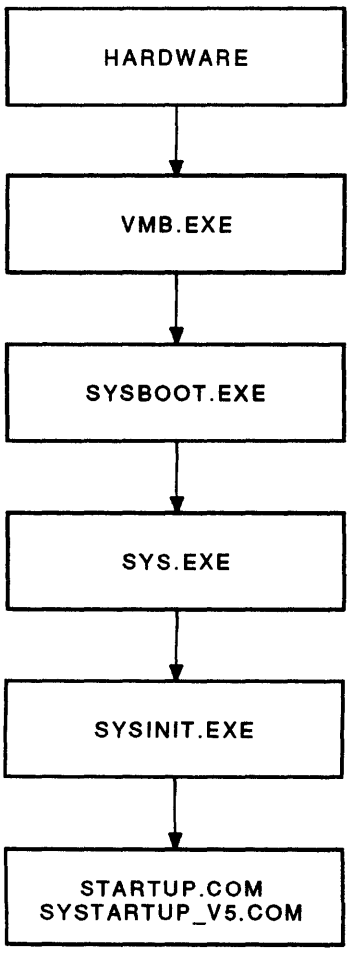
1. *VMS System Manager's Manual*
2. *Guide to Setting Up a VMS System*
3. *VMS System Generation Utility Manual*
4. *VMS Installation and Operations Manual* for your specific VAX processor

STARTING UP A VMS SYSTEM

Phases of Startup

1. Hardware loads VMB.EXE, primary bootstrap program
2. VMB.EXE loads SYS$SYSTEM:SYSBOOT.EXE, secondary bootstrap program
3. SYSBOOT.EXE loads SYS$SYSTEM:SYS.EXE and other executive images
4. VMS executive initializes VMS database, creates process running SYS$SYSTEM:SYSINIT.EXE
5. SYSINIT.EXE continues system initialization, creates batch process executing DCL commands in SYS$SYSTEM:STARTUP.COM
6. The STARTUP.COM command procedure:
   - Executes its own DCL commands
   - Executes DCL commands in nested command procedures
   - Completes system initialization

Figure 6–1: Startup Phases of VMS



VAX Console Subsystem

- Enables you to exercise control over the remaining VAX hardware
- Consists of six parts:
  - — Front panel switches and lights
  - — Console processor
  - — Console device
  - — Console terminal
  - — Console command language (CCL)
  - — Remote diagnosis port (optional)

Table 6–1: VAX Console Subsystem

Component	Function
Console processor	Interprets and executes console commands affecting VAX hardware. It does not require the presence of VMS software.
Console command language (CCL)	Used to express console commands affecting VAX hardware.
Console terminal	Printing terminal used to enter CCL commands and display responses (console I/O mode). When VMS is running, it can also serve as an interactive user terminal (program I/O mode).
Console device	Holds console volume, which may contain console processor microcode, VAX microcode and/or CCL command files. Used to update VMS, install optional VAX software, and run diagnostics.
Front panel switches and lights	Used to control system startup, recover from system failure, display system status, and perform remote diagnosis. Key switch turns power on and off.
Remote diagnosis port	Enables a terminal at the DIGITAL Diagnosis Center to function as a remote console terminal.

Front Panel Switches and Lights

- Used to start up system
- Switches control processor
- Lights indicate processor status

Console Processor and Console Device

- Console processor
  - — A small processor that "front–ends" the VAX processor
  - — Used to monitor the overall operation of the VAX processor
  - — Accomplishes first startup phase
- Console device
  - — Contains console volume
  - — Console volume contains files needed by console processor

    Processor microcode
    Primary bootstrap program
    CCL command files
  - — Console volume usually required for startup
  - — Console volume must be mounted in console device

Default System Device

- Each VAX processor supports a method of specifying a default system device
- If system volume is mounted on default system device, startup is almost automatic
- Steps for startup:
  1. Prepare the system for booting (by setting switches)
  2. If the power is off, turn it on
  3. Boot the system
  4. Prepare system for automatic restart (optional)

Automatic Restart

- Battery backup makes system recovery possible after power failure
  - — An optional hardware component
- Recovery can occur immediately
- If memory is valid after recovery, all processes and batch jobs resume where they left off
- If memory is not valid after recovery, system is initialized

Alternate System Devices

- Can specify system device explicitly at system startup
  - — If console volume is not customized for default startup
  - — To override customized default
- Device codes used at startup are NOT necessarily standard VMS device codes

Standalone Utilities and Diagnostics

- Can run on VAX processor without VMS
  - — Hardware diagnostics
  - — VMS utilities
- Start up like VMS, using console
  - — Larger selection of startup devices
  - — Typically use console device
- Example: standalone BACKUP

Console Commands

Each VAX processor has its own CCL.

Issuing Console Commands at System Startup

- For the startup methods discussed in this module, issue CCL commands after receiving ">>>" prompt
- Console terminal serves as CCL command terminal at startup

Issuing Console Commands While VMS is Running

- Use the console terminal
  - — Program mode: Interactive VMS terminal
  - — Console mode: CCL command terminal
- When console terminal is in console mode
  - — Most VAX processors continue to run
  - — Some VAX processors are halted

Example 6–1: Using the Console Terminal Interactively After Startup

```
%SET-I-INTSET, login interactive limit = 64, current interactive value = 0
  SYSTEM       job terminated at 18-JUL-1984 20:16:46.67

Username:  SYSTEM
Password:

           Welcome to VAX/VMS Version 5.0
$
```

Example 6–2: Using the Console Terminal in Console Mode

```
$ SHOW DEFAULT
  WORK1:[BIERLY]
$ SHOW TERMINAL
Terminal:  _OPA0:     Device_Type:  LA36          Owner:  _OPA0:
                                                Username:  BIERLY

    Input:   300    LFfill:     0      Width:  132      Parity:  None
    Output:  300    CRfill:     0      Page:    24

Terminal Characteristics:
    Interactive     Echo                Type-ahead          No Escape
    No Hostsync     TTsync              Lowercase           No Tab
    Wrap            Hardcopy            No Remote           No Holdscreen
^P
>>>
>>>SET TERMINAL PROGRAM
   Broadcast          No Readsync        No Form
   Fulldup            No Modem           No Local_echo       No Autobaud
   No Hangup          No Brdcstmbx       No DMA              No Altypehd
   Set_sped           Line Editing       Overstrike editing  No Fallback
   No Dialup          No Secure server   No Disconnect       No Pasthru
   No Syspassword     No SIXEL Graphics  No Soft Characters  No Printer Port
   Numeric Keypad     No ANSI_CRT        No Regis            No Block_mode
   No Advanced_video  No Edit_mode       No DEC_CRT          No DEC_CRT2
$
^P
>>>SET TERMINAL PROGRAM
$
```

Customizing Startup and Automatic Restart

- Setting the default system device
  - — System dependent
  - — Can be set by hardware or software (depending on the VAX processor)
- DIGITAL provides many command files on the console volume that can be chosen for DEFBOO.CMD
  - — Use command procedure SYS$UPDATE:SETDEFBOO.COM

SYSTEM PARAMETERS

- Stored in SYS$SYSTEM:VAXVMSSYS.PAR
- Used to build data structures, configure peripheral devices, and customize user interface each time the system starts up
- If modifications are required, they are usually accomplished by the SYSGEN utility or the command procedure SYS$UPDATE:AUTOGEN.COM while VMS is running
- To make modifications while starting up, use the SYSBOOT program during conversational startup
  - — You can specify a minimum startup
  - — Select an alternate file as the source of system parameter values
  - — Set and show individual parameter values
  - — Specify an alternate site-independent startup procedure

Table 6–2: Using SYSBOOT During Conversational Startup

Function	Command Format and Examples

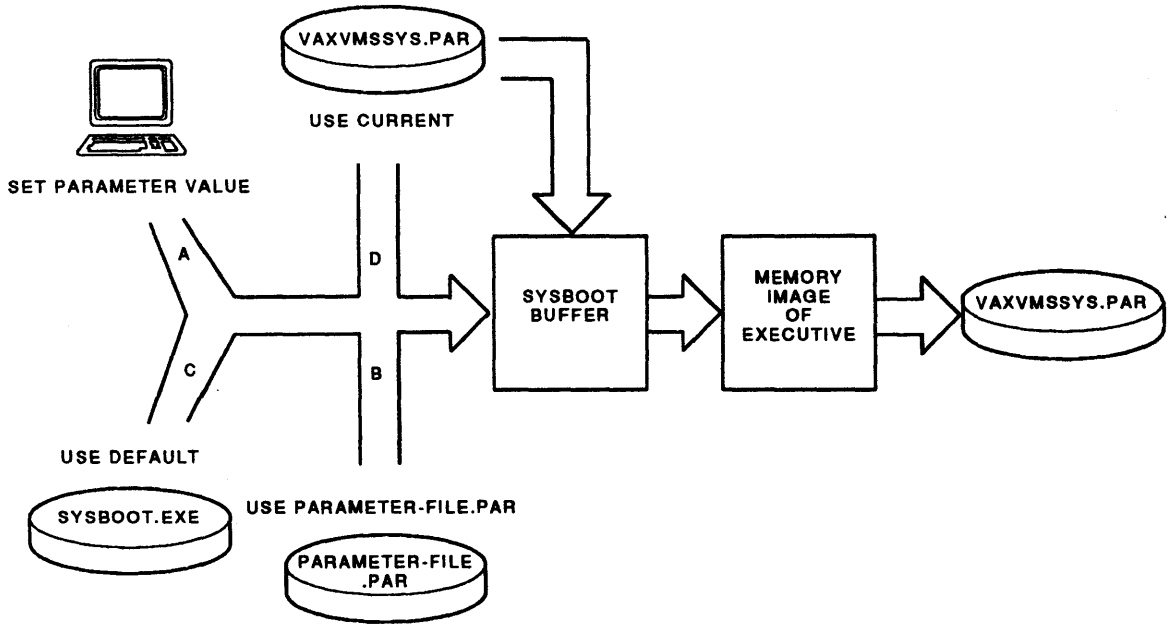
Examining a system parameter or group of parameters	`SYSBOOT> SHOW parameter`
	`SYSBOOT> SHOW /parameter-group`
	`SYSBOOT> SHOW MAXPROCESSCNT`
	`SYSBOOT> SHOW /ALL`

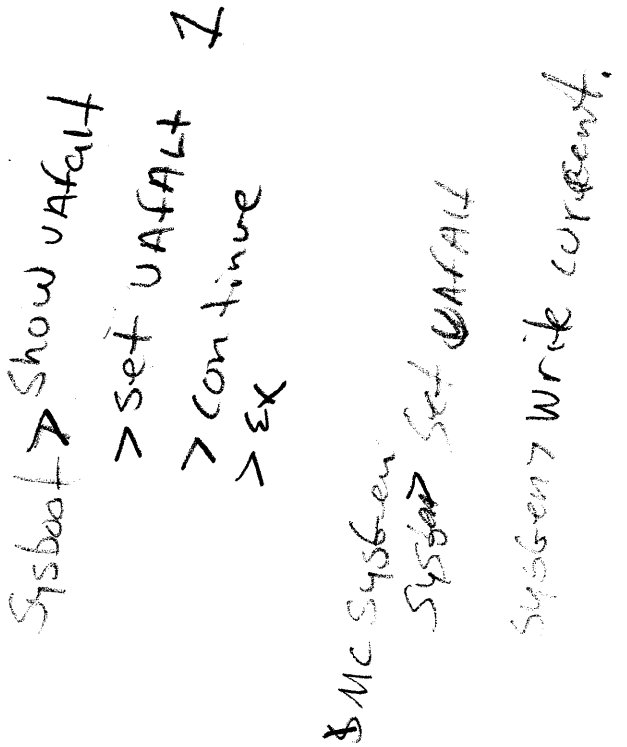
Modifying a system parameter	`SYSBOOT> SET parameter value` `SYSBOOT> SET UAFALTERNATE 1`

Modifying a group of system parameters (.PAR files should be in SYS$SYSTEM)	`SYSBOOT> USE parameter-file.PAR`
	`SYSBOOT> USE CURRENT`
	`SYSBOOT> USE DEFAULT`
	`SYSBOOT> USE ALTPARAM.PAR`

Using an alternate DCL startup file	`SYSBOOT> SET/STARTUP SYS$SYSTEM:startup-file` `SYSBOOT> SET/STARTUP SYS$SYSTEM:ALTSTART`
Exiting SYSBOOT to continue the startup procedure	`SYSBOOT> CONTINUE`

Figure 6–2: Effect of VMS Startup on System Parameters





SPECIFYING THE SYSTEM CONFIGURATION

- At each startup
  - — System device is selected
  - — System parameters determine how VMS database is constructed
  - — DCL commands create system environment
- Files determine default system configuration
  - — Must be customized

Table 6–3: Customizing the System Configuration

Customize	Contents of File	Location of File	Utility Used for Customization
DEFBOO.CMD	CCL (or BOOT58) commands	Console volume	SYS$UPDATE: SETDEFBOO.COM
VAXVMSSYS.PAR	System parameters	SYS$SYSTEM	SYSBOOT, SYS-GEN, or AUTOGEN
SYSTARTUP_ V5.COM SYCONFIG.COM	DCL commands	SYS$MANAGER	Text editor (such as EDT)

Controlling Multiprocessing

- *Symmetric multiprocessing* (SMP)
  - — Introduced with VMS Version 5
  - — Two or more CPUs in one physical cabinet

    Tightly coupled (equal access to VMS code and resources)
    Address a common pool of memory
    Capable of executing instructions simultaneously
    All CPUs must be at same hardware and firmware level
  - — Each CPU has a CPU ID
  - — Jobs can run on any CPU (*dynamic load leveling*)
- *Primary processor*
  - — Logically or physically attached to the console device
  - — Performs initialization activities at system startup
  - — Serves as system timekeeper
- *Secondary processor(s)*
  - — All processors other than the primary processor
- *Available set*
  - — Processors that have passed power-on hardware diagnostics
  - — May or may not be actively involved in system operations
- *Active set*
  - — Processors in the available set that are actively participating in system operations

SYSGEN parameters control processor membership and character.

Table 6–4: SYSGEN Parameters for Multiprocessing Systems

Parameter	Function
MULTIPROCESSING	Determines which synchronization image is loaded into the operating system at boot time.
SMP_CPUS	Determines which processors are brought into the multiprocessing environment at boot time.

DCL commands for SMP systems (require CMKRNL privilege).

Table 6–5: DCL Commands to Control Multiprocessing Systems

Command†	Function
START/CPU [cpu-id,...]	Starts the specified secondary processor or processors.
STOP/CPU [cpu-id,...]	Stops the specified secondary processor or processors.
SHOW CPU [cpu-id,...]	Displays the current state of the processors.

†Requires CMKRNL privilege.

SHUTDOWN

- Three methods of system shutdown
  - — Orderly shutdown
  - — Emergency shutdown from DCL
  - — Emergency shutdown from CCL

Orderly Shutdown

- Preferred method
  - — Warns users of approaching shutdown
  - — Preserves all system and user data
- Command procedures used
  - — SYS$SYSTEM:SHUTDOWN.COM (site-independent)
  - — SYS$MANAGER:SYSHUTDWN.COM (site-specific)
- Steps to follow
  1. Log in as SYSTEM
  2. Issue: `@SYS$SYSTEM:SHUTDOWN`
  3. Respond to message prompts
  4. Wait until the following message is displayed on console terminal:

     ```
     SYSTEM SHUTDOWN COMPLETE -- USE CONSOLE TO HALT THE SYSTEM
     ```

  5. Use the console terminal to type:
     a. CTRL/P
     b. >>> H
- Canceling shutdown
  - — CTRL/Y cancels remaining shutdown functions
  - — Shutdown functions already accomplished must be manually reversed

Functions of SHUTDOWN.COM

- Displays a notice on all terminals that a shutdown is planned
  - — Reminds interactive users periodically as shutdown time approaches
- Prevents users from logging in (unless they have OPER privilege)
- Shuts down DECnet
- Stops batch and print queues
- Executes site–specific DCL commands in SYSHUTDWN.COM
- Stops user processes
- Removes installed images
- Dismounts mounted volumes other than the system disk (and, if requested, spins down the volumes)
- Closes the operator's log file
- Writes error log entries to disk
- Causes the VMS paging mechanism to record on disk any file changes that have occurred in memory
- Writes a copy of physical memory to the dump file
- Shuts down secondary processors for multiprocessor systems
- Dismounts the system disk

Automatic Reboot After Orderly Shutdown

- For automatic rebooting
  - — Set the processor for auto–reboot (processor–dependent).
  - — Answer "YES" to the automatic reboot question in the shutdown procedure.

Example 6–3: User View of Orderly System Shutdown

```
Username:  JONES
Password:

           Welcome to node SUPER
$
$ SHOW DEFAULT
  WORK1:[JONES]
$

SHUTDOWN message from user SYSTEM at _OPA0:     19:59:56
SUPER will shut down in 10 minutes; back up AT 10:00.
PREVENTATIVE MAINTENANCE

$ SHOW DEFAULT
  WORK1:[JONES]
$

SHUTDOWN message from user SYSTEM at _OPA0:       20:05:18
SUPER will shut down in 5 minutes; back up AT 10:00.  Please log out.
PREVENTATIVE MAINTENANCE

$

SHUTDOWN message from user SYSTEM at _OPA0:       20:07:24
SUPER will shut down in 3 minutes; back up AT 10:00.  Please log out.
PREVENTATIVE MAINTENANCE

$

SHUTDOWN message from user SYSTEM at _OPA0:       20:08:30
SUPER will shut down in 2 minutes; back up AT 10:00.  Please log out.
PREVENTATIVE MAINTENANCE

$


SHUTDOWN message from user SYSTEM at _OPA0:       20:09:39
SUPER will shut down in 1 minutes; back up AT 10:00.  Please log out.
PREVENTATIVE MAINTENANCE

$

SHUTDOWN message from user SYSTEM at _OPA0:       20:10:45
SUPER will shut down in 0 minutes; back up AT 10:00.  Please log out.
PREVENTATIVE MAINTENANCE

$ SHO
```

(The system has logged out the user's process)

Emergency Shutdown with DCL Commands

- Use if orderly shutdown fails
- Use in emergency for speed
- For speed when no emergency exists, usually suffices to use orderly shutdown
  - — Specify "0" as the number of minutes until shutdown
- Steps to follow:
  1. Log in as SYSTEM (procedure requires CMKRNL privilege)
  2. Issue: `$ RUN SYS$SYSTEM:OPCCRASH`
  3. Wait until the following message is displayed on console terminal:

     ```
     SYSTEM SHUTDOWN COMPLETE -- USE CONSOLE TO HALT THE SYSTEM
     ```

  4. Use the console terminal to type:
     a. CTRL/P
     b. >>> H
- Functions of OPCCRASH
  - — Writes error log entries to disk
  - — Causes the VMS paging mechanism to record on disk any file changes that have occurred in memory
  - — Writes a copy of physical memory to the dump file
  - — Dismounts the system disk

Forcing a Shutdown with CCL Commands

- Use if no system response to interactive terminal input
- Console terminal only
- Functions
  - — Writes crash dump file
  - — Displays register contents on console terminal

Automatic Reboot After Forcing a Shutdown

- For automatic rebooting
  - — Set the processor for auto–reboot (processor–dependent)

SUMMARY

- System is started in phases
- Console system exercises control over the remainder of the system
  - — A fundamental component of startup activities
  - — Console configuration and commands vary depending on processor type
  - — Console commands are issued on the console terminal

    At boot time (conversational boot)
    While VMS is running
  - — Used to change default system parameters

    Default system device
    Other system parameters
- Use standalone SYSBOOT utility to change system parameters
- Three types of system shutdown situations
  - — Orderly shutdown

    Preferred method
    Warns users of impending shutdown
    Preserves all system and user data
  - — Emergency shutdown from DCL
  - — Emergency shutdown from CCL
- Automatic restart
  - — Selected by front panel switches or system parameter
  - — Complete recovery can be achieved with battery backup hardware

APPENDIX A

This appendix contains processor–specific information and examples of terminal output, specifically for:

- VAX–11/730
- VAX–11/750
- VAX–11/780

Information on other processors can be found in the *VMS Installation and Operation Guide* for the particular processor.

Table 6–6: VAX Front Panel Switches

Function	Switch		
	VAX–11/780	VAX–11/750	VAX–11/730
Turns power on and off. Enables and disables console terminal and remote diagnosis port.	Key switch	Key switch	Key switch
Affects recovery from system failure.	AUTORESTART	POWER ON ACTION†	AUTORESTART/BOOT
Requests system initialization	BOOT	INITIALIZE	AUTORESTART/BOOT
Selects system device at system startup and automatic restart.	Not done by switch for this processor	BOOT DEVICE	Not done by switch for this processor

† Action taken on recovery is a choice of four, read counterclockwise from the top of the switch: BOOT, RESTART/BOOT, HALT, RESTART/HALT.

Table 6–7: VAX Front Panel Lights

Function When Lit	Panel Light		
	VAX–11/780	VAX–11/750	VAX–11/730
Processor is not running. (May indicate a problem.)	ATTN (red)	ERROR (red)	—
Processor is running.	RUN (green)	RUN (green)	RUN (red)
Processor is powered on. (Not lit when key switch is in the OFF position.)	POWER (green)	POWER (green)	DC ON (red)
Remote diagnosis operation is in progress.	REMOTE (red)	See note†	R/D (red)

†For the VAX–11/750, remote diagnosis is indicated by the lighting of one or more labels to the left of the lights mentioned in this table. Figure 6–3 shows these block letter labels.

Figure 6–3: VAX Console Control Panels

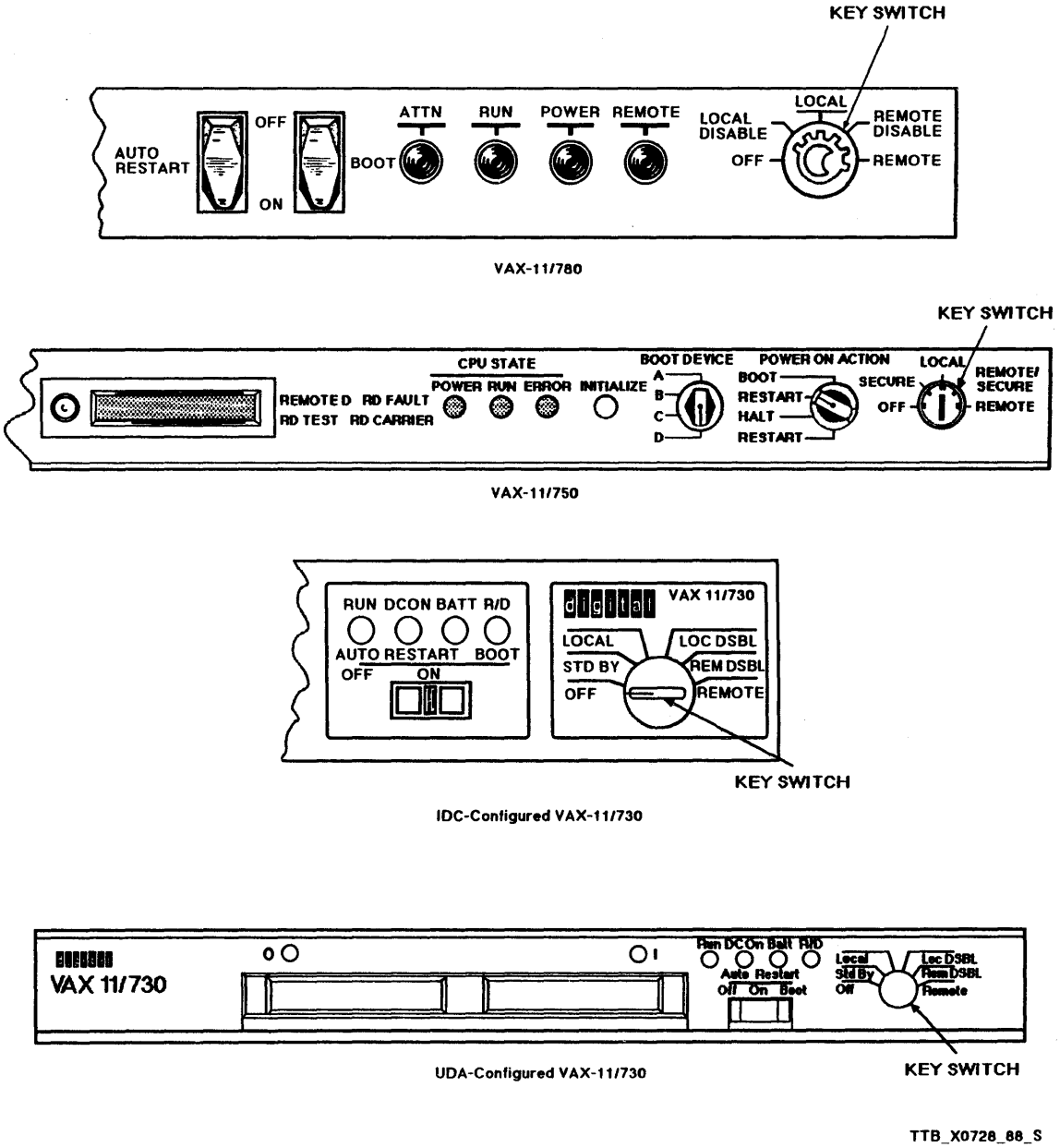


Table 6–8: VAX Console Devices

Processor	Console Device	Location	Device
VAX–11/780	RX01 diskette drive	Inside right door of processor cabinet, at bottom. Pivots out for access.	CSA1:
VAX–11/750	TU58 cartridge tape drive	On console front panel.	CSA1:
VAX–11/730	Dual TU58 cartridge tape drive	On console front panel.	CSA1:
		On one type of VAX–11/730 processor, the second drive is inside the processor cabinet. You must slide the processor out of the cabinet to reach it.	CSA2:
		On the other type of VAX–11/730 processor, the second drive is located on the console front panel next to the first drive.	
		Both types of processors boot from the second console drive by default.	

Table 6–9: Starting Up a VMS System from Power Off Using the Default System Device

Step	VAX–11/780	VAX–11/750	VAX–11/750 (using BOOT58)	VAX–11/730
1	Set AUTORESTART switch to OFF. Place console volume in console device.	Set POWER ON ACTION switch to HALT. Set BOOT DEVICE switch to default setting for system device (B, C, or D). The default setting is normally D, but this depends on your configuration.	Set POWER ON ACTION switch to HALT. Place console volume in console device. Set BOOT DEVICE switch to setting for console device, A.	Set AUTORESTART/ BOOT switch to OFF. Place the console volume in the console device.†
2	Set key switch from OFF to LOCAL.	Set key switch from OFF to LOCAL.	Set key switch from OFF to LOCAL.	Set key switch from OFF to LOCAL.
3	When the >>> prompt appears on console terminal, toggle BOOT switch from OFF to ON.	When the >>> prompt appears on console terminal, set POWER ON ACTION switch to BOOT. Press INITIALIZE button.	When the >>> prompt appears on console terminal, set POWER ON ACTION switch to BOOT. Press INITIALIZE button. When the BOOT58> prompt appears, type BOOT RETURN.	When the >>> prompt appears on console terminal, toggle AUTORESTART/BOOT switch from OFF to BOOT.
4	Set key switch from LOCAL to LOCAL DISABLE. Set AUTORESTART switch to ON.	Set key switch from LOCAL to SECURE. Set POWER ON ACTION switch to RESTART/ BOOT.	Set key switch from LOCAL to SECURE. Set BOOT DEVICE switch to default setting for system device (B, C, or D), and set POWER ON ACTION switch to RESTART/BOOT.	Set key switch from LOCAL to LOC DSBL. (AUTORESTART/BOOT switch is set to ON.)

†For one type of VAX–11/730, the console volume (CSA2:) is always loaded in the console drive located inside the processor cabinet. You do not have to load the console volume on this type of processor.

Example 6–4: Default Startup of a VAX-11/780 from Power Off

```
        CPU HALTED, SOMM CLEAR,STEP=NONE,CLOCK=NORM
        RAD=HEX,ADD=PHYS,DAT=LONG,FILL=00,REL=00000000
        INIT SEQ DONE
        HALTED AT 000000

        (RELOADING WCS)
        LOAD DONE, 0800 MICROWORDS LOADED
        VER: PCS=01 WCS=0E-10 FPLA-0E CON=V07-00-L

** >>>

        CPU HALTED
        INIT SEQ DONE
        HALT INST EXECUTED
        HALTED AT 200034F9

        G 00000000E 00000200
        LOAD DONE, 00005000 BYTES LOADED

VAX/VMS Version X2M9 3-JUN-1984 08:31

%%%%%%%%%%  OPCOM 24-JUL-1984 18:36:03  %%%%%%%%%%
Logfile has been initialized by operator _OPA0:
Logfile is SYS$SYSROOT:[SYSMGR]OPERATOR.LOG;9

%SET-I-INTSET, login interactive limit = 64, current interactive value = 0
  SYSTEM       job terminated at 24-JUL-1984 18:36:40.72

@
```

Example 6–5: Default Startup of a VAX-11/750 Using the Default System Disk from Power Off

```
   %%
   00000000  16
** >>>
   %%

   VAX/VMS Version Z2M9 11-JUL-1984 08:31

   %%%%%%%%%%  OPCOM  24-JUL-1984 18:32:34.36  %%%%%%%%%%
   Logfile has been initialized by operator _OPA0:
   Logfile is SYS$SYSROOT:[SYSMGR]OPERATOR.LOG;2

   %SET-I-INTSET, login interactive limit = 50, current interactive value = 0
    SYSTEM        job terminated at 23-JUL-1984 18:36:53.85

$
```

Example 6–6: Default Startup of a VAX-11/730 from Power Off

```
       CONV011
       ?27  READ ERROR DD1
       VERSION 03.00
       >>>@POWER.CMD
       >>>L/C CONSOLE.CPU                !Version 55
       >>>L/C/S:0800 MMIE.CPU            !Version 55
       >>>L/C/S:0E00 POWER.CPU           !Version 55
       >>>S/C OB
       >>>W
       >>>@CODE01.CMD
       >>>L/C/S:0E00 FP.CPU              !Version 55
       >>>L/C/S:1A00 BITFLD.CPU          !Version 55
       >>>L/C/S:1D00 CM.CPU              !Version 55
       >>>L/C/S:2200 BASIC.CPU           !Version 55
       >>>L/C/S:3B00 QUEUE.CPU           !Version 55
       >>>L/C/S:4000 IDC.CPU             !Version 55
       >>>I
** >>>
       >>>@DD0:DEFBOO.CMD
       >>>I
       >>>D/G/L 0 00A80003
       >>>D/G/L 1 3
       >>>D/G/L 2 3FB86
       >>>D/G/L 3 0
       >>>D/G/L 4 0
       >>>D/G/L 5 0
       >>>E SP
       G 0000000E 00000200
       >>>L/P/S:@ VMB.EXE
       >>>S @

VAX/VMS Version X2LR 18-MAY-1984 20:50

PLEASE ENTER DATE AND TIME (DD-MMM-YYYY HH:MM) 07-JUL-1984 18:55
%%%%%%%%%%  OPCOM  7-JUL-1984 18:55:59.65 %%%%%%%%%%
Logfile has been initialized by operator _OPA0:
Logfile is SYS$SYSROOT:[SYSMGR]OPERATOR.LOG;50

%SET-I-INTSET, login interactive limit = 64, current interactive value = 0
  SYSTEM       job terminated at 7-JUL-1984 18:57:40.34
$
```

Table 6–10: Starting Up a VMS System from Power On Using the Default System Device

Step	VAX–11/780	VAX–11/750	VAX–11/750 (using BOOT58)	VAX–11/730
1	Set AUTORESTART switch to OFF. Place console volume in console device.	Set POWER ON ACTION switch to HALT. Set BOOT DEVICE switch to default setting for system device (B, C, or D). The default setting is normally D, but this depends on your configuration.	Set POWER ON ACTION switch to HALT. Place console volume in console device. Set BOOT DEVICE switch to setting for console device, A.	Set AUTORESTART/BOOT switch to OFF. Place the console volume in the console device.†
2	Set key switch to LOCAL.	Set key switch to LOCAL.	Set key switch to LOCAL.	Set key switch to LOCAL.
3	Toggle BOOT switch from OFF to ON. If nothing happens at console terminal, type CTRL/P. When >>> prompt appears, type HALT RETURN. When >>> prompt appears again, type BOOT RETURN.	Set POWER ON ACTION switch to BOOT. Press INITIALIZE button.	Set POWER ON ACTION switch to BOOT. Press INITIALIZE button. When the BOOT58> prompt appears, type: BOOT RETURN.	Toggle AUTORESTART/BOOT switch from OFF to BOOT. If nothing happens at console terminal, type CTRL/P. When >>> prompt appears, type B RETURN.
4	Set key switch from LOCAL to LOCAL DISABLE. Set AUTORESTART switch to ON.	Set key switch from LOCAL to SECURE. Set POWER ON ACTION switch to RESTART/BOOT.	Set key switch from LOCAL to SECURE. Set BOOT DEVICE switch to default setting for system device (B, C, or D) and set POWER ON ACTION switch to RESTART/BOOT.	Set key switch from LOCAL to LOC DSBL. Set AUTORESTART/BOOT switch to ON.

†For one type of VAX–11/730, the console volume (CSA2:) is always loaded in the console drive located inside the processor cabinet. You do not have to load the console volume on this type of processor.

Table 6–11: Automatic Powerfail Recovery

Restart Response After Power Failure	VAX–11/780 AUTORESTART Switch Setting	VAX–11/750 POWER ON ACTION Switch Setting	VAX–11/730 AUTORESTART /BOOT Switch Setting
If memory is valid, try to recover from system failure. Otherwise, perform fresh startup.	ON	RESTART/BOOT	ON
If memory is valid, try to recover from system failure. Otherwise, halt processor.	—	RESTART/HALT	—
Halt processor	OFF	HALT	OFF
Perform fresh startup	—	BOOT	—

Table 6–12: Starting Up a VMS System, Explicitly Specifying the System Device

VAX–11/780	VAX–11/750	VAX–11/750 (using BOOT58)	VAX–11/730
Step 1: Set AUTORESTART switch to OFF. Place console volume in console device.	Set POWER ON ACTION switch to HALT.	Set POWER ON ACTION switch to HALT. Place console volume in console device. Set BOOT DEVICE switch to setting for console device, A.	Set AUTORESTART/BOOT switch to OFF. Place the console volume in the console device.[1]
Step 2: Set key switch to LOCAL.	Set key switch to LOCAL.	Set key switch to LOCAL.	Set key switch to LOCAL.
Step 3: If >>> prompt is not displayed on console terminal, obtain it by typing CTRL/P. Type HALT RETURN. Then, type BOOT ddu RETURN, where ddu is the startup code for system device.	If >>> prompt is not displayed on console terminal, obtain it by typing CTRL/P. Type B ddcu RETURN, where ddcu is the startup code for the system device. (You must have the ROM for this device installed first.)	Set POWER ON ACTION switch to BOOT. Press INITIALIZE button. When the BOOT58> prompt appears on console terminal, type BOOT ddcu RETURN, where ddcu is the startup code for the system device.	If >>> prompt is not displayed on console terminal, obtain it by typing CTRL/P. Type B ddu RETURN, where ddu is the startup code for the system device.
Step 4: Set key switch from LOCAL to LOCAL DISABLE. Set AUTORESTART switch to ON.	Set key switch from LOCAL to SECURE. Set BOOT DEVICE switch to default setting for system device (B, C, or D), and set POWER ON ACTION switch to RESTART/BOOT.	Set key switch from LOCAL to SECURE. Set BOOT DEVICE switch to default setting for system device (B, C, or D), and set POWER ON ACTION switch to RESTART/BOOT.	Set key switch from LOCAL to LOC DSBL. Set AUTORESTART/BOOT switch to ON.

[1]For one type of VAX–11/730, the console volume (CSA2:) is always loaded in the console drive located inside the processor cabinet. You do not have to load the console volume on this type of processor.

Table 6–13: Device Codes Used at System Startup

Device and Controller[1]	VAX–11/780 Support[2]	VAX–11/750 Support[3]	VAX–11/730 Support[2]	Device Code
Unibus				
RA60	S	S	S	DU
RA80	S	S	S	DU
RA81	S	S	S	DU
RL02		D	D	DL
RK06	D			DM
RK07	S	S	D	DM
Massbus				
RP05	S			DB
RP06	S	D		DB
RP07	S			DB
RM03	S	S		DB
RM05	S	S		DB
RM80	S	S		DB
IDC				
RL02			S	DQ
R80			S	DQ
Console device				
RX01	D			CS
TU58		D	D	CS[4]

[1] Letter codes:
S = device supported as system device or data device.
D = device supported as data device.

[2] For VAX–11/780, VAX–11/730: the full device code format is **dd**, where **dd** is the device code and **u** is the unit number.

[3] For the VAX–11/750, the full device code format is **ddcu**, where **dd** is the code in the above table, **c** is the channel letter, and **u** is the unit number. The channel letter, which is hardware dependent, is usually A.

[4] On VAX–11/750: specify CS when using BOOT58; otherwise, specify DD when using CCL.

Example 6–7: Startup of a VAX-11/780 Specifying an RM05 as System Device from Power On

```
>>>BOOT DB0

        CPU HALTED
        INIT SEQ DONE
        HALT INST EXECUTED
        HALTED AT 20003552

        G  0000000E  00000200
        LOAD DONE,  00005000 BYTES LOADED

  VAX/VMS Version X2M9 3-JUN-1984 08:31

%%%%%%%%%%  OPCOM  23-JUN-1984 08:38:22.82  %%%%%%%%%%
Logfile has been initialized by operator _OPA0:
Logfile is SYS$SYSROOT:[SYSMGR]OPERATOR.LOG;9

%MOUNT-I-MOUNTED, BIG_BIRD     mounted on _DRA1:
%MOUNT-I-MOUNTED, BERT         mounted on _DRA2:
%MOUNT-I-MOUNTED, ERNIE        mounted on _DRA3:
Job STARTNET (queue SYS$BATCH, entry 1166) started on SYS$BATCH
%RUN-S-PROC_ID, identification of created process is 00000089
%%%%%%%%%%  OPCOM  23-JUN-1984 08:39:33.82  %%%%%%%%%%
Message from user SYSTEM
VAXsim E1.4 startup - Using SYS$SYSROOT:[SYSERR]VAXSIMDAT.DAT;1

Reply received from user SYSTEM at Batch  08:39:42
                            -- Ready for use.

%SET-I-INTSET, login interactive limit = 64, current interactive value = 0
%%%%%%%%%%  OPCOM  23-JUN-1984 08:39:48.60  %%%%%%%%%%
Message from user DECNET
DECnet starting

   SYSTEM       job terminated at 23-JUN-1984 08:39:48.90
```

Example 6–8: Startup of a VAX-11/750 Specifying an RM03 as System Device from Power On

```
>>>B DBA0
%%


  VAX/VMS Version X2M9 3-JUN-1984 08:31

%%%%%%%%%%  OPCOM  23-JUN-1984 09:28:33.96  %%%%%%%%%%%%%%
Logfile has been initialized by operator _MOTHER$OPA0:
Logfile is SYS$SYSROOT:[SYSMGR]OPERATOR.LOG;32

%MOUNT-I-MOUNTED,CSSEDISK     mounted on _DRB1:
%RUN-S-PROC_ID, identification of created process is 00000048
Job STARTNET (queue SYS$BATCH, entry 152) started on SYS$BATCH
%SET-I-INTSET, login interactive limit = 50, current interactive value = 0

Reply received on MOTHER from user SYSTEM at MOTHER Batch   09:30:12
        VAX-11/750 System initialized

   SYSTEM        job terminated at 23-JUN-1984 09:30:12.52
%%%%%%%%%%  OPCOM  23-JUN-1984 09:30:34.07  %%%%%%%%%%
Message from user DECNET on MOTHER
DECnet starting
```

Example 6–9: Startup of a VAX–11/730 Specifying an RL02 as System Device from Power on

```
>>B DM0
>>>@DD1:D,0BOO.CMD
>>>I
>>>D/G 0 1
>>>D/G 1 3
>>>D/G 2 3FF20
>>>D/G 3 0
>>>D/G 4 0
>>>D/G 5 0
>>>E SP
G 0000000E 00000200
>>>L/P/S:@ VMB.EXE
>>>S@

  VAX/VMS Version X2M9  3-JUN-1984 08:31

%%%%%%%%%%  OPCOM  23-JUN-1984  09:21:10.93  %%%%%%%%%%
Logfile has been initialized by operator _OPA0:
Logfile is SYS$SYSROOT:[SYSMGR]OPERATOR.LOG;4

%SET-I-INTSET, login interactive limit = 64, current interactive value = 0
  SYSTEM       job terminated at 23-JUN-1984 09:21:24/59
```

Table 6–14: Typical Console Commands

Command Function	VAX–11/780	VAX–11/750	VAX–11/750 (BOOT58)	VAX–11/730
Starting up VMS or a standalone utility or diagnostic	BOOT	B	BOOT	B
Examining a processor register or memory location	EXAMINE	E	EXAMINE	E
Placing data in a processor register or memory location	DEPOSIT	D	DEPOSIT	D
Initializing state of VAX processor	INITIALIZE	I	—	I
Executing commands in a file	@file–spec	—	@file–spec	@file–spec
Reloading and restarting the console processor	REBOOT	—	—	—

Table 6–15: Issuing Console Commands While VMS is Running

Command Function	VAX–11/780 CCL	VAX–11/750 CCL	VAX–11/730 CCL
Changing mode of console terminal from program mode to console mode	CTRL/P (does not halt VAX processor)	CTRL/P (halts VAX processor)	CTRL/P (halts VAX processor)
Changing mode of console terminal from console mode to program mode	SET TERMINAL PROGRAM	C (restarts halted VAX processor)	C (restarts halted VAX processor)
Halting VAX processor	HALT	CTRL/P	CTRL/P
Restarting halted VAX processor without taking console terminal out of console mode	CONTINUE	Not possible on this processor	Not possible on this processor

Table 6–16: Disabling the Use of Console Mode

Function of Key Switch Setting	VAX–11/780	VAX–11/750	VAX–11/730
Enables CTRL/P	LOCAL	LOCAL	LOCAL
Disables CTRL/P	LOCAL DISABLE	SECURE	LOC DSBL

Table 6–17: CCL Command Files Used at Startup

Device Code Used at System Startup (See Table 6–13)	Command Files on Console Volume Executed by BOOT Command
DB (11/750 and 11/780 only)	DBnBOO.CMD (n = 0,1,...,7,A,B)
DM (11/750 and 11/780 only)	DMnBOO.CMD (n = 0,1,2,3,A,B,)
DQ (11/730 only)	DQnBOO.CMD (n = 0,1,2,3)
DU (11/780 and 11/730 only)	DU0BOO.CMD
CS	CS1BOO.CMD

Example 6–10: File DB0BOO.CMD from a VAX-11/780 Console Volume

```
$ @SYS$UPDATE:DXCOPY
        Command file to copy files to/from the system
        console storage medium and the current directory.

Is system console storage medium mounted (Y/N)?:  N
Please place the system console medium in the console drive
and type [RETURN] when ready:
%MOUNT-I-MOUNTED, VAX console mounted on _CSA1:
Copy from console medium (Y/N)?:  Y
Enter console file name:  DB0BOO.CMD
%EXCHANGE-S-MOUNTED, th RT-11 volume _CSA1:  has been mounted
%EXCHANGE-S-COPIED, _CSA1:DB0BOO.CMD copied to SYS$SYSROOT:[SYSMGR]DB0BOO.CMD;3,
 20 records
%MOUNT-I-MOUNTED, VAX console mounted on _CSA1:
$
$ TYPE SYS$MANAGER:DB0BOO.CMD
!
!       DB0 BOOT COMMAND FILE - DB0BOO.CMD
!
HALT                    !  HALT PROCESSOR
UNJAM                   !  UNJAM SBI
INIT                    !  INIT PROCESSOR
DEPOSIT/I 11 20003800   !  SET UP SCBB
DEPOSIT R0 0            !  DISK PACK DEVICE TYPE
DEPOSIT R1 8            !  MBA TR=8
DEPOSIT R2 0            !  ADAPTER UNIT = 0
DEPOSIT R3 0            !  CONTROLLER UNIT = 0
DEPOSIT R4 0            !  BOOT BLOCK LBN (UNUSED)
DEPOSIT R5 4000         !  SOFTWARE BOOT FLAGS
DEPOSIT FP 0            !  SET NO MACHINE CHECK EXPECTED
START 20003000          !  START ROM PROGRAM
WAIT DONE               !  WAIT FOR COMPLETION
                        !
EXAMINE SP              !  SHOW ADDRESS OF WORKING MEMORY+2X200
LOAD VMB.EXE/START:@    !  LOAD PRIMARY BOOTSTRAP
START @                 !  AND START IT
$
```

Example 6–11: Customizing the Console Volume

```
$  @SYS$UPDATE:SETDEFBOO

          Command file to set default boot command file.

Please place the system console medium in the console drive.

Is the console medium ready to be mounted?  (Y/N):  Y

%MOUNT-I-MOUNTED, VAX console mounted on _CSA1:

Enter name of default boot command file:  DB0B00.CMD

%EXCHANGE-S-MOUNTED, the RT-11 volume _CSA1:  has been mounted
%EXCHANGE-S-DELETEPREV, previous copy of _CSA1:DEFB00.CMD deleted
%EXCHANGE-S-COPIED, _CSA1:DB0B00.CMD copied to _CSA1:DEFB00.CMD, 2 blocks

Default boot command file now replaced ith DB0B00.CMD.

Are you satisfied with this file as the default boot command file?  (Y/N):  Y

%MOUNT-I-MOUNTED, VAX console mounted on _CSA1:
$ DISMOUNT CSA1
$
```

Example 6–12: File RESTAR.CMD from a VAX-11/780 Console Volume

```
$  @SYS$UPDATE:DXCOPY
         Command file to copy files to/from the system
         console storage medium and the current directory.

Is system console storage medium mounted (Y/N)?:  N
Please place the system console medium on the console drive
and type [RETURN] when ready:
%MOUNT-I-MOUNTED, VAX console mounted on _CSA1:
Copy from console medium (Y/N)?:  Y
Enter console file name:  RESTAR.CMD
%EXCHANGE-S-MOUNTED, the RT-11 volume _CSA1:  has been mounted
%EXCHANGE-S-COPIED, _CSA1:RESTAR.CMD copied to SYS$SYSROOT:[SYSMGR]RESTAR.CMD;1,
 20 records
%MOUNT-I-MOUNTED, VAX console mounted on _CSA1:
$
$ TYPE RESTAR.CMD
!
!       RESTART COMMAND FILE - RESTAR.CMD
!
!       THIS COMMAND FILE IS INVOKED IN THE EVENT OF POWER RECOVERY AND
!       OTHER CONSOLE DETECTED RESTART CONDITIONS IF THE AUTO RESTART SWITCH
!       IS SET.  IT CAN ALSO BE INVOKED MANUALLY WITH THE COMMAND:
!                               @RESTAR.CMD
!
HALT                    ! HALT PROCESSOR
UNJAM                   ! UNJAM SBI
INIT                    ! INITIALIZE PROCESSOR
DEPOSIT/I 11 20003800   ! SET ADDRESS OF SCB BASE
DEPOSIT R0 0            ! CLEAR UNUSED REGISTERS
DEPOSIT R1 3            ! UBA TR=3
DEPOSIT R2 0            ! CLEAR UNUSED REGISTER
DEPOSIT R3 0            ! CLEAR UNUSED REGISTER
DEPOSIT R4 0            ! CLEAR UNUSED REGISTER
DEPOSIT R5 0            ! CLEAR UNUSED REGISTER
DEPOSIT FP 0            ! NO MACHINE CHECK EXPECTED
START 20003004          ! START RESTART REFEREE
$
$  DISMOUNT CSA1
$
```

Table 6–18: Starting Up a VMS Operating System Conversationally

VAX–11/780	VAX–11/750	VAX–11/750 (using BOOT58)	VAX–11/730
Step 1: Set AUTORESTART switch to OFF. Place console volume in console device.	Set POWER ON ACTION switch to HALT. Set BOOT DEVICE switch to default setting for system device (B, C, or D).	Set POWER ON ACTION switch to HALT. Place console volume in console device. Set BOOT DEVICE switch to setting for console device, A.	Set AUTORESTART/BOOT switch to OFF. Place the console volume in the console device.†
Step 2: Set key switch to LOCAL.	Set key switch to LOCAL.	Set key switch to LOCAL.	Set key switch to LOCAL.
Step 3: If >>> prompt is not displayed on console terminal, obtain it by typing CTRL/P. Type HALT RETURN. Then, type @dduGEN RETURN where ddu is the startup code for the system device.	If >>> prompt is not displayed on console terminal, obtain it by typing CTRL/P. Type B/1 RETURN.	Set POWER ON ACTION switch to BOOT. Press INITIALIZE button. When the BOOT58> prompt appears on console terminal, type D/G 3 u RETURN, @ddcGEN RETURN, where ddcu is the startup code for the system device.	If >>> prompt is not displayed on console terminal, obtain it by typing CTRL/P. Type @dduGEN RETURN, where ddu is the startup code for the system device.
Step 4: Set key switch from LOCAL to LOCAL DISABLE. Set AUTORESTART switch to ON.	Set key switch from LOCAL to SECURE. Set POWER ON ACTION switch to RESTART/BOOT.	Set key switch from LOCAL to SECURE. Set BOOT DEVICE switch to default setting for system device (B, C, or D), and set POWER ON ACTION switch to RESTART/BOOT.	Set key switch from LOCAL to LOC DSBL. Set AUTORESTART/BOOT switch to ON.

†For one type of VAX–11/730, the console volume (CSA2:) is always loaded in the console drive located inside the processor cabinet. You do not have to load the console volume on this type of processor.

Example 6–13: Typical Conversational Startup

```
>>>@DBOGEN
!
!       DBO CONVERSATIONAL BOOT COMMAND FILE - DBOGEN.
!       BOOT FROM DBO AND STOP IN SYSBOOT TO ALTER PARAMETERS
!
HALT                        ! HALT PROCESSOR
          CPU HALTED
UNJAM                       ! UNJAM SBI
INIT                        ! INIT PROCESSOR
          INIT SEQ DONE
DEPOSIT/I 11 20003800       ! SET UP SCBB
DEPOSIT R0 0                ! DISK PACK DEVICE TYPE
DEPOSIT R1 8                ! MBA TR=8
DEPOSIT R2 0                ! ADAPTER UNIT = 0
DEPOSIT R3 0                ! CONTROLLER UNIT = 0
DEPOSIT R4 0                ! BOOT BLOCK LBN (UNUSED)
DEPOSIT R5 4001             ! SOFTWARE BOOT FLAGS (CONVERSATIONAL BOOT)
DEPOSIT FP 0                ! SET NO MACHINE CHECK EXPECTED
START 20003000              ! START ROM PROGRAM
WAIT DONE                   ! WAIT FOR COMPLETION
             HALT INST EXECUTED
             HALTED AT 20003552
                            !
EXAMINE SP                  !  SHOW ADDRESS OF WORKING MEMORY +2X200
          G 0000000E  00000200
LOAD VMB.EXE/START:@        !  LOAD PRIMARY BOOTSTRAP
          LOAD DONE, 00005000 BYTES LOADED
START @                     ! AND START IT
<@EOF>
<@EXIT>
SYSBOOT>  USE ALTPARAM.PAR
SYSBOOT>  SET UAFALTERNATE 1
SYSBOOT>  SET/STARTUP SYS$SYSTEM:ALTSTART
SYSBOOT>  CONTINUE
```

Example 6–13: Typical Conversational Startup (Cont.)

```
     VAX/VMS Version X2M9 3-JUN-1984 08:31

%%%%%%%%%%  OPCOM  26-JUN-1984 20:17:24.90  %%%%%%%%%%
Logfile has been initialized by operator _OPA0:
Logfile is SYS$SYSROOT:[SYSMGR]OPERATOR.LOG;13

%MOUNT-I-MOUNTED, BIG_BIRD      mounted on _DRA1:
%MOUNT-I-MOUNTED, BERT          mounted on _DRA2:
%MOUNT-I-MOUNTED, ERNIE         mounted on _DRA3:
%MOUNT-F-VOLINV, volume is not software enabled
Job STARTNET (queue SYS$BATCH, entry 259) started on SYS$BATCH
%RUN-S-PROC_ID, identification of created process is 0000008A
%%%%%%%%%%  OPCOM  26-JUN-1984 20:18:42.98  %%%%%%%%%%
Message from user SYSTEM
VAXsim E1.4 startup - Using SYS$SYSROOT:[SYSERR]VAXSIMDAT.DAT;1

Reply received from user SYSTEM at Batch   20:18:52
Proto 13 -- THE SUPERSTITION -- Ready for use.

%SET-I-INTSET, login interactive limit = 64, current interactive value = 0
   SYSTEM      job terminated at 26-JUN-1984 20:18:59.26
%%%%%%%%%%  OPCOM  26-JUN-1984  20:19:05.73  %%%%%%%%%%
Message from user DECNET
DECnet starting
```

Example 6–14: Performing an Orderly System Shutdown on a VAX-11/780 System

```
Username:  SYSTEM
Password:

        Welcome to VAX/VMS version V5.0 on node SUPER
    Last interactive login on Saturday, 23-JUN-1988 08:20
    Last non-interactive login on Saturday, 23-JUN-1988 01:24

            You have 7 new Mail messages.

$ @SYS$SYSTEM:SHUTDOWN

        SHUTDOWN -- Perform an Orderly System Shutdown

How many minutes until final shutdown [0]:  10
Reason for shutdown [Standalone]:  PREVENTATIVE MAINTENANCE
Do you want to spin down the disk volumes [NO]?  YES
Do you want to invoke the site-specific shutdown procedure [YES]? YES
Should an automatic system reboot be performed [NO]?  NO
When will the system be rebooted [later]:  AT 10:00
Shutdown options (enter as a comma-separated list):
 SAVE_FEEDBACK       Saves feedback data for AUTOGEN calculations
 REMOVE_NODE         Remaining nodes in the cluster should adjust quorum
 CLUSTER_SHUTDOWN    Entire cluster is shutting down
 REBOOT_CHECK        Check existence of basic system files
Shutdown options [NONE]:  NONE

SHUTDOWN message from user SYSTEM at _OPA0:     08:23:36
SUPER will shut down in 10 minutes; back up AT 10:00.
PREVENTATIVE MAINTENANCE

44 terminals have been notified.
%SHUTDOWN-I-OPERATOR, This terminal is now an operator's console.
%%%%%%%%%%  OPCOM  23-JUN-1988 08:28:44.11  %%%%%%%%%%
Operator status for operator _OPA0:
CENTRAL, PRINTER, TAPES, DISKS, DEVICES, CARDS, NETWORK, CLUSTER, SECURITY,
REPLY, SOFTWARE, OPER1, OPER2, OPER3, OPER4, OPER5, OPER6, OPER7, OPER8, OPER9,
OPER10, OPER11, OPER12

%SHUTDOWN-I-DISLOGINS, Interactive logins will now be disabled.
%SET-I-INTSET, login interactive limit = 0, current interactive value = 3
%SHUTDOWN-I-SHUTNET, The DECnet network willn ow be shut down.

SHUTDOWN message from user SYSTEM at _OPA0:  08:29:03
SUPER will shut down in 5 minutes; back up AT 10:00.  Please log out.
PREVENTATIVE MAINTENANCE

3 terminals have been notified.

SHUTDOWN message from user SYSTEM at _OPA0:   08:31:09
SUPER will shut down in 2 minutes; back up AT 10:00.  Please log out.
PREVENTATIVE MAINTENANCE

3 terminals have been notified.
```

Example 6–14: Performing an Orderly System Shutdown on a VAX-11/780 System (Cont.)

```
SHUTDOWN message from user SYSTEM at _OPA0:   08:00:00
SUPER will shut down in 2 minutes; back up AT 10:00.  Please log out.
PREVENTATIVE MAINTENANCE

3 terminals have been notified.
%SHUTDOWN-I-STOPQUEMAN, The queue manager will now be stopped.

SHUTDOWN message from user SYSTEM at _OPA0:   08:01:00
SUPER will shut down in 1 minutes; back up AT 10:00.  Please log out.
PREVENTATIVE MAINTENANCE

3 terminals have been notified.

SHUTDOWN message from user SYSTEM at _OPA0:   08:02:00
SUPER will shut down in 0 minutes; back up AT 10:00.  Please log out.
PREVENTATIVE MAINTENANCE

3 terminals have been notified.
%SHUTDOWN-I=SITESHUT, The site-specific shutdown procedure will now be invoked.
%SHUTDOWN-I-STOPUSER, All user processes will now be stopped.
%SHUTDOWN-I-REMOVE, All installed images willnow be removed.
%%%%%%%%%%  OPCOM  23-JUN-1988  08:34:44.22  %%%%%%%%%%
Message from user DECNET
DECnet shutting down

%SHUTDOWN-I-DISMOUNT, All volumes will now be dismounted.
%SHUTDOWN-I-DISMOUNTDEV, Dismounting device _DRA1:.
%SHUTDOWN-I-DISMOUNTDEV, Dismounting device _DRA2:.
%SHUTDOWN-I-DISMOUNTDEV, Dismounting device _DRA3:.
%%%%%%%%%%  OPCOM  23-JUN-1988 08:35:03.13  %%%%%%%%%%
Message from user SYSTEM
_OPA0:, SUPER shutdown was requested by the operator.

%%%%%%%%%%  OPCOM  23-JUN-1988 08:35:03.69  %%%%%%%%%%
Logfile was closed by operator _OPA0:
Logfile was SYS$SYSROOT:[SYSMGR]OPERATOR.LOG;8

%%%%%%%%%%  OPCOM  23-JUN-1988 08:35:03.82  %%%%%%%%%%
Operator _OPA0:  has been disabled, username SYSTEM

        SYSTEM SHUTDOWN COMPLETE - USE CONSOLE TO HALT SYSTEM

>>>H

        HALTED AT 80008CE8
>>>
```

Table 6–19: Forcing a Shutdown Using CCL Commands

VAX–11/780	VAX–11/750	VAX–11/730
CTRL/P	CTRL/P	CTRL/P
>>>HALT	>>>E/G F	>>>@CRASH
>>>@CRASH	>>>E P	
	>>>E/I 0	
	>>>E/I 1	
	>>>E/I 2	
	>>>E/I 3	
	>>>E/I 4	
	>>>D/G F FFFFFFFF	
	>>>D P 1F0000	
	>>>C	

MODULE 7
INSTALLING AND UPDATING SYSTEM SOFTWARE

INTRODUCTION

The system manager of a VMS system must install and customize the software on the system before users can log in and work. This module focuses on the skills needed to install, upgrade, and add maintenance updates or optional software to your VMS system. Software installation, whether you are installing the operating system or an optional product, involves two basic steps:

- Registering a Product Authorization Key (PAK). This key represents your software license and allows the software to run.
- Installing the software itself. This means copying the software from a distribution volume onto the system disk.

After you install the operating system, customize it using the skills discussed in the other modules in this course. Finally, you should run the User Environment Test Package (UETP) and make backup copies of your system disk and console volume.

OBJECTIVES

- Use the License Management Facility to manage the software license database
- Install VMS system software
- Upgrade VMS system software
- Install optional software and maintenance updates
- Customize and save the contents of the system volume and the console volume
- Run the User Environment Test Package (UETP) to test the hardware and the software

RESOURCES

1. *VMS Installation and Operations* manual for each processor
2. *VMS Release Notes* for latest version of the operating system
3. *VMS License Management Utility Manual*

MANAGING PRODUCT LICENSES USING THE LICENSE MANAGEMENT FACILITY (LMF)

Overview of the LMF

- Most DIGITAL products now require a *license key* to be installed in order to operate VMS V5.0 and later systems. You install the key by copying information from a paper *Product Authorization Key (PAK).*
- These product keys are entered into a cluster–wide database.
- As each node is started up, licenses are loaded into a volatile LMF database in memory.
- Products on any node in the cluster check the volatile database to determine whether a particular use of the product is licensed or not.
- There are two basic types of licenses:
  - — Availability License: Allows the use of a product on a host
  - — Activity License: Allows a specific number of concurrent users to access a product.
- It is also possible to combine the two licenses:
  - — Compound License: Requires a host license and further licenses for each concurrent user.
- Products can also be grouped together under a single license so that a single key will enable the use of several products.

LMF Features and Benefits

- Managed by the manipulation of license keys
- Ability to INCLUDE or EXCLUDE nodes for subcluster licensing
- Administration and tracking
- Concurrent user licensing distributed over the cluster
- Centralized license management
- Allows generous copying; focuses on use
- Spans the VAX family
- Allows combination of license keys
- Supports possible future technologies such as distribution of software and documentation on CD–ROM

Components of the LMF

- License Unit Requirements Tables (LURT)
  - — LURTs contain the number of units required for each VAX model number.
  - — The LURTs are specified by a letter table code in the PAK in either the availability field, activity field, or both.
- The License Management utility

  Format:

```
$ LICENSE  subcommand  parameter
```

  The License Management utility (LICENSE) is a DCL–level interface to the License Management Facility (LMF) on the VMS operating system.
- The LICENSE database
  - — SYS$COMMON:[SYSEXE]LMF$LICENSE.LDB is the default location.
  - — If there is more than one system disk in the cluster, logical name LMF$LICENSE should point to a single database for the cluster containing all product licenses for all cluster hosts.
  - — If there is no disk accessible to all hosts, separate databases should be maintained *identically* (see *VMS License Management Utility Manual*).
  - — The LICENSE database is where all key information is kept. The database is managed by the LICENSE utility.
- SYS$UPDATE:VMSLICENSE.COM
  - — A command procedure to assist you in registering your keys in the LICENSE database.

License Units and License Unit Requirement Tables

License Unit

- Basic element by which DIGITAL specifies how much product use a license authorizes.
- Each license key specifies the number of units purchased with the corresponding license.
- Each processor has a series of license unit requirements, with higher performance processors requiring more license units.
- Products query the LMF to determine if there are sufficient units available to activate the product. If the number of units available are greater or equal to the units required, then the product may be used.

License Unit Requirement Tables (LURTs)

- LURTs are lists of the unit requirements for each supported processor for particular types of products.
- A LURT for availability licenses specifies the number of units required to load a product's license key on a specific VAX system for unlimited use.

  Example: Unlimited use of a product RETRIEVE on a VAX 6230 might require 1500 license units, while unlimited use of RETRIEVE on a MicroVAX II might require only 200 units.
- A LURT for activity licenses specifies the number of units required for each concurrent user on a specific VAX system model using the product.
  - — The user can be anywhere in the cluster.
  - — Different hosts in the cluster might require different numbers of units per user.
  - — As each user activates the product, the number of available units decreases according to the processor they are accessing the product from.
  - — If there is not enough available units for another user on a particular host, that user gets an error message.
  - — When a user stops using the product, the units they were using again become available.
  - — Activity licenses are useful when you expect to use a product infrequently and by a small number of users on a large machine.
  - — Additional activity licenses can be purchased to add more users.
  - — Products licensed by activity licenses can also be licensed by availability licenses at your discretion if product usage increases beyond the point where an activity license is cost effective.

Table 7–1: Values for an Activity License LURT

VAX Model	Number of license units required per user
VAX 8650	75
VAX 8350	60
VAXstation 2000	30

Product Usage with Activity License in Cluster

A product QUERY has an activity license for 1000 units in a cluster consisting of a VAX 8650, VAX 8350, and three VAXstation 2000 systems. For this discussion, the activity license LURT will have values as shown above.

As each user anywhere in the cluster accesses QUERY, the number of units required per user by that host will be subtracted from the number of units available to all other hosts in the cluster.

If there were eight users on the VAX 8650, that would require 8x75=600 units. If there were five users on the VAX 8350 at the same time, that would require an additional 5x60=300 units. This leaves only 100 units. This would allow each of the three VAXstation 2000s to have one user of QUERY, that would require 3x1x30=90 units. The next user anywhere in the cluster would receive an error message because the 10 units remaining are not enough for any users.

Note that the three VAXstation 2000 users could not all have accessed QUERY from either the VAX 8650 or the VAX 8350. With 100 units remaining, only one additional user could have used the product from either VAX system.

LICENSE UTILITY

LICENSE Subcommand Overview

Table 7–2: LICENSE Subcommands

Command	Definition
AMEND	Changes a license currently in the LICENSE database.
CANCEL	Specifies a new termination date for a product currently in the LICENSE database.
CREATE	Creates a LICENSE database with no license records.
DISABLE	Disables an existing license in the LICENSE database.
ENABLE	Enables an existing license in the LICENSE database so it can be activated with the LICENSE LOAD command.
ISSUE	Produces a replica of a PAK that is sent to a file or displayed on your terminal (the default). *This command disables the license in the database.*
LIST	Displays information from the LICENSE database about the specified license or licenses.
LOAD	Activates a license or licenses making them available for product authorization for the current node.
MODIFY	Modifies a license for system management and license sharing purposes.
REGISTER	Adds a new license to the LICENSE database.
START	Sets up an in–memory table for your system, and activates all licenses that are registered and enabled in the LICENSE database.
UNLOAD	Deactivates a license, making the product unavailable from the current node.

AMEND

- Changes a license currently in the LICENSE database.
- Use the LICENSE AMEND command only when the software vendor provides amendment information. (Currently, DIGITAL does not issue amendments to licenses.)

Format: LICENSE AMEND product-name

Example:

```
$ LICENSE AMEND GIZMO /PRODUCER=DEC /ISSUER=DEC -
_$ /AUTHORIZATION=USA4321 -
_$ /CHECKSUM=1-GEAD-OODA-HIDN-PLAC /VERSION=9.3
```

This command amends the license for the DIGITAL software product named GIZMO. Entering this command upgrades an existing GIZMO license to Version 9.3. The producer name, issuer name, authorization number, and checksum number are typed exactly as they appear in the amendment information.

CANCEL

Specifies a new termination date for a product currently in the LICENSE database. You must use the /TERMINATION=date qualifier.

Format: LICENSE CANCEL /TERMINATION=date product-name

Example:

```
$ LICENSE CANCEL/AUTHORIZATION=USA1776 -
_$ /TERMINATION=04-JUL-1989 VAX-VMS
```

Unless an earlier termination date exists, this command sets a new cancellation date of July 4, 1989 for the license on VMS.

Note that the product name is entered with a hyphen (-) character as it was specified on the PAK.

CREATE

Creates a LICENSE database with no license records. LMF creates a default LICENSE database in SYS$COMMON:[SYSEXE]LMF$LICENSE.LDB. Therefore, you need not specify this command.

Format: LICENSE CREATE

Example:

```
$  LICENSE CREATE/DATABASE=SYS$MANAGER:LMF$LICENSE.LDB
```

DISABLE

Disables an existing license in the LICENSE database. A disabled license cannot be activated to authorize product use. The LICENSE DISABLE command does not immediately affect any active processes. Active processes are affected only if you enter a LICENSE UNLOAD command or if the system shuts down.

Format: LICENSE DISABLE product-name

Example:

```
$ LICENSE DISABLE VAXset /PRODUCER=DEC
```

This command disables the license for VAXset, produced by Digital Equipment Corporation (DEC). Because no database is specified, LMF uses the default database.

ENABLE

Enables an existing license in the LICENSE database so that it can be activated with the LICENSE LOAD command. This command cancels the effect of the LICENSE DISABLE command. Newly registered licenses are enabled by default.

Format: LICENSE ENABLE product-name

Example:

```
$  LICENSE ENABLE VAXSET /PRODUCER=DEC
```

ISSUE

Produces a replica of a PAK that is sent to a file or displayed on your terminal (the default). If the terms and conditions of your license contract allow it, you can then enter this PAK replica in the LICENSE database of another processor. When you enter a LICENSE ISSUE command, LMF disables the license in the current LICENSE database and marks the license ISSUED. To enable a license that has been marked ISSUED, enter the LICENSE ENABLE command.

Format: LICENSE ISSUE product-name

Example:

```
$  LICENSE ISSUE /OUTPUT=SYS$MANAGER:FORTRAN.PAK -
_$     /PRODUCER=DEC FORTRAN
```

LIST

Displays information from the LICENSE database about the specified license or licenses. You can control the form, content, and location of information displayed with the /BRIEF, /FULL, /HISTORY, and /OUTPUT qualifiers.

Format: LICENSE LIST [product-name]

Example

```
$  LICENSE LIST
Press return to exit, use arrow keys to scroll.

Product Name:                Producer
-----------------------------------

 FORTRAN                     DEC
 COBOL                       DEC
 PASCAL                      DEC
[End of List]
```

History records are written by every command that changes any fields in a license record. These commands are AMEND, CANCEL, ENABLE, DISABLE, ISSUE, and MODIFY.

Example 7–1: $ LICENSE/LIST/FULL/HISTORY Output

```
$  LICENSE LIST /FULL /HISTORY FORTRAN
Press return to exit, use arrow keys to scroll.

License Management Facility

LICENSE database File:        ART::SYS$COMMON:[SYSEXE]LMF$LICENSE.LDB
Created on:                   17-AUG-1988
Created by user:              MONET
LMF Version:                  V1.0

------------------------------------------------------
Issuer:                       DEC
Authorization:                USA-2468
Product Name:                 FORTRAN
Producer:                     DEC
Units:                        2000
Date:                         (none)
Version:                      V4.7
Termination Date:             10-DEC-1990
Availability:                 F (Layered Products)
Activity:                     0
Options:
Hardware ID:

Revision Level:               2
Status:                       Active
Command:                      AMEND
Modified by user:             DEGAS
Modified on:                  19-AUG-1989 14:32:23.41
Include:                      ART

-----------------------------------
Issuer:                       DEC
Authorization:                USA-2468
Product Name:                 FORTRAN
Producer:                     DEC
Units:                        2000
Modified Units:               9999
Date:                         (none)
Version:                      V4.5
Termination Date:             20-AUG-1989
Availability:                 F (Layered Products)
Activity:                     0
Options:                      MOD_UNITS
Hardware ID:

Revision Level:               1
Status:                       History
Command:                      AMEND
Modified by user:             DEGAS
Modified on:                  29-AUG-1988 12:12:27.33
[End of List]
```

LOAD

Activates a license or licenses making them available for product authorization for the current node. The product license or licenses must currently exist and be active in the LICENSE database. If the license is already loaded, the LMF returns an error message and makes no changes.

To use this command you need the privileges to change mode to kernel (CMKRNL), to create system logical names (SYSNAM), and system privileges (SYSPRV).

Format: LICENSE LOAD [product-name]

Example:

```
$  LICENSE UNLOAD FORTRAN
$  LICENSE MODIFY/INCLUDE=MUSIC FORTRAN
$  LICENSE LOAD FORTRAN
```

Whenever a load is successful, the utility displays a message showing the number of license units loaded. You can also use the DCL command SHOW LICENSE to see what licenses are loaded.

MODIFY

Modifies a license for system management and license-sharing purposes. The LICENSE MODIFY command changes data in the LICENSE database immediately, but the modifications do not affect a running system until you activate the modified license with a LICENSE LOAD command.

Before using this command, refer to your software license agreement to determine whether the modifications you want to make are valid under the terms of the license.

Format: LICENSE MODIFY product-name

Example:

```
$  LICENSE MODIFY /EXCLUDE=(DANCE,THEATR) -
_$  /COMMENT="Modified to exclude nodes DANCE & THEATR 10/23/89" -
_$   FORTRAN
```

REGISTER

Adds a new license to the LICENSE database. A Product Authorization Key (PAK) provides the product name and information you need to register the license. You must enter all information provided by your PAK exactly as specified.

Often the command procedure SYS$UPDATE:VMSLICENSE.COM is used to register a new product license. This provides a prompt–based interface to the LICENSE REGISTER command.

Format: LICENSE REGISTER product–name

Example:

```
 $ LICENSE REGISTER FORTRAN  /ISSUER=DEC  /AUTHORIZATION=USA-10 -
 _$ /PRODUCER=DEC /UNITS=400 /VERSION=4.6 -
 _$ /AVAILABILITY=F /CHECKSUM=1-HIDN-INDA-COMP-DAHH


$  LICENSE REGISTER  DVNETRTG /ISSUER=DEC -
_$ /AUTHORIZATION=USA-15 -
_$ /PRODUCER=DEC /UNITS=1000 /VERSION=4.0 -
_$ /AVAILABILITY=E /CHECKSUM=1-COOD-AGON-EFIC-HING
```

START

Sets up the License Unit Requirement Table (LURT) for your system, and activates all licenses that are registered and enabled in the LICENSE database. Because the VMS operating system issues a LICENSE START command during system startup, you should need this command only if startup fails.

To use this command, you need the privileges to change mode to kernel (CMKRNL), to create system logical names (SYSNAM), and system privileges (SYSPRV).

Format: LICENSE START

Example:

```
$  LICENSE START
```

This command sets up the LURT for your system and activates all the licenses that are registered and enabled in the LICENSE database.

UNLOAD

Deactivates a license, making the product unavailable from the current node. The product license or licenses must be registered in the LICENSE database and must have been previously activated with an interactive or automatic LICENSE LOAD command. The LICENSE UNLOAD command has no effect on active processes.

To use this command you need the privileges to change mode to kernel (CMKRNL), to create system logical names (SYSNAM), and system privileges (SYSPRV).

Format: LICENSE UNLOAD product-name

Example:

```
$  LICENSE UNLOAD/PRODUCER=DEC FORTRAN
```

Messages

To ensure that LMF messages are displayed through the Operator's Communication Facility (OPCOM), you must define the logical name LMF$DISPLAY_OPCOM_MESSAGE as follows:

```
$ DEFINE/EXEC/SYSTEM LMF$DISPLAY_OPCOM_MESSAGE TRUE
```

Example 7–2: Product Authorization Key

```
 - - - - - - -
| | | | | | | |          LICENSE SOFTWARE PRODUCT          | DOCUMENT ISSUE DATE |
|d|i|g|i|t|a|l|          PRODUCT AUTHORIZATION KEY        |      8-MAR-1988     |
| | | | | | | |                                            |_____________________|
 - - - - - - -
Digital Equipment Corporation
Maynard, MA.

 ___________________________________
| LICENSE ADMINISTRATION LOCATION:  |  ORDERED BY: Newton Scientific Inst.
|                                   |              Mr. Isaac Newton
| Digital Equipment Corporation     |              128 Main St.
| Maynard, Massachusetts            |              Newton, MA 03300
|                                   |
|                                   |
|                                   |
 -----------------------------------
*******************************************************************************
 PAK ID:
                    Issuer: DEC
      Authorization Number: USA000877

 PRODUCT ID:
               Product Name: FORTRAN
                   Producer: DEC

 NUMBER OF UNITS:
            Number of units: 5000

 KEY LEVEL:
                    Version: 5.0
      Product Release Date:

 KEY TERMINATION DATE:
      Key Termination Date:

 RATING:
   Availability Table Code: F
       Activity Table Code:

 MISCELLANEOUS:
                Key Options: MOD_UNITS
              Product Token:
                Hardware-Id:
                   Checksum: 1-LDFA-PGEA-AIEA-ICAL
*******************************************************************************
```

Example 7–3: VMSLICENSE Session

```
$ SET DEFAULT SYS$UPDATE
$ @VMSLICENSE.COM

   VMS License Management Utility Options:

       1.  Register a Product Authorization Key
       2.  Amend an existing Product Authorization Key
       3.  Exit this procedure

Select option: 1

* Do you have your Product Authorization Key? Y

   When prompted for input, enter data from corresponding fields on
   your Product Authorization Key (PAK) or Product Authorization
   Amendment (PAAM).

   Some prompts display a default reply (shown in brackets).  To use
   the default, press the RETURN key.  To replace default data, enter
   the new data to be used.  To cancel the use of default data without
   entering new data, enter the backslash (\) character.

   You will have the opportunity to review and correct your responses
   before actually registering this license.  If you wish to exit from
   giving responses, do so by typing CTRL/Z.


   PAK ID:

* Issuer [DEC]:

* Authorization Number: USA000877


   PRODUCT ID:

* Product Name: FORTRAN

* Producer [DEC]:


   NUMBER OF UNITS:

* Number of Units: 5000


   KEY LEVEL:

* Version (vv.uu): 5.0


   KEY TERMINATION DATE:

* Key Termination Date (dd-mmm-yyyy):


   RATING:

* Availability Table Code: f

* Activity Table Code:
```

Example 7–3: VMSLICENSE Session Cont.

```
    MISCELLANEOUS:
* Key Options: mod_units
* Product Token:
* Hardware ID:
* Checksum: 1-LDFA-PGEA-AIEA-ICAL


               LMF Database:  SYS$COMMON:[SYSEXE]LMF$LICENSE.LDB
                     Issuer:  DEC
       Authorization Number:  USA000877
               Product Name:  FORTRAN
                   Producer:  DEC
             Number of Units:  5000
                    Version:  5.0
       Product Release Date:
       Key Termination Date:
    Availability Table Code:  F
        Activity Table Code:
                Key Options:  MOD_UNITS
              Product Token:
                Hardware ID:
                   Checksum:  1-LDFA-PGEA-AIEA-ICAL
* Is this information correct? Y

        DEC FORTRAN has been registered.
* Do you want to LOAD this license on this system [YES]?
%LICENSE-I-LOADED, DEC FORTRAN was successfully loaded with 5000 units
* Do you want to register another PAK? n

    VMS License Management Utility Options:

        1.  Register a Product Authorization Key
        2.  Amend an existing Product Authorization Key
        3.  Exit this procedure

Select option: 3
$
```

LICENSE MANAGEMENT FACILITY AND LICENSE AGREEMENTS

The terms and conditions of your product contract determine your legal use of software. The LMF is a management tool that can help you comply with your license agreement. However, the LMF offers options for many kinds of license agreements. Using some of these options can be illegal for your specific contract. You must read your contract carefully to determine which LMF options you can use legally.

For more information on your legal responsibilities, contact your DIGITAL representatives.

SOFTWARE INSTALLATION

Installing a Major Release of the Operating System

- Instructions for installing the system are in *VMS Installation and Operations* manuals.
- There is one manual for each family of systems, for example:
  - *VMS Installation and Operations: VAX 8820, VAX 8830, VAX 8840*
  - *VMS Installation and Operations: VAX 6200 Series*
  - *VMS Installation and Operations: VAX 8200, VAX 8250, VAX 8300, VAX 8350*
  - *VMS Installation and Operations: VAX-11/780, VAX-11/785*

Basic Steps in Installation

1. Build a VMS system volume on the new disk.
   a. If system device is removable, load scratch system volume in system device.
   b. Use standalone BACKUP to copy portion of VMS from distribution medium to disk, to form kernel VMS system.
   c. Start kernel VMS system.
   d. BACKUP utility automatically copies remainder of VMS from distribution medium to scratch volume (during this step, you may need to answer some questions).
   e. Complete VMS system should exist on new system disk volume.
2. Boot system using new system disk.
3. Use SYS$UPDATE:VMSINSTAL.COM to install the mandatory update.
4. Use the LICENSE utility to register the VMS license.

NOTE

In a new installation, you do not need to use the SYS$UPDATE:VMSINSTAL.COM to install updates or the LICENSE utility step. These operations are performed automatically.

5. Customize the new system volume and console volume for your site.

   - Use SYS$UPDATE:VMSTAILOR.EXE to remove unwanted files from the system volume.
   - Use AUTHORIZE to add UAF records, proxy records, and rights database information to the UAF.
   - Use SYSMAN to set up disk quotas on data volumes.
   - Modify site–specific startup command procedure files.
   - Decompress the system libraries.
   - If necessary, change default boot file on console volume, following instructions in the appropriate *VMS Installation and Operations* manual.
     - — On some systems, you use console commands.
     - — On other systems, you use SYS$UPDATE:SETDEFBOO.COM. boot file.
   - Run the User Environment Test Package (UETP).
   - If necessary, create a spare console volume, following instructions in the appropriate *VMS Installation and Operations* manual.
   - Use standalone BACKUP to make a copy of the system volume.

Upgrading the Operating System

- Perform an upgrade to convert an existing system to a new major release of VMS, leaving all user files intact.
- The upgrade procedure is documented in a single place, generally the VMS *Release Notes* for that version.
- Before upgrading, use standalone BACKUP to back up the system disk.
- Following the directions in the release notes, use SYS$UPDATE:VMSINSTAL.COM to perform the upgrade. The upgrade procedure:
  - — Purges and deletes some VMS files to make room for the upgrade
  - — Transfers VMS files from the distribution medium to the system disk
  - — Merges the old system files and the new system files
  - — Cleans up files and structures used only during the upgrade
- After the upgrade:
  - — Use SYS$UPDATE:VMSINSTAL to install the mandatory update.
  - — Run UETP to test installation.
  - — Use BACKUP to make a copy of the new system volume.
  - — If the console volume was modified during the upgrade procedure, make a spare copy of the console volume.

Installing Maintenance Updates

- Two situations in which you update the VMS operating system:
  - — After an installation or upgrade, you must apply a mandatory update.
  - — Some releases of the VMS operating system do not require a full upgrade, merely an update.
- The update procedure is documented in the VMS *Release Notes* for that version.
- Before the update, use standalone BACKUP to back up the system disk.
- Following the directions in the release notes, use SYS$UPDATE:VMSINSTAL.COM to perform the update. The update procedure:
  - — Applies patches to some system files
  - — Replaces some system files
- After the update:
  - — Run UETP to test system integrity.
  - — Use BACKUP to make a copy of the new system volume.

NOTE

Maintenance updates MUST be installed in the proper sequence.

Running UETP

- The User Environment Test Package (UETP) tests:
  - — All standard peripheral devices
  - — Various commands and operating system functions
  - — The system's multiuser capability
  - — DECnet–VAX software
- Instructions for running UETP are in the *VMS Installation and Operations* manual for your processor.
- Basic steps in using UETP:
  - — Log in with user name SYSTEST.
  - — Prepare devices for testing.
  - — Execute UETP.
    1. $ @UETP
    2. At UETP prompts, press RETURN key to request default action.
    3. By default, UETP runs all possible tests.
  - — If UETP does not complete successfully, see the chapter on UETP in your *VMS Installation and Operations* manual.
  - — If you are unable to correct an error by following directions in the manual, get help from your DIGITAL Field Service representative.

Installing Optional (Layered) Products

- Install optional products after installing VMS and running UETP.
- Use SYS$UPDATE:VMSLICENSE.COM to register Product Authorization Key (PAK) for the product.
- Consult optional product's software installation guide for additional steps and further instructions.
  - — Some optional software is part of the VMS kit and requires no further installation. These *System Integrated Products* include:

    DECnet–VAX
    VAXcluster Software
    VAX Volume Shadowing
    RMS Journaling

  - — For most products, you use SYS$UPDATE:VMSINSTAL to copy the product software to the system disk.

Example 7–4: Using the VMSINSTAL Command Procedure to Install FORTRAN

```
$ @SYS$UPDATE:VMSINSTAL FORT050 MUA0:

        VAX/VMS Software Product Installation Procedure V5.0

It is 12-MAY-1988 at 12:11.
Enter a question mark (?) at any time for help.

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

The following products will be processed:
  FORT V5.0

        Beginning installation of FORT V5.0 at 12:12

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

        Product:      FORTRAN
        Producer:     DEC
        Version:      5.0
        Release Date: 1-FEB-1988

* Does this product have an authorization key registered and loaded? YES

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

* Do you want to install the VAX FORTRAN compiler [YES]? RETURN

        This kit contains an Installation Verification Procedure
        (IVP) to verify the correct installation of the VAX
        FORTRAN compiler.  The IVP will be left in:
             SYS$SYSROOT:[SYSTEST.FORTRAN]FORTRAN$IVP.COM.
        After the installation is complete, you can invoke the
        command file at any time to reverify that VAX FORTRAN is
        installed and working correctly.
```

Example 7–4: Using the VMSINSTAL Command Procedure to Install FORTRAN Cont.

```
* Do you want to run the IVP after the installation [YES]? RETURN

        This kit contains a file summarizing the new features,
        changes, restrictions, and compatibility issues in this
        release of VAX FORTRAN. The name of this file is
        FORT050.RELEASE_NOTES and it is placed in SYS$HELP:.

        This file contains information valuable to VAX FORTRAN
        programmers. Please inform your user community of this
        file's existence.

        This kit also contains the file, FORTRANFIXES050.MEM,
        summarizing the bug fixes made to the VAX FORTRAN
        compiler since its last release. This file will be
        placed in SYS$HELP:.

* Would you like a copy of it printed now? [NO]? RETURN

        In order to build your FORSYSDEF library, this procedure
        requires at least 6000 blocks of available disk space,
        most of which is used for temporary work files. The
        FORSYSDEF library itself will take approximately 1900
        blocks of disk space upon completion of this procedure
        and will be placed in your SYS$LIBRARY area.

                                NOTE

        Before installing FORSYSDEF, be sure to have read the
        appropriate section of the installation guide which
        addresses the question of when a new FORSYSDEF should
        be built.

* Do you want to build a new FORSYSDEF.TLB [NO]? YES

* Do you want to install FORTRAN help [YES]?

        This kit contains two separate HELP files, a large
        version (approximately 600 blocks) including information
        on FORTRAN language features, and a smaller version
        (approximately 100 blocks) describing only the FORTRAN
        command.
```

Example 7–4: Using the VMSINSTAL Command Procedure to Install FORTRAN Cont.

```
* Do you want to install the larger version of FORTRAN help [YES]?

        All questions  regarding the installation of VAX FORTRAN
        have now been asked.  Depending upon your configuration,
        time estimates for the installation(s) have been provided.

        VAX FORTRAN compiler:   3  to  60 minutes
        FORSYSDEF.TLB:         10 to 120 minutes
        FORTRAN HELP:           1  to  15 minutes

%VMSINSTAL-I-SYSDIR, This product creates system disk directory  VMI$ROOT:[SYSTEST.FORTRAN].
%CREATE-I-EXISTS, VMI$ROOT:[SYSTEST.FORTRAN] already exists

        +-----------------------------------------------------+
        |       Installing the VAX FORTRAN V5 Compiler        |
        +-----------------------------------------------------+

        +-----------------------------------------------------+
        |               Installing FORSYSDEF.TLB              |
        +-----------------------------------------------------+

        +-----------------------------------------------------+
        |              Installing VAX FORTRAN HELP            |
        +-----------------------------------------------------+

        Your  VMS  system  will now be updated  to  include  the
        following new and modified file(s):

        SYS$HELP:FORT050.RELEASE_NOTES  [new]
        SYS$SYSTEM:FORTRAN.EXE          [new]
        SYS$MESSAGE:FORTERR1.EXE        [new]
        SYS$MESSAGE:FORTERR2.EXE        [new]
        SYS$LIBRARY:FORTV5CLD.CLD       [new]
        SYS$LIBRARY:DCLTABLES.EXE       [modified]
        SYS$HELP:FORTRANFIXES050.MEM    [new]
        SYS$TEST:FORTRAN$IVP.COM        [new]
        SYS$LIBRARY:FORSYSDEF.TLB       [new]
        SYS$TEST:FORSYSDEFTST.COM       [new]
        SYS$HELP:HELPLIB.HLB            [modified]
%VMSINSTAL-I-MOVEFILES, Files will now be moved to their target directories...

        +-----------------------------------------------------+
        |           Verification Command Procedure for        |
        |                      VAX FORTRAN                    |
        +-----------------------------------------------------+

VAX FORTRAN V5.0-34 TEST PASSED

        VMSINSTAL procedure done at 12:36
```

CUSTOMIZING AND BACKING UP THE CONSOLE VOLUME

- If your system has a console volume, you must customize the boot environment to match the system configuration. If necessary, modify:
  - — The default bootstrap command procedure
  - — Any other command procedure on console volume
- Back up the console volume
  - — For systems with a removable console volume, (for example, VAX 8250, VAX 8350, VAX 8650, VAX–11/780, VAX–11/750, and VAX–11/730) use SYS$UPDATE:CONSCOPY.COM to copy the console volume. Follow directions in your *VMS Installation and Operations* manual.

    First use CONSCOPY.COM to copy the console files to a directory on the VAX system.

    Then use CONSCOPY.COM to copy the files from the directory on the VAX system to a new console volume.
  - — For systems with a fixed console disk, (for example, VAX 8810, VAX 8820, VAX 8830, and VAX 8840) the tape cartridges or diskettes shipped with your system serve as a backup copy of the console volume. Follow directions in the *Console User's Guide* should you need to reinstall the console files.

Example 7–5: Saving the Contents of the Console Volume

```
$ @sys$update:conscopy

    S Y S $ U P D A T E : C O N S C O P Y . C O M

        Save or restore a VMS console medium.


Which CPU kit do you want to build?

   78X  includes 11/780, 11/782 and 11/785
   8600 includes 8650
   8200 includes 8300

[8600, 8200, 78X, 750 or 730, default 8200]: RETURN

        A SAVE operation involves copying the  console medium to
        an RT-11  virtual volume, which  is a Files-11 file that
        is an image of the RT-11 console volume.

        A RESTORE operation involves copying the entire contents
        of a virtual volume to a console medium.

Do you want to SAVE or RESTORE your console RX50?: SAVE

Enter file name of virtual disk [default SYS$DISK:CONSOLE.DSK]: RETURN

Do you want log messages as files are copied? [Y/N, default Yes]: YES

Enter console device drive (DDCU:): CSA1:

Put your console RX50 into drive _CSA1:,
and type <RETURN> when ready: RETURN
%MOUNT-I-MOUNTED, VAX console  mounted on _CSA1:

%EXCHANGE-S-INITIALIZED, the RT-11 volume WORK3:[MATTHEWS]CONSOLE.DSK;1 has been initialized
%EXCHANGE-S-MOUNTVIR, the RT-11 virtual volume CONSOLE_SAVE: has been mounted
        using the file WORK3:[MATTHEWS]CONSOLE.DSK;1
%EXCHANGE-S-MOUNTED, the RT-11 volume _CSA1: has been mounted
%EXCHANGE-S-COPIED, _CSA1:FG81.VF0 copied to CONSOLE_SAVE:FG81.VF0, 0 blocks
%EXCHANGE-S-COPIED, _CSA1:BOOT58.EXE copied to CONSOLE_SAVE:BOOT58.EXE, 23 blocks
%EXCHANGE-S-COPIED, _CSA1:VMB.EXE copied to CONSOLE_SAVE:VMB.EXE, 55 blocks
%EXCHANGE-S-COPIED, _CSA1:CI780.BIN copied to CONSOLE_SAVE:CI780.BIN, 36 blocks
%EXCHANGE-S-COPIED, _CSA1:CIBCA.BIN copied to CONSOLE_SAVE:CIBCA.BIN, 65 blocks
%EXCHANGE-S-COPIED, _CSA1:CSABOO.CMD copied to CONSOLE_SAVE:CSABOO.CMD, 1 block
%EXCHANGE-S-COPIED, _CSA1:CIBOO.CMD copied to CONSOLE_SAVE:CIBOO.CMD, 2 blocks
%EXCHANGE-S-COPIED, _CSA1:KDBBOO.CMD copied to CONSOLE_SAVE:KDBBOO.CMD, 2 blocks
%EXCHANGE-S-COPIED, _CSA1:CONSOL.HLP copied to CONSOLE_SAVE:CONSOL.HLP, 5 blocks
%EXCHANGE-S-COPIED, _CSA1:CI78V5.BIN copied to CONSOLE_SAVE:CI78V5.BIN, 36 blocks
%EXCHANGE-S-COPIED, _CSA1:CI78V6.BIN copied to CONSOLE_SAVE:CI78V6.BIN, 36 blocks
%EXCHANGE-S-COPIED, _CSA1:DEFBOO.CMD copied to CONSOLE_SAVE:DEFBOO.CMD, 1 block

Directory of RT-11 volume CONSOLE_SAVE:      11-MAY-1988 12:13
                    using WORK3:[MATTHEWS]CONSOLE.DSK;1
Volume ID: 8200 console   Volume Owner: MATTHEWS       System ID: DECVMSEXCHNG

  FG81.VF0      0 11-May-1988     BOOT58.EXE   23 11-May-1988
  VMB.EXE      55 11-May-1988     CI780.BIN    36 11-May-1988
  CIBCA.BIN    65 11-May-1988     CSABOO.CMD    1 11-May-1988
  CIBOO.CMD     2 11-May-1988     KDBBOO.CMD    2 11-May-1988
  CONSOL.HLP    5 11-May-1988     CI78V5.BIN   36 11-May-1988
  CI78V6.BIN   36 11-May-1988     DEFBOO.CMD    1 11-May-1988

Total of 12 files, 262 blocks.  Free space 526 blocks, largest 526.
```

Example 7–5: Saving the Contents of the Console Volume Cont.

```
        The SAVE of your console RX50 is complete.

        Since the console device is now accessible to users, the
        console should be remounted write-locked for protection.
        Please  place  your  console medium  back in the console
        device, and  it will be  mounted  with  /SYSTEM /NOWRITE
        protection to prevent access by unprivileged users.

Type literal(<)RETURN> when ready: RETURN
%MOUNT-I-MOUNTED, VAX console  mounted on _CSA1:

$ @SYS$UPDATE:CONSCOPY

    S Y S $ U P D A T E : C O N S C O P Y . C O M

        Save or restore a VMS console medium.


Which CPU kit do you want to build?

   78X  includes 11/780, 11/782 and 11/785
   8600 includes 8650
   8200 includes 8300

[8600, 8200, 78X, 750 or 730, default 8200]: RETURN

        A SAVE operation involves copying the  console medium to
        an RT-11  virtual volume, which  is a Files-11 file that
        is an image of the RT-11 console volume.

        A RESTORE operation involves copying the entire contents
        of a virtual volume to a console medium.

Do you want to SAVE or RESTORE your console RX50?: RESTORE

Enter file name of virtual disk [default SYS$DISK:CONSOLE.DSK]: RETURN

Do you want log messages as files are copied? [Y/N, default Yes]: YES

Enter console device drive (DDCU:): CSA1:

Put your console RX50 into drive _CSA1:,
and type <RETURN> when ready: RETURN
%MOUNT-I-MOUNTED, VAX console  mounted on _CSA1:

%EXCHANGE-S-INITIALIZED, the RT-11 volume _CSA1: has been initialized
%EXCHANGE-S-MOUNTVIR, the RT-11 virtual volume CONSOLE_RESTORE: has been mounted
          using the file WORK3:[MATTHEWS]CONSOLE.DSK;1
%EXCHANGE-S-MOUNTED, the RT-11 volume _CSA1: has been mounted
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:FG81.VF0 copied to _CSA1:FG81.VF0, 0 blocks
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:BOOT58.EXE copied to _CSA1:BOOT58.EXE, 23 blocks
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:VMB.EXE copied to _CSA1:VMB.EXE, 55 blocks
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:CI780.BIN copied to _CSA1:CI780.BIN, 36 blocks
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:CIBCA.BIN copied to _CSA1:CIBCA.BIN, 65 blocks
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:CSAB00.CMD copied to _CSA1:CSAB00.CMD, 1 block
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:CIB00.CMD copied to _CSA1:CIB00.CMD, 2 blocks
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:KDBB00.CMD copied to _CSA1:KDBB00.CMD, 2 blocks
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:CONSOL.HLP copied to _CSA1:CONSOL.HLP, 5 blocks
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:CI78V5.BIN copied to _CSA1:CI78V5.BIN, 36 blocks
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:CI78V6.BIN copied to _CSA1:CI78V6.BIN, 36 blocks
%EXCHANGE-S-COPIED, CONSOLE_RESTORE:DEFB00.CMD copied to _CSA1:DEFB00.CMD, 1 block
```

Example 7–5: Saving the Contents of the Console Volume Cont.

```
Directory of RT-11 volume _CSA1:      11-MAY-1988 12:15
Volume ID: 8200 console   Volume Owner: MATTHEWS       System ID: DECVMSEXCHNG

  FG81.VF0      0 11-May-1988     BOOT58.EXE   23 11-May-1988
  VMB.EXE      55 11-May-1988     CI780.BIN    36 11-May-1988
  CIBCA.BIN    65 11-May-1988     CSABOO.CMD    1 11-May-1988
  CIBOO.CMD     2 11-May-1988     KDBBOO.CMD    2 11-May-1988
  CONSOL.HLP    5 11-May-1988     CI78V5.BIN   36 11-May-1988
  CI78V6.BIN   36 11-May-1988     DEFBOO.CMD    1 11-May-1988

Total of 12 files, 262 blocks.  Free space 526 blocks, largest 526.

        The RESTORE of your console RX50 is complete.

        Since the console device is now accessible to users, the
        console should be remounted write-locked for protection.
        Please  place  your  console medium  back in the console
        device, and  it will be  mounted  with  /SYSTEM /NOWRITE
        protection to prevent access by unprivileged users.

Type <RETURN>
 when ready: RETURN
%MOUNT-I-MOUNTED, VAX console  mounted on _CSA1:
```

CREATING A NEW SYSTEM DISK FROM ANOTHER SYSTEM DISK

- Ways to create a new VMS system disk:
  - — Perform a VMS installation onto the disk.
  - — Use SYS$UPDATE:VMSKITBLD.COM to copy VMS files from an existing system disk to the new disk.
  - — Use SYS$MANAGER:CLUSTER_CONFIG.COM to copy VMS.

Example 7–6: Building a System Disk with VMSKITBLD.COM

```
$ @SYS$UPDATE:VMSKITBLD

$!  Copyright (c) 1988 Digital Equipment Corporation.  All rights reserved.
Operation [BUILD,ADD,COPY,COMMON]? BUILD
Enter mounted SOURCE disk name (ddcu:): SYS$SYSDEVICE:
Enter SOURCE top level system directory [default = SYS0]: VMS$COMMON
Enter TARGET disk name (ddcu:): KNIFE$DUA0:
Enter the TARGET disk's label [default = VAXVMSRL5]:
Enter TARGET disk top level system directory [default = SYS0]:
%DCL-I-ALLOC, _KNIFE$DUA0: allocated
The target disk will be initialized.
Target disk, _KNIFE$DUA0:, ready to be initialized? (Y/N): Y
Target disk, _KNIFE$DUA0:, has been initialized.
%MOUNT-I-MOUNTED, VAXVMSRL5     mounted on _KNIFE$DUA0:
Creating directories for system.
Creating SYSGEN files.
%SYSGEN-I-CREATED, _KNIFE$DUA0:<SYS0.SYSEXE>SWAPFILE.SYS;1 created
%SYSGEN-I-CREATED, _KNIFE$DUA0:<SYS0.SYSEXE>PAGEFILE.SYS;1 created
%SYSGEN-I-CREATED, _KNIFE$DUA0:<SYS0.SYSEXE>SYSDUMP.DMP;1 created
Copying files from source disk.
Writing a boot block.
$IF ((EXIT_STATUS .EQ. 1) .AND. (.NOT. KIT)) THEN SAY "System disk complete."
System disk complete.
$EXIT EXIT_STATUS
```

STANDALONE BACKUP

- Use standalone BACKUP, not on-line BACKUP, to back up the system disk.

  1. Shut down the system.
  2. Boot the system using standalone BACKUP kit on console volume or system volume. Generally, booting from the system volume is faster.
  3. Use standalone BACKUP to copy the system volume to disk or tape.

Creating a Standalone BACKUP Kit

- On console media
  - SYS$UPDATE:STABACKIT.COM
  - Requires at least:

    Five RX01 floppy diskettes
    Four RX50 floppy diskettes
    One TK50 tape cartridge

- On system volume
  - SYS$UPDATE:STABACKIT.COM
  - Procedure copies kit into root directory [SYSE.]

Example 7–7: Creating a Standalone BACKUP Kit on the System Disk

```
$ @SYS$UPDATE:STABACKIT
STABACKIT-I-SYMDEL, all global symbols deleted

Enter the name of the device on which to build the kit: SYS$SYSDEVICE:

Sysgen parameters for standalone VMS have been placed in file
                SYS$SYSROOT:<SYSUPD>VAXVMSSYS.PAR-TEMP-2080023D;1
%COPY-S-COPIED, SYS$COMMON:<SYSEXE>STASYSGEN.EXE;1 copied to
_BROWNY$DJA0:<SYSE.SYSEXE>SYSINIT.EXE;1 (92 blocks)
%COPY-S-COPIED, SYS$COMMON:<SYSEXE>SYSBOOT.EXE;2 copied to
_BROWNY$DJA0:<SYSE.SYSEXE>SYSBOOT.EXE;1 (117 blocks)
%COPY-S-COPIED, SYS$SYSROOT:<SYSUPD>VAXVMSSYS.PAR-TEMP-2080023D;1 copied to
_BROWNY$DJA0:<SYSE.SYSEXE>VAXVMSSYS.PAR;1 (15 blocks)
%SET-I-ENTERED, BROWNY$DJA0:<SYS0>SYSCOMMON.DIR;1 entered as
_BROWNY$DJA0:<SYSE>SYSCOMMON.DIR;1
%COPY-S-COPIED, SYS$COMMON:<SYS$LDR>DUDRIVER.EXE;1 copied to
_BROWNY$DJA0:<SYSE.SYS$LDR>DSDRIVER.EXE;1 (35 blocks)
%COPY-S-COPIED, SYS$COMMON:<SYSEXE>STABACKUP.EXE;1 copied to
_BROWNY$DJA0:<SYSE.SYSEXE>STANDALON.EXE;1 (429 blocks)
%DELETE-I-FILDEL, SYS$SYSROOT:<SYSUPD>VAXVMSSYS.PAR-TEMP-2080023D;1 deleted
(16 blocks)

Ending time   4-MAY-1988 16:49:07.89
Starting time 4-MAY-1988 16:48:20.82

The kit is complete.

$
```

Using a Standalone BACKUP Kit

- To boot standalone BACKUP from the system disk, see your *VMS Installation and Operations* manual.
  - — Some systems already have a console command procedure that boots standalone BACKUP.
  - — On other systems, you must create a new command procedure to boot the system from root [SYSE.]
- To boot standalone BACKUP from the console device:
  - — If you are booting from a removable console volume, load the first standalone BACKUP volume into the console drive.
  - — Enter the appropriate boot command. See the *VMS Installation and Operations* manual.
  - — If you are booting from a removable console volume, standalone BACKUP directs loading operation. Load appropriate volumes into the console drive as requested.
- After standalone BACKUP is loaded, the DCL prompt appears on console terminal.
- At the DCL prompt, enter BACKUP commands.

SUMMARY

- Software installation involves two steps:
  - — Installing the Product Authorization Key (PAK)
  - — Transferring the software from the distribution medium to the system disk
- The VMS License Management Facility (LMF) provides software for managing PAKs. The LMF consists of:
  - — License Unit Requirements Tables (LURTs), which specify the number of units required for each type of VAX system
  - — The LICENSE database LMF$LICENSE.LDB, which contains information about installed keys
  - — The License Management utility, which maintains the LICENSE database
  - — SYS$UPDATE:VMSLICENSE.COM, which helps you use the License Management utility
- To run VMS on a new system, you perform an installation. Instructions for installing the system are in *VMS Installation and Operations* manuals. There is a different manual for each series of processors.
- To install a new major release of VMS on an existing system, you perform an upgrade. The upgrade procedure is documented in the *VMS Release Notes* for that version.
- To install a minor release or a mandatory update of VMS, you perform an update. The update procedure is documented in the *VMS Release Notes* for that version.
- After installing, upgrading, or updating, run the User Environment Test Package (UETP) to test system integrity.
- If your system has a console volume, you must customize boot procedures for your system configuration. Make a backup copy of the console volume after changing it.
- To install VMS upgrades or updates or optional (layered) software products, use the command procedure SYS$UPDATE:VMSINSTAL.
  - — For a layered product, there may be additional steps in installation. These steps are documented in the product's *Installation Guide*.
- Use VMSKITBLD.COM if you need to create a new VMS system disk on an existing system.
- Use standalone BACKUP to back up the system disk. You can use SYS$UPDATE:STABACKIT.COM to build standalone backup on the system disk or on console media.

MODULE 8
MAINTAINING SYSTEM INTEGRITY

INTRODUCTION

The system manager is responsible for coordinating both the maintenance and security of the hardware and software on a system to establish system integrity. Users should be able to modify data files, access devices, and run long programs without fear that their work will be corrupted or will need to be repeated.

The manager has several tools to maintain system integrity. Some tools analyze the error data that the system collects, others establish security of data files and programs. The manager must define the level of integrity required, choose tools to maintain that level of integrity, and use these tools effectively.

When a problem occurs on a system, the manager must try to determine whether a software error or a hardware error has caused it. To determine this, the manager should analyze recent error logs and run simple diagnostics. Some problems require more intensive diagnostics that a field service specialist should run.

The system manager is not expected to identify and solve every VMS system problem that may arise. Rather, he or she should be able to:

- Investigate problems and collect information
- Use system documentation to research and solve elementary problems
- Pass more difficult problems on to DIGITAL Field Service personnel or to experienced DIGITAL Software Services personnel

This module covers the tools that the system manager needs to maintain the hardware and software, establish security, and diagnose problems.

OBJECTIVES

To collect information useful for troubleshooting, the system manager should:

- Perform regular hardware and software maintenance checks
- Ensure system security
  - — For physical security of the computer
  - — In the software environment on the computer
- Be able to use the AUTHORIZE utility
- Read system dump files using the System Dump Analyzer

RESOURCES

1. *Guide to Maintaining a VMS System*
2. *Guide to VMS System Security*
3. *VMS System Generation Utility Manual*
4. *VMS DCL Dictionary*
5. *VMS Error Log Utility Manual*
6. *VMS System Dump Analyzer Utility Manual*
7. *DIGITAL Site Preparation Guide*

HARDWARE MAINTENANCE

- Hardware maintenance involves
  - — Care of devices (CPU, disks, terminals)
  - — Care of media (tapes, disks, line printer paper)

Handling and Storing Media

- Devices and media are sensitive to:
  - — Rough treatment
  - — Dust
  - — Sudden impact
  - — Electrical noise
- Handling media
  - — Only trained personnel should handle and load media
  - — Store unused media in closed metal cabinets
  - — Media containing valuable information should be stored off site in a fireproof vault

Cleaning Media

- Cleaning and maintenance tasks usually performed by customers include
  - — Cleaning heads, rollers, and vacuum area of tape drive
  - — Cleaning outside cover of disk drive
  - — Changing printer ribbons
  - — Wiping terminal screen
- DIGITAL Field Service tasks include
  - — Aligning heads of disk and tape drives
  - — Cleaning interiors of disk and tape drives
  - — Fixing print heads
- Unsafe media is "rejected" at cleaning time
  - — Back up all media before having it cleaned

Maintaining the Environment

- System manager should set up and control
  - — Temperature
  - — Humidity
  - — Power
  - — Flooring
  - — Dust
  - — Electrical noise
  - — Lighting
  - — Fire and safety precautions

Preventive Maintenance

- Regular hardware maintenance is vital
  - — Disk and tape head misalignment can result in loss of data
- DIGITAL Field Service can satisfy maintenance needs
- *Self-Maintenance Handbook* contains useful maintenance instructions

SOFTWARE MAINTENANCE

- Software maintenance includes:
  - — Maintenance of system files
  - — Maintenance of user programs and data files
- To perform maintenance:
  - — Install updates and upgrades correctly
  - — Run appropriate utilities regularly to observe contents of certain system files
  - — Back up system and user files regularly

SYSTEM SECURITY

Physical Security

- Security issues
  - — Physical security of computer
    - Access to console terminal
    - Availability of dial–up lines
  - — Media storage

Software Security

- For minimum security, system manager should
  - Keep SYSUAF.DAT and NETPROXY.DAT up to date
  - Use flags and hourly restrictions with AUTHORIZE
  - Insist on nontrivial passwords
  - Not publicize dial-up numbers
  - Not permit WORLD access to SYSUAF.LIS
  - Encourage use of file protection codes and ACLs
  - Request secondary passwords
  - Restrict user privileges
  - Create ACCOUNTING reports to check system usage
  - Label disks and tapes in systematic manner
- For additional security, manager can include
  - Erase-on-delete and erase-on-allocate for files
  - Login security
  - Break-in detection
  - Security auditing

Erase-on-Delete and Erase-on-Allocate

- Erase-on-delete (EOD) refers to activity of file system when files are deleted
  - — File system overwrites blocks with zeros
  - — Blocks unavailable for allocation until overwritten
  - — Users implement EOD on file-by-file basis
  - — Managers implement EOD on volume-by-volume basis
- Erase-on-allocate (EOA, also known as highwater marking) refers to activity of file system when files are created or extended
  - — File system overwrites blocks with zeros (or pattern of your choice) before allocating them
  - — Set automatically for each volume at initialization
  - — Disable highwater marking for less security-sensitive volumes

```
$ SET VOLUME/NOHIGHWATER_MARKING
```

Table 8–1: Setting Erase-on-Delete for a File or Volume

Command/Qualifier and Example	Comments
`$ SET FILE/ERASE_ON_DELETE -` `_$ file-name`	Sets characteristic of file so file system performs an EOD when you delete it.
`$ SET FILE/ERASE_ON_DELETE -` `_$ GOVERNMENT_SECRETS.DAT`	
`$ DELETE/ERASE_ON_DELETE -` `_$ file-name`	Tells the file system to perform an EOD as you delete the specified file or files.
`$ DELETE/ERASE_ON_DELETE -` `_$ MY_SECRETS.DAT`	
`$ PURGE/ERASE_ON_DELETE -` `_$ file-name`	Tells the file system to perform an EOD on each file it deletes during the purge.
`$ PURGE/ERASE_ON_DELETE -` `_$ COMPANY_SECRETS.*`	
`$ INITIALIZE/ERASE_ON_DELETE -` `_$ volume-name`	Tells the file system to perform an EOD for every file on this volume that users delete.
`$ INITIALIZE/ERASE_ON_DELETE -` `_$ SECURE_VOLUME:`	
`$ SET VOLUME/ERASE_ON_DELETE -` `_$ SECURE_VOLUME:`	Modifies the volume's characteristic so the file system now performs an EOD on every file deleted from it.

Login Security

AUTHORIZE and UAF Record Controls

- Use the AUTHORIZE utility to set
  - — Expiration date on UAF records
  - — Minimum password length
  - — Expiration date for password
  - — Passwords that must be changed when user logs in for the first time
- Can define secondary password for accounts
- Can have VMS automatically provide a list of randomly generated passwords to select from

Table 8–2: Defining User Passwords

Commands	Comments
`$ RUN AUTHORIZE` `UAF> MODIFY SMITH /PASSWORD=MARY`	Modifies any user password with the AUTHORIZE utility.
`$ RUN AUTHORIZE` `UAF> MODIFY SMITH /PASSWORD=("",SECOND)`	To add a secondary password to a UAF record, use the **/PASSWORD** qualifier with the **MODIFY** command. This command does not affect the current value of the primary password if you properly specify a null first password string.
`$ SET PASSWORD`	Modifies your own primary password.
`$ SET PASSWORD/SECONDARY`	Modifies your own secondary password.
`$ SET PASSWORD/GENERATE`	You can request or require (using the **/FLAG=GENPWD** qualifier in AUTHORIZE) that VMS generate a random list of passwords to choose from. If you enter the **/GENERATE** qualifier and receive a list, you must choose a password from that list, or request another list. (You do not need the **/GENERATE** qualifier if password generation has been set in your UAF record.)

- Can require system password for terminals
- System password kept as a special UAF record
- System password can be set
  - — At DCL: **$ SET PASSWORD/SYSTEM**
  - — Within AUTHORIZE: **MODIFY /SYSTEM_PASSWORD**

Table 8–3: Defining a System Password for a Terminal

Step/Function	Comments

Step 1:
 	
`$ SET PASSWORD/SYSTEM`
`Old password:`
`New password:`
`Verification:`
`$`
	Enter the old password in response to the first prompt. Then enter the new password in response to the next two prompts. None of the passwords are echoed while being entered. This command requires SECURITY and CMKRNL privileges.

Step 2:
 	
`$ SET TERMINAL -`
`_$ /SYSPWD /PERMANENT -`
`_$ TXA2`
	Set the specified terminal to require the system password in order to be used. To log in to this terminal a user must press the RETURN key, enter the system password (no prompt is given). If successful, the user continues the normal login procedure when the "Username:" prompt appears.

Users and managers should be aware of password requirements for accounts and terminals.

Table 8–4: Using Passwords

Situation	Example	Comments
Normal account: requires one password	RETURN Username: SMITH Password:	The system does not echo the password. Note that you can define the password to be null. If you do that, you do not receive any password prompt.
Accounting requiring primary and secondary passwords	RETURN Username: SMITH Password: Password:	Passwords are not echoed. Typically, one person knows the primary password, but another knows the secondary one. Therefore, both must be present whenever this account is used.
Terminal requiring a system password	<RETURN> Username: SMITH Password:	You do not receive a prompt for the system password. To avoid excessive reports of broken terminals, make users aware of the ones requiring a system password, because unless they enter it successfully, the system will not display the "Username:" prompt.

Protecting Terminals and Other Nonshareable Devices

- Terminals (nonshareable devices) have owner UIC and protection code that determine access
  - — To control device ownership and protection

```
$ SET PROTECTION/DEVICE
```

  - — To place access control list (ACL) protection on terminals and tape drives

```
$ SET DEVICE/ACL
```

Table 8–5: Establishing Ownership and Protection of Terminals and Other Nonshareable Devices

Operation	Format of $ SET PROTECTION/DEVICE Command and Examples (Requires OPER privilege)	Comments
Establishing protection ownership	`$ SET PROTECTION=code/DEVICE-` `_$/OWNER_UIC=[uic] device`	By default, all terminals have the owner specified by the system parameter TTY_OWNER and the protection specified by the system parameter TTY_PROT.
Allowing all users access to a device	`$ SET PROTECTION/DEVICE device` `$ SET PROTECTION/DEVICE TTA3:`	The default value for TTY_PROT allows all users access to the device. If you do not specify a protection code with this command, you are assigning the default protection to the device.
Establishing system users as the owner	`$ SET PROTECTION = (S:R,O:R,G,W)-` `_$ /DEVICE/OWNER=[1,4] TTA3:`	Only system users may allocate this terminal from a program. By specifying that Group and World users have no access, you protect the terminal against password collection programs run by users.

- Set secure server characteristic on terminals to protect them from password–collecting programs

  ```
  $ SET TERMINAL/PERMANENT/SECURE_SERVER/DISCONNECT TTC1:
  ```

  — Ensures that only VMS login program receives user name and password

  — Press BREAK key before RETURN key to receive "Username:" prompt

  — Cannot set secure server characteristic on terminals with AUTOBAUD characteristic

Break-In Detection at Login

- System records information about login failures in suspect lists
  - — User name suspect list
  - — Terminal name suspect list
  - — Node name suspect list
- Several SYSGEN parameters provide break-in control
  - — LGI category parameters

Table 8-6: SYSGEN Parameters for Break-In Detection

Parameter	Comments
LGI_BRK_DISUSER	Once an intruder has been detected, VMS sets the DISUSER flag in the account's UAF record (if the parameter is set to 1). Manual intervention by the system manager is necessary to reactivate the account. Use this feature with caution.
LGI_BRK_LIM	Break-in limit defining the total number of consecutive login failures allowed within a reasonable time limit before a SUSPECT becomes an INTRUDER.
LGI_BRK_TERM	Controls the association of terminals and user names for counting failures. By default, VMS sets this parameter to 1 so that terminals and user names are tracked together. If you use terminal servers, then you might want to set this parameter to 0 (only track user names), since a LAT port (on the VAX side) is generally not a useful indication of the actual terminal being used.
LGI_BRK_TMO	Timeout factor expressed in seconds (a delta time). Used in conjunction with LGI_BRK_LIM to decide if a SUSPECT is an INTRUDER. The larger this value, the more secure your system.
LGI_HID_TIM	Time factor expressed in seconds. System uses this value in an equation to determine the time interval during which an INTRUDER is subject to evasive action. The time interval calculated using LGI_HID_ TIM is different for each instance, so you never know exactly how long an INTRUDER will be evaded.
LGI_RETRY_LIM	Limits the number of times a user can retry the login procedure when coming in through dial-up lines. The default value is 3.
LGI_RETRY_TMO	The number of seconds allowed between login attempts on dial-up lines. The default value is 20 seconds. This means the user must properly log in within 20 seconds of a failed attempt, or the system will hang up the line.

Suspects can gain access to the system by entering the correct user name and password.

Example 8–1: Suspect Logs in Successfully

```
Username: JONES
Password:
User authorization failure
Username: JONES
Password:
User authorization failure
Username: JONES
Password:

          Welcome to node TIDE running V5.0
Last interactive login on Saturday, 11-JUN-1988  08:59
        2 failures since last successful login

$
```

- System creates intruder lists when login failures exceed break–in limit set by system manager
  - — Separate intruder lists for user name, terminal name, and node name
  - — Intruders subject to evasive action
- Intruders CANNOT gain access to system even if they enter correct user name and password

Example 8–2: Suspect Becomes an Intruder

```
<RETURN>

           THIS IS NODE TIDE

Username:  WEBSTER
Password:
User authorization failure
Username:  WEBSTER
Password:
User authorization failure
Username:  WEBSTER
Password:
User authorization failure

          This is node TIDE

Username:  WEBSTER
Password:
User authorization failure
Username:  WEBSTER
Password:
User authorization failure
Username:  WEBSTER
Password:
User authorization failure
```

UIC and ACL Protection

VMS Protection Using UICs

- In assigning UICs to users, determine the extent to which user processes need to:
  - — Share access to files, volumes, and devices
  - — Communicate interactively with one other
  - — Affect or control one other
- For greater interaction, place users in the same UIC group
- For greater protection, place users in different groups

Table 8–7: Dividing Users into Groups

User Situation	Course of Action	Result
Users must allow some users access to their files and structures, but deny access to others on the system.	Divide users into groups along project or departmental lines. Assign a different group UIC number to each group. Leave room between each assigned number for additions later.	Users within a group can access each other's files with the access set for GROUP in the protection code of the accessed file. Users outside a group will be able to get access to files belonging to a group member with the access defined for WORLD only.
Users do not need to protect files against access by each other, but you must protect files from their access. (System files usually have a group number from 0–10.)	Assign all users a UIC of [200,200] or some other UIC.*	All users own all user files and have OWNER access to all user files. Users cannot access system files.
Users need not protect files, but should divide them into groups for accounting and observation purposes.	Divide users into groups.† Either:	
	1. Modify the system parameter RMS–FILEPROT to set the default protection assigned to all new files so WORLD has complete access. 2. You can collect accounting information on separate users (according to their UICs) but since all files created allow all users complete access, the files are not really protected. In this case, newly created system files are not protected either.	1. Create a LOGIN.COM file for each user and include the **SET PROTECTION/DEFAULT=(W:RWED)** command in it. Do not include this command in your LOGIN.COM file. 2. You can collect accounting information on separate users (according to their UICs). Users can access all files on the system except your files and system files.

† Do not assign a group number reserved for system users (typically 0–10). Modify the MAXSYSGROUP parameter to set the upper limit of the group number for system users as needed.

Table 8–8: Interaction Between Processes in Same Group

Interaction Available	Comment
Can share files, volumes, and devices, yet deny access to processes in other groups	Accomplished by setting the protection codes of the files, volumes, and devices properly.
Can communicate with each other by means of structures that processes in other groups cannot access	For example: • Group logical names • Group mailbox logical names • Group global section names • Common event flag clusters
Can affect and control other processes in group	Requires GROUP privilege.

VMS Protection Using ACLs

- Optional layer of protection for files
- Use to grant access to files for specific users rather than groups of users
- Based on *identifiers*
  - — Users can hold one or more identifiers
  - — Files can specify access rights for holders of various identifiers
- Record access information for files in *access control list (ACL)*
- Define identifiers and who holds them in *rights database*
- When user logs in, VMS creates *access rights list* for user, consisting of identifiers held
- When user attempts to access files, VMS compares access rights list of user with access control list of file
  - — If no ACL on file, VMS determines access rights according to UIC of user and VMS protection code on file
  - — If ACL does not allow access, SYSTEM and OWNER users can still gain the type of access allowed them in the VMS protection code
  - — If user does not hold any identifiers listed in file's ACL, VMS determines access rights according to UIC of user and VMS protection code on file

Use the **SET RIGHTS_LIST** command to enable/disable identifiers within rights lists.

- Used for modifying process and system rights lists
- Controls access to objects:
  - — Per–process
  - — System–wide
- Must use either **/ENABLE** or **/DISABLE**
- Must have CMKRNL and SYSNAM privileges to modify system rights list

Table 8–9: Some Qualifiers for the SET RIGHTS_LIST Command

Qualifier	Description
`/ENABLE`	Adds the specified identifier to the process or system rights list.
`/DISABLE`	Removes the specified identifier from the process or system rights list.
`/SYSTEM`	Specifies that the system rights list is to be modified. Requires SYSNAM and CMKRNL privileges.
`/PROCESS=process-name`	Specifies the name of the process whose rights list is to be modified. Requires CMKRNL privilege.
`/IDENTIFICATION=pid`	Specifies the process identification value (PID) of the process whose rights list is to be modified. Requires CMKRNL privilege.

Table 8–10: Terminology Used to Discuss VMS Access Control

Term	Definition
Object	Anything that exists in the system to which protection can be applied, such as files and devices.
Agent	Anything that generates a request for access to an object. The system checks such requests to determine whether to grant access. Examples include a process or a device.
Identifier	Unique name used to identify a group to which one or more users belong. An identifier is internally represented as a unique, 32–bit number. It is externally represented as an alphanumeric name. Both the system and the AUTHORIZE utility define identifiers and store them in SYS$SYSTEM:RIGHTSLIST.DAT.
Access Rights List	List of identifiers belonging to a particular user. When you log in, the identifiers you hold in the rights database (including your UIC) are copied into a rights list that is part of your process. The rights list is the structure VMS uses to perform all protection checks.
Access Control List	Collection of ACEs (Access Control Entries) that determine what access is to be allowed to an agent for an object. Some aspects of ACLs are: • An ACL can contain one or more ACEs • An object may have only one ACL • For files, ACLs may or may not be propagated from a previous version of the file, or from the parent directory
Access Control Entry	Entry in an Access Control List. Each entry has three parts: the identifier name, the options, and the access allowed.
Holders	If an identifier is in the access rights list for a user, the user is a "holder" of that identifier. If a user holding several identifiers attempts to gain access to a file protected by an ACL, the system compares the user's identifiers to the ACEs in the ACL, and determines access according to the first match found.
Access Allowed	Recorded after the ACCESS keyword in an ACE. Defines the type of access allowed to an agent holding the identifier specified in the ACE. Types of access include: Read, Write, Execute, Delete, and None.
Options	Recorded after the OPTIONS keyword in an ACE. Specifies the characteristics of an ACL.

Table 8–11: ACL– and UIC–Based Protection

File Name and Owner UIC	Type of Protection	Access Allowed
FILE.DAT [200,011]	UIC–based protection: (S:RWED,O:RWED,G:RWE,W)	System and Owner have RWED access. Group users have RWE access. World users have no access.
PROGRAM.FOR [200,011]	UIC–based protection: (S:RWED,O:RWED,G:RWE,W) ACL allows users holding the identifier name SMITH to read the file.	System and Owner have RWED access. Group users have RWE access. Users holding the identifier name SMITH have READ access only (even if they are in the Group category). Other users have no access.
JUNK.DAT [200,011]	UIC–based protection: (S:RWED,O:RWED,G:RWE,W) ACL does not allow users holding the identifier name SMITH to have any access to the file.	System and Owner users have RWED access (even those holding the identifier name SMITH). Group users have RWE access except those holding the identifier name SMITH. World users have no access.

System-Defined and User-Defined Identifiers

- System creates default identifiers in rights database
  - — LOCAL
  - — DIAL-UP
  - — REMOTE
  - — INTERACTIVE
  - — NETWORK
  - — BATCH
- Use default identifiers in ACL to define access rights for users

```
(IDENTIFIER=MATH+LOCAL, OPTIONS=NONE, ACCESS=READ+WRITE)
```

- System creates one identifier for each UIC
- Stores identifier in rights database

Example 8–3: AUTHORIZE Utility Creates Identifiers Automatically

```
$ SET DEFAULT SYS$SYSTEM
$ SET PROCESS/PRIVILEGE=SYSPRV
$ RUN AUTHORIZE
UAF>
UAF>SHO/ID/VALUE=UIC:[11,204]
unable to complete show command
%SYSTEM-F-NOSUCHID, unknown rights identifier
UAF>
UAF>ADD TOM/UIC=[11,204]/ACCOUNT=VMS/DEVICE=WORK1/DIRECTORY=[TOM]
user record successfully added
identifier TOM value:  [000011,000204] added to RIGHTSLIST.DAT
UAF>
UAF>ADD DICK/UIC=[11,204]/ACCOUNT=VMS/DEVICE=WORK1/DIRECTORY=[DICK]
user record successfully added
unable to add DICK value:  [000011,000204] to RIGHTSLIST.DAT
%SYSTEM-F-DUPIDENT, duplicate identifier
UAF>
UAF>ADD HARRY/UIC=[11,204]/ACCOUNT=VMS/DEVICE=WORK1/DIRECTORY=[HARRY]
user record successfully added
unable to add HARRY value:  [000011,000204] to RIGHTSLIST.DAT
%SYSTEM-F-DUPIDENT, duplicate identifier
UAF>
UAF>SHOW/BRIEF [11,204]
       Owner  Username  UIC           Account Privs     Pri Default Directory

              TOM       [11,204]      VMS     Normal    4   WORK1:[TOM]
              DICK      [11,204]      VMS     Normal    4   WORK2:[DICK]
              HARRY     [11,204]      VMS     Normal    4   WORK1:[HARRY]

UAF>
UAF>SHOW/IDENTIFIER/VALUE=UIC:[11,204]
    Name                            Value             Attributes
    TOM                             [000011,000204]   NORESOURCE
UAF>
UAF>SHOW/IDENTIFIER/USER=TOM
    Name                            Value             Attributes
    TOM                             [000011,000204]   NORESOURCE
UAF>SHOW/IDENTIFIER/USER=HARRY
    Name                            Value             Attributes
    TOM                             [000011,000204]   NORESOURCE
UAF>SHOW/IDENTIFIER/USER=DICK
    Name                            Value             Attributes
    TOM                             [000011,000204]   NORESOURCE
UAF>
UAF>EXIT
system authorization file modified
no modifications made to network authorization file
rights data base modified
$
$
```

- System creates group identifiers
  - — When you add a UAF record with a UIC group number that is different from any previously recorded
  - — Name of group identifier taken from ACCOUNT field in the new UAF record you are adding
  - — Value of group identifier: [group–number,177777]
  - — Sample group identifier value: [000011,177777]
  - — Subsequent additions of UAF records with this group UIC do not affect name or value of existing group identifier
- You create general identifiers
  - — You assign value or system assigns hexadecimal value
  - — Value must be unique

Example 8–4: Displaying Identifiers and Values in the Rights Database

```
$ SET DEFAULT SYS$SYSTEM
$ SET PROCESS/PRIVILEGE=SYSPRV
$ RUN AUTHORIZE
UAF>SHOW/IDENTIFIER *
  Name                            Value           Attributes
  BATCH                           %X80000001      NORESOURCE
  BEYER                           [000011,000005] NORESOURCE
  BIERLY                          [000011,000340] NORESOURCE
  CHAPUT                          [000031,000030] NORESOURCE
  CHERPAS                         [000071,000040] NORESOURCE
  DECNET                          [000240,000240] NORESOURCE
  DIALUP                          %X80000002      NORESOURCE
  FIELD                           [000001,000010] NORESOURCE
  FIELDTEST2                      %X80080011      RESOURCE
  GROUP11                         %X80080021      RESOURCE
  HARBO                           [000031,000007] NORESOURCE
  HYLNKA                          [000011,000120] NORESOURCE
  HOUSSELL                        [000011,000130] NORESOURCE
  INVENTORY                       [000011,000343] NORESOURCE
  INTERACTIVE                     %X80000003      NORESOURCE
  J65                             [000370,177777] NORESOURCE
  JULIAN                          [000021,000150] NORESOURCE
  LOCAL                           %X80000004      NORESOURCE
  MARSH                           [000011,000220] NORESOURCE
  MATH                            %X8001000A      NORESOURCE
  MAYWALT                         [000131,000120] NORESOURCE
  NETMGR                          [000013,000002] NORESOURCE
  NETPRINT                        [000020,000002] NORESOURCE
  NETWORK                         %X80000005      NORESOURCE
  OPERATOR                        [000011,000001] NORESOURCE
  PROJECT                         %X8001000B      NORESOURCE
  PROJMGMT                        %X80010007      NORESOURCE
  REMOTE                          %X80010006      NORESOURCE
  SASSMAN                         [000131,000350] NORESOURCE
  SMITH                           [000011,000200] NORESOURCE
  SUPERVISOR                      %X80080022      NORESOURCE
  SYSTEM                          [000001,000004] NORESOURCE
  SYSTEST                         [000001,000007] NORESOURCE
UAF>SHOW/RIGHTS/USER=[*,*]
  Name                            Value           Attributes
Identifiers held by BEYER :
  FIELDTEST2                      %X80080011      NORESOURCE
Identifiers held by MARSH :
  FIELDTEST2                      %X80080011      NORESOURCE
  GROUP11                         %X80080021      NORESOURCE
Identifiers held by BIERLY :
  PROJMGMT                        %X80010007      NORESOURCE
  FIELDTEST2                      %X80080011      NORESOURCE

UAF>SHOW/IDENTIFIER/FULL FIELDTEST2
  Name                            Value           Attributes
  FIELDTEST2                      %X80080011      RESOURCE
    Holder                          Attributes
    MARSH                           NORESOURCE
    BIERLY                          NORESOURCE
    BEYER                           NORESOURCE
UAF>
```

Example 8–5: Managing the Rights Database

```
$ SET DEFAULT SYS$SYSTEM
$ SET PROCESS/PRIVILEGE=SYSPRV
$ RUN AUTHORIZE
UAF>ADD/IDENTIFIER MATH
identifier MATH value:  %X8001000C added to RIGHTSLIST.DAT
UAF>GRANT/IDENTIFIER MATH BIERLY
identifier MATH granted to BIERLY
UAF>
UAF>GRANT/IDENTIFIER MATH TOM
identifier MATH granted to TOM
UAF>
UAF>SHO/RIGHTS/USER=TOM
 Name                                Value            Attributes
Identifiers held by TOM :
 PROJECT                             %X8001000B       NORESOURCE
 MATH                                %X8001000C       NORESOURCE
UAF>
UAF>SHOW/RIGHTS/USER=BIERLY
  Name                               Value            Attributes
Identifiers held by BIERLY  :
  PROJMGMT                            %X80010007       NORESOURCE
  FIELDTEST2                          %X80080011       NORESOURCE
  MATH                                %X8001000C       NORESOURCE
UAF>
UAF>SHOW TOM

Username:  TOM                                 Owner:
Account:   VMS                                 UIC:     [11,204]   ([VMS,TOM])
CLI:       DCL                                 Tables:
Default:   WORK1:[TOM]
LGICMD:    SYS$MANAGER:SYLOGIN
Login Flags:
Primary days:  Mon Tue Wed Thu Fri Sat Sun
Secondary days:
No access restrictions
Expiration:      (none)      Pwdminimum:   6     Login Fails:   0
Pwdlifetime: 180 00:00       Pwdchange:   23-MAY-1984 22:42
Last Login:      (none)  (interactive),         (none)  (non-interactive)
Maxjobs:        0    Fillm:      20    Bytlm:        12288
Maxacctjobs:    0    Shrfillm:    0    Pbytlm:           0
Maxdetach:      0    BIOlm:       6    JTquota:       1024
Prclm:          2    DIOlm:       6    WSdef:          150
Prio:           4    ASTlm:      20    WSquo:          200
Queprio:        0    TQElm:      10    WSextent:       500
CPU:       (none)    Enqlm:      40    Psflquo:      10000
Authorized Privileges:
  TMPMBX NETMBX
Default Privileges:
  TMPMBX NETMBX
  Name                          Value            Attributes
Identifiers held by TOM :
  PROJECT                       %X8001000B       NORESOURCE
  MATH                          %X8001000C       NORESOURCE
UAF>
```

Table 8–12: AUTHORIZE Commands Used to Manage the Rights Database

Command/Example	Function
`UAF>ADD/IDENTIFIER id-name` `UAF>ADD/IDENTIFIER PAYROLL`	Creates a general identifier. AUTHORIZE gives it a numerical value unless you assign it a different value.
`UAF>SHOW/IDENTIFIER/FULL id-name` `UAF>SHOW/IDENTIFIER/FULL PAYROLL`	Displays the name of the identifier, its value and attributes, and the names of the users that hold it.
`UAF>GRANT/IDENTIFIER id-name user-name` `UAF>GRANT/IDENTIFIER PAYROLL SMITH`	Allows a user to hold an identifier. A user automatically holds the system–defined identifier with the value of his/her UIC. You may create an unlimited number of general identifiers.
`UAF>SHOW/RIGHTS/USER=user-name` `UAF>SHOW/RIGHTS/USER=TOM`	Displays a list of the general identifiers held by TOM, as well as their values and attributes.
`UAF>SHOW/IDENTIFIER/USER=user-name` `UAF>SHOW/IDENTIFIER/USER=SMITH`	Displays the system–defined identifier and its UIC value for the user name specified, such as SMITH.
`UAF>RENAME/IDENTIFIER old-name new-name` `UAF>RENAME/IDENTIFIER PAYROLL NEWPAY`	When you rename an identifier in the rights database, the change is reflected in every ACE that contains the old identifier.
`UAF>REMOVE/IDENTIFIER id-name` `UAF>REMOVE/IDENTIFIER NEWPAY`	Removing an identifier does not remove it from any ACE. You must remove the identifier from any ACE that references it.
`UAF>REVOKE/IDENTIFIER id-name user-name` `UAF>REVOKE/IDENTIFIER PAYROLL TOM`	This is the converse of GRANT/IDENTIFIER.

Restricting User Access to Files

- VMS protection using UICs
- VMS protection using ACLs
  - — Define ACL for a file with ACL editor
  - — VMS uses ACLs to determine user access rights to file

Example 8–6: Defining an ACL for a File

```
$ EDIT/ACL CALCULUS.DAT
```

The screen clears.

```
(IDENTIFIER=MATH,OPTIONS=NONE, ACCESS=NONE)
(IDENTIFIER=PROJECT,OPTIONS=NONE,ACCESS=READ+WRITE)
```

The user exits the ACL editor.

```
$ DIRECTORY/FULL CALCULUS.DAT

Directory WORK1:[BIERLY.MGR.MGRUSERS]

CALCULUS.DAT;1                File ID:  (17807,8,0)
Size:             1/1         Owner:    [VMS,BIERLY]
Created:  11-JUN-1984  10:36   Revised:  11-JUN-1984 10:37 (3)
Expires:   <None specified>    Backup:    <No backup done>
File organization:  Sequential
File attributes:    Allocation:  1,  Extend:  0, Global buffer
count:  0, No version limit
Record format:      Variable length, maximum  52 bytes
Record attributes:  Carriage return carriage control
File protection:    System:RWED, Owner:RWED, Group: RE, World:
Access Cntrl List:  (IDENTIFIER=MATH,ACCESS=NONE)
                    (IDENTIFIER=PROJECT,ACCESS=READ+WRITE)

Total of 1 file, 1/1 block.

$
$
```

Security Auditing

Security auditing allows the system manager to track certain classes of activities.

Table 8–13: Event Classes You Can Audit

Type of Event	Comments
ALL	All possible events are audited.
ACL–requested	If a file has an ACL, and AUDIT is requested in that ACL, then access to the file is audited.
Authorization file access	All modifications to authorization files (SYSUAF.DAT, NET–PROXY.DAT) are audited.
Break–in attempts Detached Dial–up Local Network Remote	You can specify all or a combination of the types of break–in attempts listed.
File Access Success Failure BYPASS SYSPRV GRPPRV READALL	You can track all file access attempts, whether or not they were successful. You can also determine whether a process that gets access to a file has one of the listed privileges
Login/Login Failure/Logout All Batch Detached Dial–up Local Network Remote Subprocess	You can audit logins, login failures, and logouts from any type of process in this list.
Mount/Dismount	You can audit each time a mount or dismount request is made.

- When an audited activity occurs, the system:
  - — Sends an alarm message to the operator terminal enabled to receive security messages

Example 8–7: Security Alarm Message on Console Terminal

```
%%%%%%%%%%  OPCOM 11-JUN-1988 09:32::53.39  %%%%%%%%%%
Security alarm on TIDE / Local interactive breakin detection
         Time:              11-JUN-1988 09:32:53.38
  PID:               20800513
  User Name:         JONES
  Password:          DDD
  Dev Name:          _TTD6:
```

- To enable or disable types of security auditing:

```
$ SET AUDIT
```

- To list classes and types of audit currently enabled:

```
$ SHOW AUDIT
```

Table 8–14: Defining and Listing Audit Classes

Command/Example	Comments
`$ SET AUDIT/ALARM -` `_$ /ENABLE=(class[=keyword])`	You must include the **/ALARM** qualifier when enabling audits. The **/ALARM** qualifier displays audit messages on operator terminals that are enabled to receive security messages.
`$ SET AUDIT/ALARM -` `_$ /ENABLE=AUTHORIZATION`	
`$ SET AUDIT/ALARM -` `_$ /ENABLE=(BREAKIN=DIALUP)`	
`$ SET AUDIT/ALARM -` `_$ /ENABLE=(ACL,MOUNT)`	You can specify more than one class in a single command.
`$ SET AUDIT/ALARM -` `_$ /ENABLE=(BREAKIN=(DIALUP,NETWORK))`	You can specify more than one keyword for a class.
`$ SHOW AUDIT`	You can display a list of the classes and types of events within classes that are currently set for audit.

SYSTEM PROBLEMS

- Two categories of problems
  - — Software

    Operator error (incorrect input)
    Programming error (infinite loops, wrong logic)

  - — Hardware

    Head crash (on disk)
    Memory failure
    Misalignment of disk or tape drives

- Problems can result in
  - — Corruption of data
  - — Loss of data
  - — Loss of system resources
  - — Loss of computation

Software Problems

- When software problems occur, system manager must
  - — Retrain users in proper use of software
  - — Correct errors in programs causing problems
- When problems reoccur despite proper attempts at correction, system manager should
  - — Document problem
  - — Submit Software Performance Report (SPR)
  - — Include the following information in the SPR:

    Listings that reveal state of entire system
    Explanations of how to reproduce the problem
    Information in machine–readable form
    Aspects of system environment not apparent from listings
    Personal clues and analyses

Hardware Problems

- Problems can be
  - — Obvious (head crash)
  - — Intermittent (memory errors)
- Detect obvious problems by checking system periodically
- Tools for detecting intermittent problems
  - — Error logging facility
  - — System failure dump facility
  - — Hardware diagnositcs
  - — User and operator comments
  - — Operator's log file
  - — UETP

Error Logging Facility

- ERRFMT process records certain activities in system error log file
  - — Name of file is SYS$ERRORLOG:ERRLOG.SYS
  - — Records activities such as

    Device errors
    Memory errors
    Volume mounts and dismounts
    System startups

  - — Logs errors for all disk and tape devices
- Can specify which devices will have error logging

```
$ SET DEVICE/ERRORLOGGING
```

- To determine if error logging enabled

```
$ SHOW DEVICE/FULL
```

Table 8–15: Enabling and Disabling Error Logging

Operation	Command Format/Examples (Requires OPER Privilege)
Enabling error logging on a device	`$ SET DEVICE/ERROR_LOGGING device-name` `$ SET DEVICE/ERROR_LOGGING_DBB2:`
Disabling error logging on a device	`$ SET DEVICE/NOERROR_LOGGING device-name` `$ SET DEVICE/NOERROR_LOGGING_DBB2:`

- Use Errorlog Report Formatter (ERF) to examine contents of error log file

```
$ ANALYZE/ERROR_LOG [/qualifiers...]
```

- Can generate different types of reports by selecting various entries

Table 8–16: Selecting Entries for an Error Log Report

Qualifier	Comment
/BEFORE /SINCE /ENTRY	Specifies a range of entries to examine when creating a report. **/BEFORE** and **/SINCE** specify a range within a time interval. **/ENTRY** specifies a range using entry numbers.
/INCLUDE /EXCLUDE	Includes or excludes entries on the basis of device name (MTA0:, DBA1:, etc.), device class (DISKS, TAPES, etc.), and entry types (BUGCHECKS, DEVICE_ERRORS, MEMORY, VOLUME_CHANGES, etc.)

Example 8–8: Portion of Device Error and Volume Changes Report Generated by ANALYZE/ERROR

```
V A X / V M S       SYSTEM ERROR REPORT       COMPILED 11-JUN-1988 10:03
                                                              PAGE 25.
***************************** ENTRY          699. **********************
ERROR SEQUENCE 313.                                LOGGED ON SID 0138700D

DEVICE ERROR, 11-JUN-1988  05:05:48.64  KA780  REV 7 , SERIAL 13,

MASSBUS SUB-SYSTEM, UNIT _SUPER$DRA3:

      CS   DS   ER1   MR1   AS   DA   DT   LA   SN   OF   DC   HR

      0830 51C0 0180 0008 0000 0101 2014 0040 8609 1000 0002 0002
***************************** ENTRY          700. **************************
ERROR SEQUENCE 314.                                LOGGED ON SID 0138700D

DEVICE ERROR, 11-JUN-1988  05:06:39.66  KA780 REV 7, SERIAL 13,

MASSBUS SUB-SYSTEM, UNIT _SUPER$DRA3:

      CS   DS   ER1   MR1   AS   DA   DT   LA   SN   OF   DC   HR

      0830 51C0 0090 000C 0000 011F 2014 07C0 8609 1000 0001 0001
***************************** ENTRY          701. **************************
ERROR SEQUENCE 315.                                LOGGED ON SID 0138700D

DEVICE ERROR, 11-JUN-1988  05:08:23.81  KA780 REV 7, SERIAL 13,

MASSBUS SUB-SYSTEM, UNIT _SUPER$DRA3:

      CS   DS   ER1   MR1   AS   DA   DT   LA   SN   OF   DC   HR

      0830 51C0 0180 0008 0000 0200 2014 0000 8609 1000 00BC 00BC
***************************** ENTRY          703. **************************
ERROR SEQUENCE 318.                                LOGGED ON SID 0138700D

DEVICE ERROR, 11-JUN-1988  05:24:24.06  KA780 REV 7, SERIAL 13,

MASSBUS SUB-SYSTEM, UNIT _SUPER$DRA3:

      CS   DS   ER1   MR1   AS   DA   DT   LA   SN   OF   DC   HR

      0830 51C0 0180 0008 0000 0101 2014 0040 8609 1000 0002 0002
***************************** ENTRY          705. **************************
ERROR SEQUENCE 332.                                LOGGED ON SID 0138700D

DEVICE ERROR, 11-JUN-1988  07:35:28.76  KA780 REV 7, SERIAL 13,

MASSBUS SUB-SYSTEM, UNIT _SUPER$DRA1:

      CS   DS   ER1   MR1   AS   DA   DT   LA   SN   OF   DC   HR

      0838 51C0 0190 0008 0000 0211 2017 0440 5417 1000 0325 0325
ANAL/ERR/OUTPUT=DEV_VOL.DAT/SINCE=9-JUN-1988 00:00:00.00
/INCLUDE=(DEVICE,VOLUME)/BRIEF
```

- ERRFMT distinguishes between two types of errors
  - — Soft errors (recoverable errors)
  - — Hard errors (unrecoverable errors)
- Not all error log entries represent device or volume failure
- Keep logs (in hard–copy form) for reference during preventive maintenance (PM)

System Failure Dump Facility

- System shutdown occurs for two reasons
  - — Operator or manager requests shutdown
  - — System detects unrecoverable error
- System saves copy of memory in dump file called SYS$SYSTEM:SYSDUMP.DMP
- Use System Dump Analyzer (SDA) to examine contents of dump file

```
$ ANALYZE/CRASH
```

Table 8–17: Analyzing a System Dump

Steps in Running SDA Utility	Comments
`$ ANALYZE/CRASH_DMP _$ SYS$SYSTEM:SYSDUMP`	Requires: • Read access to dump file • Read access to system symbol table • Value of system parameter VIRTUALPAGEGCNT at least 1000 pages larger than size of dump file • If executed in site-specific startup file, value of system parameter PQL_DPGFLQUOTA at least 1000 pages larger than size of dump file • If executed elsewhere, value of PGFLQUOTA field in UAF record at least 1000 pages larger than size of dump file
`SDA> command`	Table 8–18 shows typical SDA commands that produce reports about the system dump.
`SDA> EXIT`	Return to DCL (or parent process).

Table 8–18: Types of Reports Generated by the System Dump Analyzer

Subject of Report	Command Format
Reason system failed, processor registers	`SHOW CRASH`
Active processes	`SHOW SUMMARY`
Image each process was running at the time of the failure	`SHOW SUMMARY/IMAGE`
Current process	`SHOW PROCESS/ALL`
Process in balance set	`SHOW PROCESS/ALL[/INDEX=nn] [process-name]`
System device	`SHOW DEVICE device-name`

- System overwrites dump file whenever failure occurs
- Save old dump files and create reports as part of startup procedure

Example 8–9: Copying the Dump File at Startup and Creating Reports

```
$ !
$ ! Copy Dump file and create listing if system just failed
$ !
$ ANALYZE/CRASH_DUMP SYS$SYSTEM:SYSDUMP.DMP
    COPY LIB$DISK:[SYSMGR.CRASH]SAVEDUMP.DMP        ! Save dump file
    SET OUTPUT LIB$DISK:[SYSMGR.CRASH]SYSDUMP.LIS   ! Create listing file
                                                    ! containing the
                                                    ! following information:
                                                    !
    SHOW CRASH                  ! Display crash information
    SHOW STACK                  ! Show current stack
    SHOW SUMMARY                ! List all active processes
    SHOW PROCESS/PCB/PHD/REG    ! Display all current processes
    SHOW SYMBOL/ALL             ! Display systemm symbol table
    EXIT
$ !
$ ! Get rid of old system dump files.
$ PURGE/KEEP=3 LIB$DISK:[SYSMGR.CRASH]*.*
$ PRINT LIB$DISK:[SYSMGR.CRASH]SYSDUMP.LIS
```

Example 8–10: First Page of Report Produced by SDA Command SHOW CRASH

```
VAX/VMS 5.0 -- System Dump Analysis 26-OCT-1988 10:58:27.34    Page 1


                        Table of Contents
                        -----------------

System crash information                ....   2
CPU 00 Processor crash information      ....   3
Current process summary                 ....   4
```

Example 8–11: Second Page of Report Produced by SDA Command SHOW CRASH

```
VAX/VMS 5.0 -- System Dump Analysis      26-OCT-1988 10:58:27.34    Page 2
System crash information


Time of system crash: 23-JUN-1988 20:48:56.10

Version of system: VAX/VMS VERSION V5.0-1

System Version Major ID/Minor ID: 1/0

VAXcluster node: ROW, a VAXstation II/GPX

Crash CPU ID/Primary CPU ID:  00/00

Bitmask of CPUs active/available:  00000001/00000001

CPU bugcheck codes:
 CPU 00 -- INVEXCEPTN, Exception while above ASTDEL or on interrupt stack
```

Example 8–12: Third Page of Report Produced by SDA Command SHOW CRASH

```
VAX/VMS 5.0 -- System Dump Analysis      26-OCT-1988 10:58:27.3    Page 3
CPU 00 Processor crash information

CPU 00 reason for Bugcheck: INVEXCEPTN, Exception while above ASTDEL or on
interrupt stack

Process currently executing on this CPU: SWAPPER

Current IPL: 8  (decimal)

CPU database address:  808BC000

General registers:

 R0  = 00000008   R1  = 00080000   R2  = 00000002   R3  = 8045A060
 R4  = B4001FDC   R5  = 8088CA20   R6  = 00000000   R7  = 00000005
 R8  = 0000018B   R9  = 0000003E   R10 = 80ABA06C   R11 = 00000169
 AP  = 80ABA070   FP  = 00000000   SP  = 801C5520   PC  = 8019B1FE
 PSL = 00080009


Processor registers:

 P0BR   = 8061EE00     SBR    = 006E9800     ASTLVL = 00000004
 P0LR   = 00000E10     SLR    = 00005700     SISR   = 00000000
 P1BR   = 7F979400     PCBB   = 005C5A20     ICCS   = 00000040
 P1LR   = 001FFF7F     SCBB   = 006E1600     SID    = 08000000

 TODR   = 6A0DCC69     SYSTYPE= 01010000

 ISP    = 808BD200
 KSP    = 801C5520
 ESP    = 00000000
 SSP    = 00000000
 USP    = 00000000

               No spinlocks currently owned by CPU 00
```

- Use the following SDA command to
  - — Display a list of users working on the system when it failed
  - — List images that were running at the time of failure

```
SHOW SUMMARY/IMAGE
```

Example 8–13: Fourth Page of Report Produced by SDA Command SHOW SUMMARY/IMAGE

```
VAX/VMS 5.0 -- System Dump Analysis     26-OCT-1988 10:58:27.34    Page 4
Current process summary

 Extended Indx Process name    Username    State   Pri   PCB      PHD     Wkset
-- PID -- ---- --------------- ----------- ------- --- -------- -------- ------
 25400041 0001 SWAPPER                      CUR      16 801C5198 801C5000      0
 25400046 0006 ERRFMT          SYSTEM       HIB      10 8065B470 808BEA00    108
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]ERRFMT.EXE;1
 25400047 0007 CACHE_SERVER    SYSTEM       HIB      16 806545F0 808EDA00    101
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]FILESERV.EXE;1
 25400048 0008 CLUSTER_SERVER  SYSTEM       HIB      14 80654B70 80905200    216
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]CSP.EXE;1
 25400049 0009 OPCOM           SYSTEM       HIB       7 8065F3C0 8091CA00    142
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]OPCOM.EXE;2
 2540004A 000A JOB_CONTROL     SYSTEM       HIB       8 8065F540 80934200    201
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]JOBCTL.EXE;3
 2540004B 000B CONFIGURE       SYSTEM       HIB       8 8067EF80 8094BA00    147
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]CONFIGURE.EXE;3
 2540004D 000D SMISERVER       SYSTEM       HIB      13 8067F100 80963200    534
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]SMISERVER.EXE;3
 25400050 0010 NETACP          DECNET       HIB      14 806892B0 809A9A00    861
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]NETACP.EXE;1
 25400051 0011 VWS$DISPLAYMGR  SYSTEM       LEF       7 8068A920 809C1200    260
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]UISBG.EXE;4
 25400052 0012 VWS$EMULATORS   SYSTEM       HIB       5 8068B200 809D8A00     30
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]VWS$EMULATORS.EXE;4
 25400053 0013 REMACP          SYSTEM       HIB       8 8068F900 80992200     34
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]REMACP.EXE;1
 25400054 0014 AUDIT_SERVER    SYSTEM       HIB       8 806D6240 8097AA00    261
          $11$DUA13:[SYS2.SYSCOMMON.][SYSEXE]AUDIT_SERVER.EXE;88
 25400055 0015 SYSTARTUP       SYSTEM       COM       9 806D64D0 809F0200    296
 25400056 0016 UIS$SYS_FONT_LD SYSTEM       COM       9 8065D190 808D6200    221
```

Hardware Diagnostics

- Isolate and diagnose problem
- Field Service engineer runs diagnostics on site, or from remote location
- Two types of diagnostics
  - — On–line
  - — Standalone

On–Line Diagnostics

- Run on hardware; not essential for the execution of VMS
- DIGITAL Field Service engineer brings diagnostics
- FIELD account must be available to run diagnostics

Standalone Diagnostics

- Use when Field Service must diagnose components essential for the execution of VMS
- Prepare for standalone diagnostics
  - — Shut down system
  - — Remove system disk and other sensitive volumes
  - — Provide access to console terminal and console device

Remote Diagnostics

- Run from DIGITAL Diagnostics Center (DDC)
  - — System must have remote diagnosis port
  - — Set key switch on front panel to "REMOTE"

User and Operator Comments

- Create log book containing
  - — User comments
  - — Backups done
  - — Shift change information
  - — Cleaning done
  - — Unusual events
  - — Warnings and notes for other operators
  - — Work left from previous shift
  - — Specific hardware or software problems

Operator's Log File

- Operator log file (SYS$MANAGER:OPERATOR.LOG) contains
  - — **REQUEST** and **REPLY** communication
  - — Communication between operating system and operator
  - — Messages regarding enabled or disabled operator terminals
  - — Time of day (every half hour)
  - — Off–line messages indicating device taken off–line
- To close a log file and open a new one:

```
$ REPLY/LOG
```

- Operator log is text file (examine using **PRINT** or **TYPE** commands)

UETP

- Run User Environment Test Package (UETP) to test how hardware and software work together
- UETP is not a replacement for diagnostics or system exercisers
- UETP does not identify the cause of a problem

SUMMARY

- System manager must provide proper care for devices, media, and computer room
  - — Customer performs some maintenance activities
  - — DIGITAL Field Service performs sensitive activities
- System manager is responsible for software maintenance
  - — Install updates and upgrades
  - — Ensure integrity of critical system files
  - — Backup system and user volumes regularly
- System manager is also responsible for security
  - — Computer room
  - — Dial–up lines
  - — System files
  - — Media storage
  - — Encourage use of protection codes and ACLs
  - — Auditing critical security situations
- Capabilites for ensuring security on disk volumes
  - — Erase–on–Delete
  - — Erase–on–Allocate (highwater marking)
- Password security affected by proper use of AUTHORIZE
  - — System password
  - — Secondary user passwords
  - — Password expiration
  - — Minimum length passwords
  - — Forced use of system–generated passwords

- Terminal (device) security
  - **SET PROTECTION/DEVICE**
  - **SET TERMINAL/SECURE_SERVER**
- Use UIC and ACL protection to enhance security and sharing among users
- Security auditing
  - **SET AUDIT** to set audit events
  - **SHOW AUDIT** to view audit events
  - Beware of auditing everything (performance penalty)
- Types of system problems
  - Hardware
  - Software
- Error Logging Facility
  - ERRFMT process logs errors to SYS$ERRORLOG:ERRLOG.SYS

    Process initiated at system startup
  - Can selectively set devices for error logging (**SET DEVICE/[NO]ERROR_LOGGING**)
  - Use ERF to examine contents of error log file (**ANALYZE/ERROR_LOG**)

    Can select particular device errors
- Use SDA to examine system failure information (**ANALYZE/CRASH_DUMP**)
- Hardware diagnostic facilities
  - Performed by DIGITAL Field Service personnel
  - Can be performed locally or remotely
- User Environment Test Package (UETP)
  - Tests how hardware and software work together
  - Not a replacement for diagnostics
  - Cannot identify cause of problem

MODULE 9
MONITORING THE SYSTEM

INTRODUCTION

The VMS operating system provides many tools for monitoring system behavior. Using these tools, you can examine the activity of VMS processes and the utilization of system resources. If user demands on system resources become too great, system performance will become unsatisfactory.

The primary factor limiting VMS performance is hardware resources – especially physical memory and disk storage. Hardware resources that are inadequate for the workload generally provide poor performance, regardless of any other efforts to improve performance.

When the system is installed and started up for the first time, it determines the amount of physical memory and disk storage available. It then uses this information to compute appropriate sizes for its data structures and work files. If additional physical resources are added to the system to improve its performance, you can have the VMS system recompute these sizes to take advantage of the new resources.

You are not expected to configure or tune a system to match an application or workload using the information presented in this module. Another course, *VMS System Performance Management*, treats the topic of system performance in greater detail. You are simply expected to use the commands and utilities discussed in this module to monitor the performance of your system. Eventually, as you become familiar with your system and performance management in general, you can use these same tools to help you improve the performance of your system.

OBJECTIVES

- Monitor the system
- Modify system parameters to reflect a change in system resources

RESOURCES

1. *Guide to Setting Up a VMS System*
2. *VMS System Generation Utility Manual*
3. *Guide to Maintaining a VMS System*
4. *Guide to VMS Performance Management*
5. *VMS Monitor Utility Manual*
6. *VMS Accounting Utility Manual*
7. *VMS DCL Dictionary*

MONITORING SYSTEM ACTIVITY

Table 9–1: System, Process, and Device Monitoring

Information Displayed	Command or Utility
Overview of the processes on the system	$ SHOW SYSTEM
Overview of print queues	$ SHOW QUEUE /DEVICES /ALL
Overview of batch queues	$ SHOW QUEUE /BATCH /ALL
Overview of mounted disk and tape volumes	$ SHOW DEVICES/MOUNTED
Demands on system resources	$ MONITOR
Error counts for CPU, memory, and physical devices	$ SHOW ERROR
Interactive users, terminal names, and process IDs	$ SHOW USERS
Information about current activities of a certain process	$ SHOW PROCESS /CONTINUOUS /ID=pid $ SHOW PROCESS /ALL /ID=pid
Information about user limits and privileges	$ RUN SYS$SYSTEM:AUTHORIZE
Information about disk space allowances	$ SHOW QUOTA /USER=uic $ RUN SYS$SYSTEM:SYSMAN
Consumption of resources by processes	$ ACCOUNTING
Information about devices and volumes	$ SHOW DEVICE

Monitoring Active Processes

- List all processes on the system

  $ SHOW SYSTEM

- List information about a single process

  — In same UIC group (requires GROUP privilege)

  $ SHOW PROCESS process–name

  — In any UIC group (requires WORLD privilege)

  $ SHOW PROCESS /ID=pid

- Process scheduling based on software process priority
- Process scheduling states

  — Current process

  — Wait states

  COM (computable) process
  Outswapped process

Example 9–1: Output from the SHOW SYSTEM/FULL Command

```
VAX/VMS V5.0  on node SPIDER 30-APR-1988 16:10:15.13   Uptime    3 00:21:30
  Pid     Process Name    State  Pri      I/O       CPU        Page flts Ph.Mem
21200081 SWAPPER         HIB     16        0    0 00:01:03.47         0       0
         [GROUP1,SYSTEM]
21200086 ERRFMT          HIB      9     3221    0 00:00:38.08        70     111
         [1,6]
21200089 OPCOM           LEF      8     1393    0 00:00:19.59      4844      81
         [GROUP1,SYSTEM]
2120008A JOB_CONTROL     HIB      8    81129    0 00:24:52.39       200     342
         [GROUP1,SYSTEM]
2120008C NETACP          HIB     10    35622    0 00:38:08.93      5965    5640
         [GROUP1,SYSTEM]
2120008E REMACP          HIB      9      467    0 00:00:02.05        78      65
         [GROUP1,OPERATOR]
2120008F SYMBIONT_0001   HIB      6     7223    0 00:05:23.34     29327      69
         [GROUP1,SYSTEM]
21200090 SYMBIONT_0002   HIB      4       16    0 00:00:13.32       193      70
         [GROUP1,SYSTEM]
21200092 SYMBIONT_0004   HIB      4    46137    0 00:29:22.24     66427      77
         [GROUP1,SYSTEM]
2120009B VAXsim_Monitor  HIB      8      642    0 00:00:11.49       361     255
         [1,6]
2120009E VPA_DC          HIB     15     2870    0 00:10:15.45       234     323
         [GROUP1,SYSTEM]
21200524 BATCH_1036      COM      2  7137740    1 00:01:43.25       381     346  B
         [GROUP1,OPERATOR]
212006AC MATTHEWS        LEF      7     7973    0 00:01:07.32      4387    1500
         [GROUP11,MATTHEWS]
21200731 MATTHEWS_1      CUR      4      177    0 00:00:02.28       433     269  S
         [GROUP11,MATTHEWS]
21200635 SERVER_0635     LEF      5      369    0 00:00:09.17      1732     267  N
         [DECNET]
212007CC MARSH           LEF      4    28383    0 00:09:42.82     53227     300
         [GROUP11,MARSH]
212004D3 NOTES$0007_0*   HIB      6     1537    0 00:00:19.87      7787      67  N
         [GROUP111,NOTES$SERVER]
```

Table 9–2: System States

Name of State and Abbreviation	OK	Probably OK	Possible Problem	Definition of State
Computable COM/COMO		X		Available to use the processor
Common Event Flag Wait CEF/CEFO		X		Waiting for something to happen in another process
Collided Page Wait COLP			X	Waiting for the system
Current CUR	X			Using the processor
Free Page Wait FPG			X	Waiting for physical memory
Hibernating HIB/HIBO		X		Intentionally doing no work
Local Event Flag Wait LEF/LEFO		X		Waiting for something to happen in the image (typically I/O completion)
Mutex / Misc Resource MWAIT or RWxxx			X	Waiting for a system resource
Page Fault Wait PFW			X	Waiting for the system
Suspended SUSP/SUSPO		X		Deliberately prevented from executing

Monitoring System Processes

- System processes
  - — Created by VMS at system startup
  - — Perform certain operating system functions

Table 9–3: System Processes

Process Name	Purpose
SWAPPER	Maintains adequate free memory; transfers processes between physical memory and disk
ERRFMT	Records device errors in file on disk
JOB_CONTROL	Manages print jobs, batch jobs, interactive processes, and process accounting
OPCOM	Transmits and records user requests and operator replies
SYMBIONT_nnnn	Prints files on line printers
xxxACP	Manages transmission of data to and from tapes, interconnected computers, and some disks

Table 9–4: Results of System Process Deletion

Process Name	Effect of Deletion
SWAPPER	(Cannot be deleted)
ERRFMT	No device errors recorded
JOB_CONTROL	Users cannot log in and queues do not function
OPCOM	No user/operator communication
SYMBIONT_nnnn	Printers may not print
xxxACP	System slows down, possibly stops entirely

- To restart ERRFMT, OPCOM, or JOB_CONTROL:
  1. Log in as SYSTEM
  2. Run STARTUP.COM
  3. Enter process–image name as P1 parameter, for example:

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

Obtaining Information About a Device

- To display information about devices on your system:

  $ SHOW DEVICES

Table 9–5: SHOW DEVICES Command

List of all devices on the system	$ SHOW DEVICES
Amount of unused space on a mounted disk volume	$ SHOW DEVICES /MOUNTED
Characteristics or error count on a specific device	$ SHOW DEVICES device–name
Owner of an allocated device	$ SHOW DEVICES /FULL device–name
Names of all open files on a volume (You need SYSPRV and WORLD privileges to display complete information)	$ SHOW DEVICES /FILES device–name

Example 9–2: SHOW DEVICES /MOUNTED Output

```
$ SHOW DEVICES /MOUNTED

Device             Device           Error    Volume         Free  Trans Mnt
 Name              Status           Count     Label        Blocks Count Cnt
DJA1:              Mounted              1  GONZO          44418      1   3
DUA2:              Mounted              0  ANIMAL        162004      1   3
DUA3:              Mounted              0  OSCAR           2568      1   3
```

Example 9–3: SHOW DEVICES /FULL Output

```
$ SHOW DEVICE/FULL DJA0:
Disk DJA0:, device type RA60, is online, mounted, file-oriented device,
    shareable, error logging is enabled.

    Error count                    69    Operations completed              75804
    Owner process                  ""    Owner UIC                         [1,1]
    Owner process ID         00000000    Dev Prot    S:RWED,O:RWED,G:RWED,W:RWED
    Reference count                63    Default buffer size                 512
    Total blocks               400176    Sectors per track                    42
    Total cylinders              2382    Tracks per cylinder                   4

    Volume label          "BUNNY_SYS"    Relative volume number                0
    Cluster size                   2    Transaction count                   140
    Free blocks                27618    Maximum files allowed             66696
    Extend quantity                5    Mount count                           1
    Mount status              System    Cache name        "BUNNY$DJA0:XQPCACHE"
    Extent cache size             64    Maximum blocks in extent cache     2761
    File ID cache size            64    Blocks currently in extent cache   1246
    Quota cache size               0    Maximum buffers in FCP cache        129

  Volume status:  subject to mount verification, write-through caching enabled.
```

Monitoring Memory Resources

- System performance is strongly dependent on amount of physical memory.
- SHOW MEMORY command displays information about system's physical memory.

Example 9–4: SHOW MEMORY Output

```
$ SHOW MEMORY

              System Memory Resources on 30-APR-1988 16:39:33.81
Physical Memory Usage (pages):      Total        Free      In Use    Modified
  Main Memory (16.00Mb)             32768       23954        8516         298

Slot Usage (slots):                 Total        Free    Resident     Swapped
  Process Entry Slots                  30          11          19           0
  Balance Set Slots                    27          10          17           0

Fixed-Size Pool Areas (packets):    Total        Free      In Use        Size
  Small Packet (SRP) List             640         102         538          96
  I/O Request Packet (IRP) List       328          96         232         176
  Large Packet (LRP) List              39          19          20        1648

Dynamic Memory Usage (bytes):       Total        Free      In Use     Largest
  Nonpaged Dynamic Memory          643584       36512      607072       30272
  Paged Dynamic Memory             205312       75600      129712       74480

Paging File Usage (pages):                       Free  Reservable       Total
  DISK$COCOA_SYS:[SYS0.SYSEXE]SWAPFILE.SYS
                                                15000       15000       15000
  DISK$COCOA_SYS:[SYS0.SYSEXE]PAGEFILE.SYS
                                                23636       -6941       30000

Of the physical pages in use, 3976 pages are permanently allocated to VMS.
```

Table 9–6: Effect of Memory Sizes on Performance

Item	Description	Problem
Free Physical Memory	Size of the free page list (the number of pages available for processes that need memory)	If less than a few hundred blocks, swapping occurs.
Free Process Entry Slots	The number of additional processes the VMS system can create	If zero, no users can log in and no new processes can be created.
Free Balance Set Slots	The maximum number of additional processes the VMS system can swap in	If zero, then swapping occurs even if there are enough free pages available.
Fixed–size Pool Areas (Packets)	Nonpaged memory in system space, used primarily for I/O	If any item is zero, the system tries to increase it. Enter the SHOW MEMORY/FULL command for more information.
Free Paged Dynamic Memory, Free Nonpaged Dynamic Memory	The amount of dynamic memory left for the system to use	If too small, system response deteriorates.
Free Swap File Pages	The number of pages available on disk for swapping	If too small, then VMS uses page file instead (can significantly reduce performance).
Free Page File Pages	The number of pages available on disk for paging	If too small, processes wait in MWAIT state. System prints a message on the console terminal when the page file reaches 60% and 90% full.

Monitoring Print and Batch Queues

- To examine queues on the system:

  $ SHOW QUEUE/DEVICES/ALL

  $ SHOW QUEUE/BATCH/ALL

MONITOR Utility

- Display information about system resource usage

  $ MONITOR class–name(s)[/qualifiers]

- MONITOR commands can:
  - — Display a class of information
  - — Set default classes
  - — List defaults
  - — Execute command procedures
  - — Obtain help
  - — Exit from the utility
- Unlike most utilities that display system information, MONITOR can:
  - — Display several classes of information alternately
  - — Summarize statistics over a long period of time
  - — Record information in a disk file
  - — Play back information that it has recorded

Example 9–5: Invoking the MONITOR Utility

```
$ MONITOR
MONITOR>
```

Table 9–7: MONITOR Class Names

Class Description	Class Name
All classes	ALL_CLASSES
Brief display of system status in a cluster	CLUSTER
DECnet–VAX statistics	DECNET
Disk I/O statistics	DISK
Distributed lock management statistics	DLOCK
File system statistics	FCP
File system cache statistics	FILE_SYSTEM_CACHE
System I/O statistics	IO
Lock management statistics	LOCK
Time spent in each processor mode	MODES
Disk server statistics in cluster	MSCP
Page management statistics	PAGE
Space allocation in nonpaged dynamic memory	POOL
Statistics on all processes	PROCESSES
RMS file I/O statistics	RMS
System communication services statistics	SCS
Number of processes in each scheduler stats	STATES
Brief display of general system status (includes information displayed in other classes)	SYSTEM

Example 9–6: MONITOR Screen Display of the PAGE Class

```
$ MONITOR PAGE

                    VAX/VMS Monitor Utility
                PAGE MANAGEMENT STATISTICS
                      on node SPIDER
                  30-APR-1988 17:04:35

                                CUR        AVE        MIN        MAX

Page Fault Rate                2.33       2.33       2.33       2.33
Page Read Rate                 6.33       6.33       6.33       6.33
Page Read I/O Rate             1.00       1.00       1.00       1.00
Page Write Rate                0.00       0.00       0.00       0.00
Page Write I/O Rate            0.00       0.00       0.00       0.00

Free List Fault Rate           1.00       1.00       1.00       1.00
Modified List Fault Rate       0.00       0.00       0.00       0.00
Demand Zero Fault Rate         0.33       0.33       0.33       0.33
Global Valid Fault Rate        0.00       0.00       0.00       0.00
Wrt In Progress Fault Rate     0.00       0.00       0.00       0.00
System Fault Rate              0.00       0.00       0.00       0.00

Free List Size             78757.00   78757.00   78757.00   78757.00
Modified List Size           775.00     775.00     775.00     775.00
```

NOTE

Each value is updated every few seconds.

Table 9–8: MONITOR PROCESSES Class Qualifiers

Description	Qualifier
Top buffered I/O users	/TOPBIO
Top direct I/O users	/TOPDIO
Top CPU users	/TOPCPU
Top page fault users	/TOPFAULT

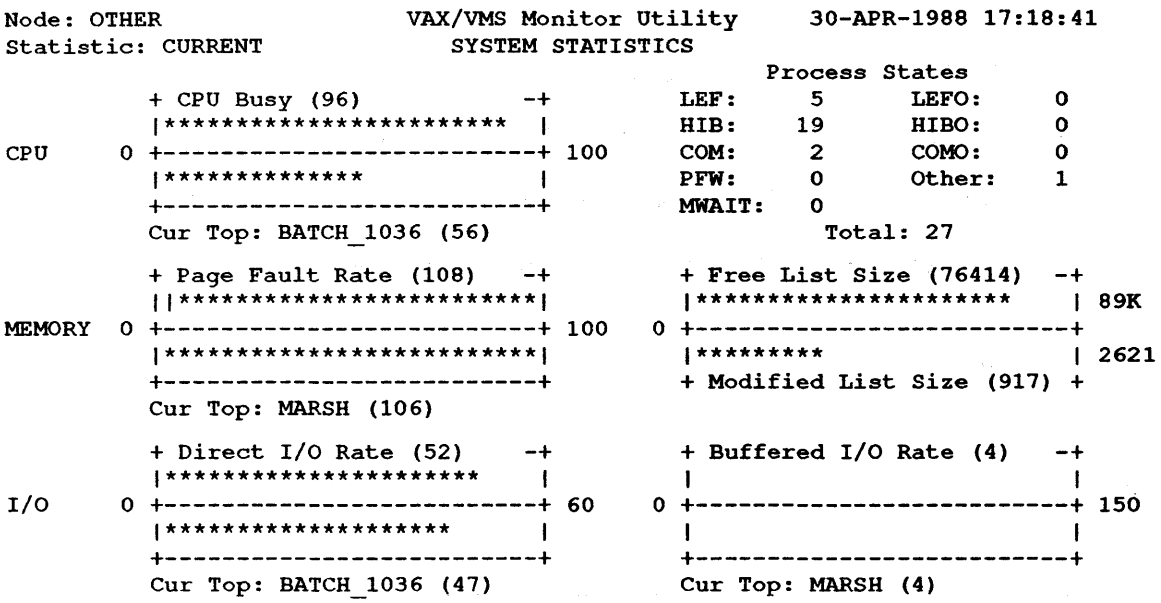
Example 9–7: MONITOR PROCESSES /TOPCPU Screen Display

```
$ MONITOR PROCESSES /TOPCPU

                          VAX/VMS Monitor Utility
                          TOP CPU TIME PROCESSES
                              on node SUPER
                           30-APR-1988 17:13:54

                                  0         25        50        75       100
                                  + - - - - + - - - - + - - - - + - - - - -+
 21200524  BATCH_1036          78  **********************************
                                  |         |         |         |          |
 21200646  JKMARTIN             9  ***
                                  |         |         |         |          |
 2120083F  MATTHEWS_1           3  *
                                  |         |         |         |          |
                                  |         |         |         |          |
                                  |         |         |         |          |
                                  |         |         |         |          |
                                  |         |         |         |          |
                                  |         |         |         |          |
                                  + - - - - + - - - - + - - - - + - - - - -+
```

NOTE

Bar graph is updated every few seconds.

Example 9–8: MONITOR SYSTEM Screen Display



Information on a Specific Process

- Display information about a particular process

  $ SHOW PROCESS/CONTINUOUS [/ID=proc-id] [proc-name]

  $ SHOW PROCESS/CONTINUOUS/ID=process_id

  — Requires WORLD privilege to show information about a process in a different UIC group

  — To exit, type E

NOTE

System updates information every few seconds.

Example 9-9: Output from SHOW PROCESS /CONTINUOUS

```
$ SHOW PROCESS /CONTINUOUS /ID=7CC

                              Process MARSH                        17:27:29

    State               LEF                 Working set                 861
    Cur/base priority   9/4                 Virtual pages              3895
    Current PC          7FFEE44C            CPU time            00:10:43.48
    Current PSL         03C00004            Direct I/O                 3691
    Current user SP     7FEF9EBC            Buffered I/O              26600
    PID                 000007CC            Page faults               58835
    UIC                 [GROUP11,MARSH]     Event flags            E0000043
                                                                   D4000002

    $1$DUA0:[SYS1.SYSCOMMON.][SYSEXE]MAIL.EXE
```

Interactive Users

- To list interactive processes and their terminal device codes:

  $ SHOW USERS

Example 9–10: Output from SHOW USERS

```
$ SHOW USERS

          VAX/VMS Interactive Users
            1-MAY-1988 12:27:52.92
    Total number of interactive users = 6

 Username     Process Name      PID      Terminal

 CASALS       CASALS          21000152  VTA22:         LTA9:
 HEIFETZ      HEIFETZ         210001A6  VTA181:        LTA12:
 KREUTZER     KREUTZER        210006A8  VTA178:        TTA3:
 PAGANINI     PAGANINI        2100033D  RTA1:
 PIATIGORSKY  PIATIGORSKY     21000544  VTA176:        LTA8:
 ROSE         ROSE            21000A6C  VTA26:         LTA12:
```

Collecting Process Information with the Accounting Utility

- System Accounting File: SYS$MANAGER:ACCOUNTNG.DAT
- Contents are used to:
  - — Record system activity
  - — Charge for system resources used
  - — Analyze relationship between system activity and performance
  - — Monitor system activity for security reasons
- Can be read by VMS Accounting utility to produce formatted reports
- Can be read by user–written processing utility
- Input/output control by system manager with DCL commands:
  - — SET ACCOUNTING
  - — ACCOUNTING
- JOB_CONTROL writes records to ACCOUNTNG.DAT when the following events occur:
  - — Process deletion/logout
  - — Print job completion
  - — Login failure
  - — Batch job completion
- Contents of accounting record
  - — System resource usage
  - — Identity of resource user

Example 9–11: Accounting Record, Full Format

```
INTERACTIVE Process Termination
-------------------------------

Username:           VAL                 UIC:                [PERSONNEL,VAL]
Account:            PERSONNEL           Finish time:        30-APR-1988 17:46:43.23
Process ID:         212007CC            Start time:         30-APR-1988 10:15:36.40
Owner ID:                               Elapsed time:                0 07:31:06.83
Terminal name:      VTA85:              Processor time:              0 00:11:37.22
Remote node addr:                       Priority:           4
Remote node name:                       Privilege <31-00>:  1014C000
Remote ID:                              Privilege <63-32>:  00000000
Queue entry:                            Final status code:  00000001
Queue name:
Job name:
Final status text:  %SYSTEM-S-NORMAL, normal successful completion

Page faults:             65489          Direct IO:              4091
Page fault reads:         2383          Buffered IO:           28753
Peak working set:         1500          Volumes mounted:           0
Peak page file:           7226          Images executed:         379
```

- Recording is enabled by default (except image activity).
- SET ACCOUNTING command controls which types of records are written to ACCOUNTNG.DAT by JOB_CONTROL.
  - BATCH
  - INTERACTIVE
  - MESSAGE
  - LOGIN_FAILURE
  - PRINT
  - PROCESS
  - IMAGE

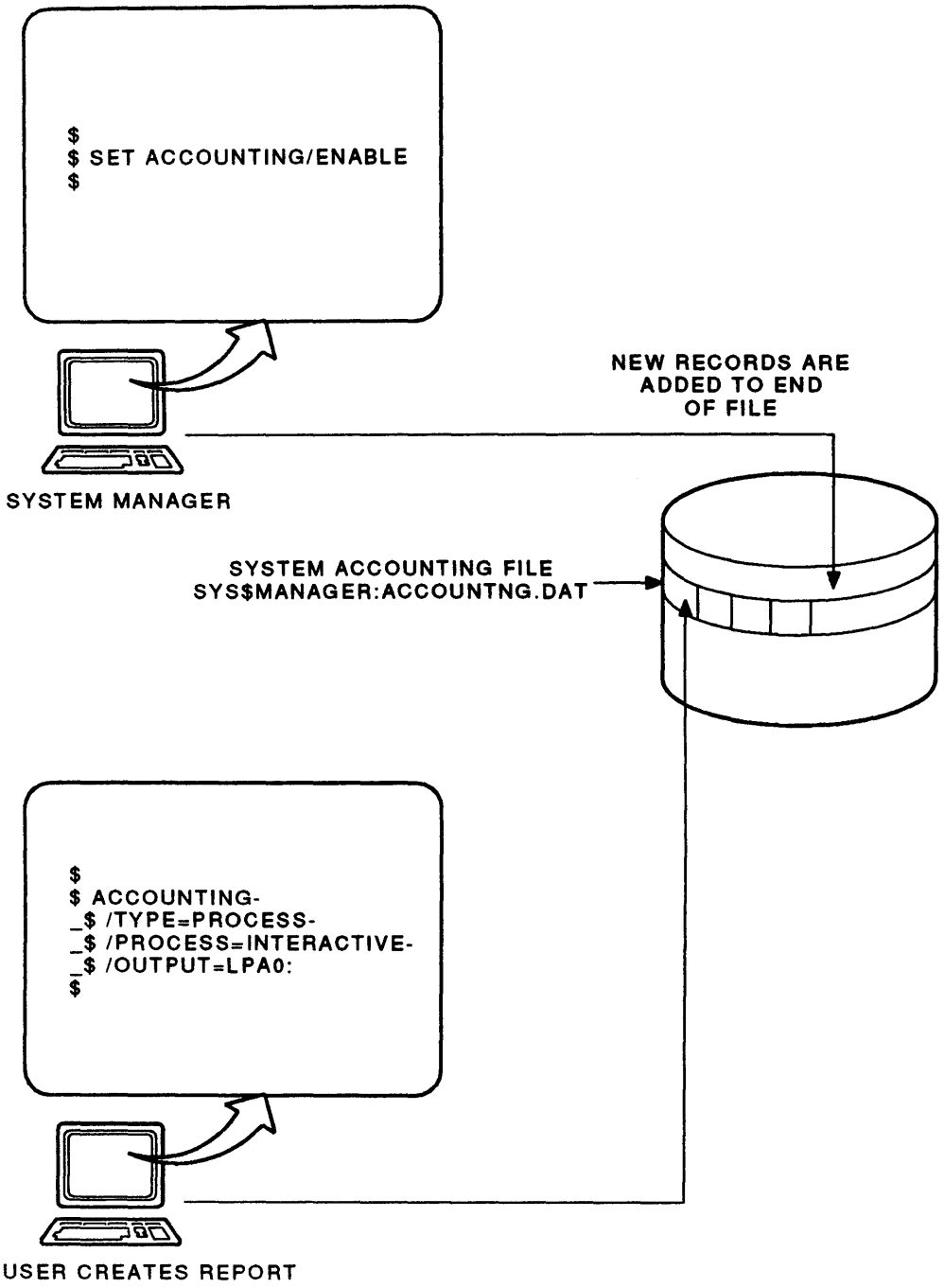
Table 9–9: Recording Accounting Information

Operation	Command Format and Examples
Enable the recording of all accounting information except image accounting	`$ SET ACCOUNTING / ENABLE`
Disable the recording of all accounting information	`$ SET ACCOUNTING /DISABLE`
Enable the recording of accounting information selectively	`$ SET ACCOUNTING /ENABLE=(record-type[,...])` `$ SET ACCOUNTING /ENABLE=(PRINT,LOGIN_FAILURE)`
Disable the recording of accounting information selectively	`$ SET ACCOUNTING /DISABLE=(record-type[,...])` `$ SET ACCOUNTING /DISABLE=(LOGIN_FAILURE)`
Close the current accounting file and open a new one	`$ SET ACCOUNTING /NEW_FILE`

Using the Accounting Utility to Produce Reports

- Accounting utility
  - — Reads system accounting file
  - — Produces reports

Figure 9–1: System Accounting File



- Qualifiers to DCL command ACCOUNTING control
  - — Which accounting records to analyze
  - — Which details to disclose

    Full
    Brief
    Summary

  - — Which order to display records
- Full format accounting report (Example 9–11) generated by:

```
$ ACCOUNTING /FULL /TYPE=PROCESS /PROCESS=INTERACTIVE
```

- Brief format accounting report (Example 9–12) generated by:

```
$ ACCOUNTING /SINCE=27-APR-1988:07:30 /BEFORE=27-APR-1988:08:00
```

Example 9–12: Accounting Records, Brief Format

```
     Date / Time       Type      Subtype      Username      ID      Source   Status
------------------------------------------------------------------------------
27-APR-1988 07:32:00 PROCESS NETWORK     DECNET       20801B24 HARDY    10000004
27-APR-1988 07:42:47 PROCESS NETWORK     DECNET       20801AA5 SCDGAT   10000004
27-APR-1988 07:56:07 PRINT               BECKER       2080009B         00040001
27-APR-1988 08:01:57 PROCESS NETWORK     DECNET       20801A28 ZEKE     10000004
27-APR-1988 08:02:00 PROCESS NETWORK     DECNET       20801A29 HARDY    10000004
27-APR-1988 08:02:42 PROCESS NETWORK     DECNET       20801B2A HARDY    10000004
27-APR-1988 08:06:53 PROCESS INTERACTIVE KENT         2080182B VTA270:  10000001
27-APR-1988 08:09:37 PROCESS NETWORK     DECNET       20801AAC SCDGAT   10000004
27-APR-1988 08:12:23 PRINT               JOHNSTON     20201448         0000002C
27-APR-1988 08:13:48 PROCESS INTERACTIVE PIANTEDOSI   208017AF PARROT   10000001
27-APR-1988 08:15:03 PRINT               JOHNSTON     20201448         00040001
27-APR-1988 08:20:51 PROCESS NETWORK     DECNET       20801BAD SCDGAT   10000004
27-APR-1988 08:26:50 PROCESS NETWORK     DECNET       208015B0 UCOUNT   10000004
27-APR-1988 08:28:10 PRINT               BECKER       2080009B         00040001
```

- To generate the summary format accounting report (Example 9–13)

```
$ ACCOUNTING /SINCE=27-APR-1988:05:00 /BEFORE=27-APR-1988:10:00 -
_$ /TYPE=PROCESS /SUMMARY=(HOUR,USER) /REPORT=BUFFERED_IO
```

Example 9–13: Accounting Report, Summary Format

```
HH Username      Buffered
                    I/O
------------------------
05 DECNET            2206
06 DECNET            2052
06 NOTES$SERVER       166
07 DECNET            1384
08 CLEARY             507
08 DECNET            2984
08 HENDRICKS        18525
08 JONES             3044
08 KENT               860
08 LMARSH             745
08 PIANTEDOSI         217
09 BAKER              285
09 BECKER             113
09 DECNET            9657
09 FRIEDMAN           407
09 JOHNSTON           239
09 KELMANSON          963
09 MATTHEWS          1025
09 NAGLE             1125
09 SYSTEM            6084
09 YWOSKUS            135
```

Table 9–10: Some Qualifiers Used to Specify Content of Accounting Report

Qualifier	Comments
/BEFORE=time	Selects records dated before specified time
/SINCE=time	Selects records dated after specified time
/QUEUE=queue–name	Name of print or batch queue
/JOB=job–name	Name of job sent to queue
/ENTRY=entry–number	Number generated when job was entered on queue
/PRIORITY=priority	Base priority of user—helps to create report on all interactive users or all real–time users
/ACCOUNT=account–name	Specified in UAF record
/UIC=uic	Specified in UAF record
/USER=user–name	Specified in UAF record
/TERMINAL=terminal–name	Device name of terminal
/PROCESS=process–type	BATCH, INTERACTIVE, DETACHED, and others
/TYPE=record–type	PRINT, LOGFAIL, PROCESS, and others

Table 9–11: Qualifiers Affecting Output Format of Accounting Report

Qualifier	Comments
/TITLE=title	Specifies text to be printed at the top of report
/REPORT=item	Includes specified items in a summary report (default is REPORT=RECORDS)
/SORT=item	Sorts records in ascending or descending order by one or more items
/SUMMARY=item	To produce summary report, grouped by the items you specify in ascending order (default is USER)
/FULL	To display all data in selected records. Do not use with /BINARY or /SUMMARY
/OUTPUT=file–spec	Sends the output to a specified file (default is SYS$OUTPUT)
/LOG	Displays log messages about progress of utility
/REJECTED=file–spec	Saves records not selected in a file in binary format
/BINARY	Produces output in binary rather than ASCII format —useful for making a smaller accounting file from which to produce multiple reports

- Default accounting file can become quite large
- Rename default file periodically or create historical accounting files with partial accounting information only

Example 9–14: Selecting Accounting Files

```
$ ACCOUNTING /TYPE=PRINT
$
$ ACCOUNTING /TYPE=PRINT SYS$MANAGER:ACCO_88_JUN_12.DAT
$
$ ACCOUNTING /TYPE=PRINT SYS$MANAGER:ACCO_88_JUN*.DAT
```

Table 9–12: Creating an Accounting Report

Operation	Command Element	Comment
Step 1: Select an accounting file.	Parameter	Default is SYS$MANAGER:ACCOUNTNG.DAT
Step 2: Select the type of record to analyze.	Qualifier /TYPE	Default is all types
Step 3: Select records to analyze, based on the contents of specific fields in the records.	Many qualifiers	Individual fields may be present in some record types and not in others.
Step 4: Sort the selected records.	Qualifier /SORT	Affects display order of records in full and brief formats
Step 5: Choose the format of the report.	Qualifiers /FULL, /SUMMARY, /REPORT	Brief display is the default if no qualifiers are specified.
Step 6: Enter the appropriate command to produce the report.	ACCOUNTING command	Requires read access to the accounting file

- Use command procedures containing qualifiers to create accounting reports

  — Full command string:

```
$ ACCOUNTING -
_$ /SINCE=02-JUL-1984:05:00 -
_$ /BEFORE=02-JUL-1984:10:00-
_$ /TYPE=PROCESS -
_$ /SUMMARY=(HOUR,USER) -
_$ /REPORT=BUFFERED_IO
```

- Contents of the DCL command procedure BUFFSUM.COM:

```
/TYPE=PROCESS /SUMMARY=(HOUR,USER) /REPORT=BUFFERED_IO
```

- Abbreviated command string incorporating the DCL command procedure BUFFSUM.COM:

```
$ ACCOUNTING -
_$ /SINCE=02-JUL-1984:05:00 -
_$ /BEFORE=02-JUL-1984:10:00@BUFFSUM
```

MAINTAINING SYSTEM PERFORMANCE

- Optimum system performance requires match of system resources with:
  - — Size of VMS data structures
  - — Number of installed image files
  - — Size of system files
- System files automatically customized at system installation to improve system performance
  - — SYS$SYSTEM:VAXVMSSYS.PAR (system parameters)
  - — SYS$MANAGER:VMSIMAGES.DAT (list of images to install)
- System files, sizes customized at system installation
  - — Paging file
  - — Swapping file
  - — Dump file

Table 9–13: System Files

File	Default File Specification	Function
Paging file	SYS$SYSTEM:PAGEFILE.SYS	Used to create and manage virtual memory
Swapping file	SYS$SYSTEM:SWAPFILE.SYS	Used to manage physical memory use
Dump file	SYS$SYSTEM:SYSDUMP.DMP	Used to save a copy of physical memory when the system fails

- Change in physical resources should be accompanied by change in:
  - — System parameter values
  - — System file sizes
- Tools to use
  - — SYS$UPDATE:AUTOGEN.COM – Command procedure (recommended method)

    Determines hardware resources
    Computes system parameters
    Creates list of image files to install
    Calculates size of page, swap, and dump files
  - — SYS$SYSTEM:SYSGEN.EXE – utility

    Changes system parameters
    Changes size of page, swap, and dump files
  - — SYSBOOT – Conversational startup utility

    Changes system parameters

Reconfiguring the System with AUTOGEN

- AUTOGEN runs automatically at system installation
  - — Determines system hardware resources
  - — Records appropriate system configuration (to be established at startup)
  - — Sets system parameter values using SYSGEN utility
  - — Creates list of images to install
  - — Creates system files using SYSGEN utility
  - — Optionally reboots the system to allow new parameters to take effect
- You should run AUTOGEN:
  - — During a new installation or upgrade
  - — When system physical resources change
  - — When system workload changes significantly
  - — When you add a layered (optional) software product

    Product installation guide tells you what parameters you need to adjust.
  - — When you install a shared image
  - — To specify a system parameter value or file size manually
- To run AUTOGEN:

  @SYS$UPDATE:AUTOGEN [start–phase] [end–phase] [execution–mode]

Table 9–14: AUTOGEN Phases

Phase	Function	Input files	Output files
SAVPARAMS	Record feedback data	None	AGEN$FEEDBACK.DAT
GETDATA	Collect data required for calculations	MODPARAMS.DAT VMSPARAMS.DAT AGEN$FEEDBACK.DAT	PARAMS.DAT
GENPARAMS	Calculate parameter values and file sizes,and generate list of images to install	PARAMS.DAT	SETPARAMS.DAT VMSIMAGES.DAT AGEN$FEEDBACK.REPORT
TESTFILES	Display calculated file sizes	PARAMS.DAT	SYS$OUTPUT
GENFILES	Generate new files	PARAMS.DAT	PAGEFILE.SYS SWAPFILE.SYS (and secondary paging and swapping files) SYSDUMP.DMP AGEN$FEEDBACK.REPORT
SETPARAMS	Save calculated parameters	SETPARAMS.DAT	VAXVMSSYS.PAR AUTOGEN.PAR VAXVMSSYS.OLD
SHUTDOWN	Shut down the system	None	None
REBOOT	Reboot to allow new files and parameters to take effect	None	None

NOTE

All data files are in the directory SYS$SYSTEM:.

Running AUTOGEN

Modifying System Parameters Without Changing File Sizes

1. `@SYS$UPDATE:AUTOGEN SAVPARAMS GENPARAMS`
2. Review these files:
   - PARAMS.DAT (input to calculations)
   - SETPARAMS.DAT (calculated parameters)
   - AGEN$FEEDBACK.REPORT (report on feedback data)
3. If you want to change any parameters, edit MODPARAMS.DAT and rerun AUTOGEN as in step 1.
4. `@SYS$UPDATE:AUTOGEN SETPARAMS REBOOT`

Changing System File Sizes

1. `@SYS$UPDATE:AUTOGEN SAVPARAMS TESTFILES`
2. Examine file sizes.
3. If you want to change file sizes, edit MODPARAMS.DAT and rerun AUTOGEN as in step 1.
4. `@SYS$UPDATE:AUTOGEN GENPARAMS REBOOT`

Example 9–15: MODPARAMS.DAT File

```
!
! MODPARAMS.DAT for node DITTO
!
SCSSYSTEMID = 2197
SCSNODE = "DITTO "
PAGEFILE = 60000            !
ADD_GBLPAGES = 425+507+157 ! CMS, BLISS32 and ADA (FJM 9/13/87)
ADD_GBLSECTIONS = 4+5+2     ! CMS, BLISS32 and ADA (FJM 9/13/87)
LOCKIDTBL = 2048            ! FOR RDB (A. B. 9/25/87)
RESHASHTBL = 16384     ! raised for CDD 40+ (D. E. 2/22/88)
MIN_VIRTUALPAGECNT = 136100! for VTX (vhm 5/25/88)
```

Reconfiguring the System with SYSGEN

- Use SYSGEN to:
  - — Make temporary changes in system parameters.
  - — Perform functions not available from AUTOGEN.

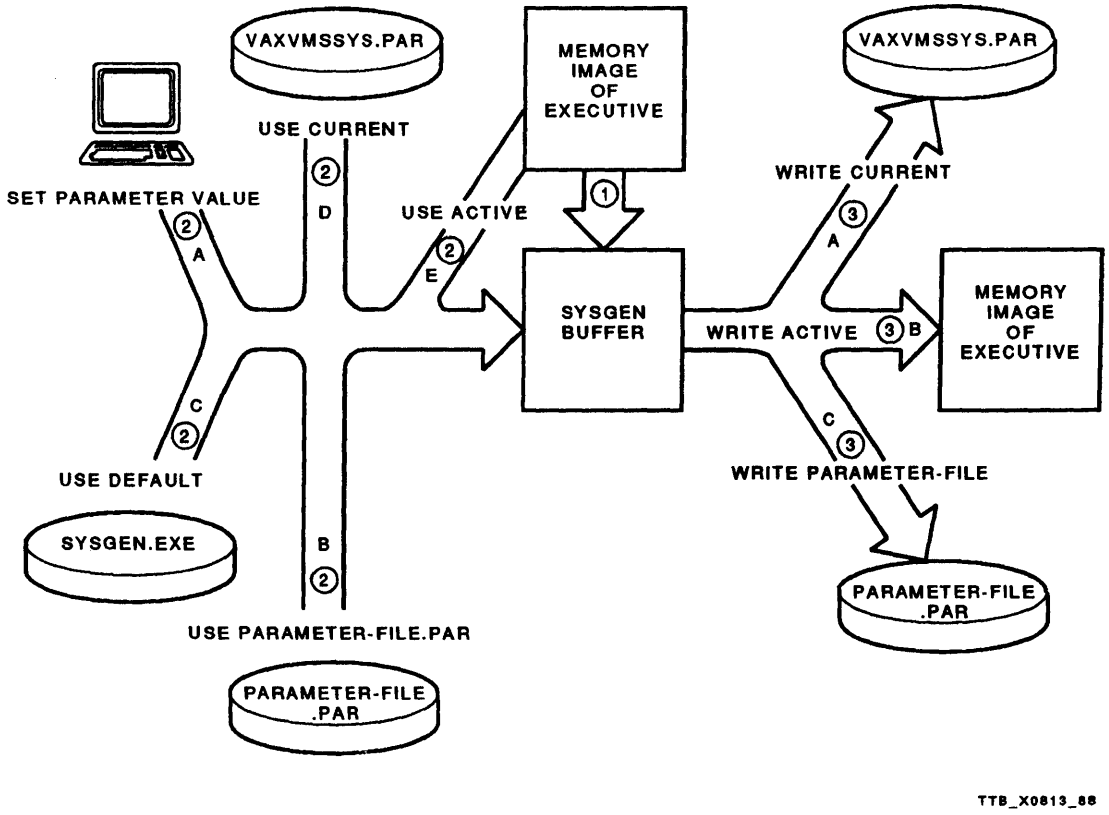
    Display parameter settings.
    Load device drivers.
  - — Save system parameters in alternate parameter files.
- System parameters
  - — Current
  - — Active
  - — Dynamic
- To invoke SYSGEN utility:

```
$ RUN SYS$SYSTEM:SYSGEN
```

Table 9–15: Using the SYSGEN Utility

Function	Command Format and Examples
Examine a system parameter or group of parameters	SYSGEN> SHOW parameter SYSGEN> SHOW MAXPROCESSCNT SYSGEN> SHOW /parameter-group SYSGEN> SHOW /ALL
Modify a system parameter in the SYSGEN buffer	SYSGEN> SET parameter value SYSGEN> SET UAFALTERNATE 1
Read a set of system parameters into the SYSGEN buffer	SYSGEN> USE source SYSGEN> USE ACTIVE SYSGEN> USE DEFAULT SYSGEN> USE CURRENT SYSGEN> USE SYS$MANAGER:ALTPARM.PAR
Copy the parameters in the SYSGEN buffer to memory or disk	SYSGEN> WRITE destination SYSGEN> WRITE ACTIVE SYSGEN> WRITE CURRENT SYSGEN> WRITE SYS$MANAGER:ALTPARM.PAR
Create or extend a paging, swapping, or dump file	SYSGEN> CREATE system-file /SIZE=blocks SYSGEN> CREATE DISKALT:[SYSEXE]NEWPAGEFILE.SYS /SIZE=85000
Activate a secondary paging or swapping file	SYSGEN> INSTALL system-file /type SYSGEN> INSTALL DISKALT:[SYSEXE]NEWPAGEFILE.SYS /PAGEFILE SYSGEN> INSTALL DISKALT:[SYSEXE]NEWSWAPFILE.SYS /SWAPFILE
Use an alternate DCL startup file	SYSGEN> SET /STARTUP startup-fil SYSGEN> SET /STARTUP SYS$SYSTEM:ALTSTART.COM
Display HELP information	SYSGEN> HELP [command [/qualifier]]
Exit from SYSGEN	SYSGEN> EXIT

Figure 9–2: Using the SYSGEN Utility to Modify System Parameters



Performance Tuning

- Use AUTOGEN to match system configuration to hardware resources and workload on your system.
- Do not use SYSGEN to modify system parameters arbitrarily.

SUMMARY

- The VMS operating system provides many commands and utilities for monitoring:
  - — System performance
  - — Process activity
  - — Device status
- The SHOW SYSTEM command shows the status of each process, including its state.
  - — Make sure all the necessary system processes are present.
- The Accounting utility shows resource consumption by process.
- The SHOW DEVICES command displays information about devices on the system.
- The SHOW MEMORY command displays information about physical memory usage and system file usage.
- The SHOW QUEUE command displays the status of print and batch queues.
- The Monitor utility displays information about system resource usage.
- The SHOW PROCESS /CONTINUOUS command displays information about a particular process.
- The SHOW USERS command lists the interactive users on the system.
- The AUTOGEN utility:
  - — Generates information used to maintain optimum performance.

    System parameters
    System file sizes
    Images to be installed
  - — Uses system information as input

    Hardware resources
    System workload
    Values specified by system manager
- The SYSGEN utility can also be used to set system parameters, and performs several functions not available with AUTOGEN.

MODULE 10
EXERCISES

UNDERSTANDING THE USER ENVIRONMENT

Written Exercises

1. In the exercise below, match each description with the appropriate component of the hardware environment. Components of the hardware environment can be used once, more than once, or not at all.

   Components of Hardware Environment:

   a. CPU

   b. Console Subsystem

   c. Main Memory

   d. I/O Subsystem

   Descriptions:

   _____ MicroVAX II processor is an example
   _____ Stores instructions and data
   _____ Used to monitor and control the system
   _____ Consists of peripherals
   _____ Executes instructions
   _____ Control panel is part of this subsystem
   _____ Used for starting up and shutting down the system

2. _____ are used to connect the various subsystems of the computer.

   a. Peripheral devices

   b. Network communication devices

   c. Interconnect devices

   d. Storage devices

3. _____ have a television–like screen for displaying information.

   a. Hard–copy terminals

   b. Video terminals

   c. Laser printers

   d. Mass storage devices

4. _____ is NOT a peripheral device.

   a. Terminal

   b. Printer

   c. CPU

   d. Disk drive

5. _____ are high–speed machines usually used for large quantities of output.

   a. Hard–copy terminals

   b. Disk drives

   c. Laser printers

   d. Line printers

6. _____ is NOT a type of disk.

   a. Reel

   b. Cartridge

   c. Diskette

   d. Disk pack

7. _____ record data on magnetic tape.

   a. Disk drives

   b. Tape drives

   c. Terminal servers

   d. VAXcluster systems

8. In the exercise below, match each device name with the appropriate description.

   Device Names:

   a. MTA0

   b. DUA2

   c. TTB4

   d. LPA0

   Descriptions:

   _____ Fifth terminal connected to controller B
   _____ First TU77 tape drive connected to controller A
   _____ Third RA81 disk connected to controller A

9. In the exercise below, match each description with the appropriate system configuration.

   System Configurations:

   a. Single processor configuration

   b. Tightly coupled system

   c. Loosely coupled system

   d. VAXcluster configuration

   Descriptions:

   _____ Nodes have separate file systems
   _____ Processors cannot operate independently
   _____ A VAX processor and its peripheral devices
   _____ An example of this type of configuration is a network
   _____ Configured mid-way between tightly coupled and loosely coupled systems
   _____ This type of configuration is also called a multiprocessor
   _____ All nodes reside in close proximity to each other

10. The _____ is an example of a tightly coupled system.

    a. MicroVAX II

    b. VAX 8820

    c. VAX 8600

    d. VAX-11/780

11. In the exercise below, match each network component with the most appropriate description.

    Network Components:

    a. Hardware communication devices

    b. Data transmission media

    c. Terminal servers

    d. DECnet software

    Descriptions:

    _____ Enables communication between networked systems
    _____ An example is Ethernet cable
    _____ Provides connections between user terminals and systems in a network
    _____ Necessary to achieve physical communication between computers
    _____ An optional component that enhances flexibility
    _____ Transmits data between communications devices

12. A _____ is a network established within a limited geographical area.

    a. Loosely coupled system

    b. Wide area network

    c. Local area network

    d. Multiprocessor

13. In addition to providing the functions of a network, _____ also provide higher availability of system resources, as well as faster and easier sharing of information and resources.

    a. VAXcluster configurations

    b. Wide area networks

    c. Tightly coupled systems

    d. Terminal servers

14. The major difference between a cluster and a network is _____.

    a. VMS DECnet software

    b. VMS cluster software

    c. Communication devices

    d. Processors

Solutions

1. In the exercise below, match each description with the appropriate component of the hardware environment. Components of the hardware environment can be used once, more than once, or not at all.

   Components of Hardware Environment:

   a. CPU

   b. Console Subsystem

   c. Main Memory

   d. I/O Subsystem

   Descriptions:

   a MicroVAX II processor is an example
   c Stores instructions and data
   b Used to monitor and control the system
   d Consists of peripherals
   a Executes instructions
   b Control panel is part of this subsystem
   b Used for starting up and shutting down the system

2. c are used to connect the various subsystems of the computer.

   a. Peripheral devices

   b. Network communication devices

   c. Interconnect devices

   d. Storage devices

3. b have a television–like screen for displaying information.

   a. Hard–copy terminals

   b. Video terminals

   c. Laser printers

   d. Mass storage devices

4. __c__ is NOT a peripheral device.

   a. Terminal

   b. Printer

   c. CPU

   d. Disk drive

5. __d__ are high–speed machines usually used for large quantities of output.

   a. Hard–copy terminals

   b. Disk drives

   c. Laser printers

   d. Line printers

6. __a__ is NOT a type of disk.

   a. Reel

   b. Cartridge

   c. Diskette

   d. Disk pack

7. __b__ record data on magnetic tape.

   a. Disk drives

   b. Tape drives

   c. Terminal servers

   d. VAXcluster systems

8. In the exercise below, match each device name with the appropriate description.

   Device Names:

   a. MTA0

   b. DUA2

   c. TTB4

   d. LPA0

   Descriptions:

   c Fifth terminal connected to controller B
   a First TU77 tape drive connected to controller A
   b Third RA81 disk connected to controller A

9. In the exercise below, match each description with the appropriate system configuration.

   System Configurations:

   a. Single processor configuration

   b. Tightly coupled system

   c. Loosely coupled system

   d. VAXcluster configuration

   Descriptions:

   c Nodes have separate file systems
   b Processors cannot operate independently
   a A VAX processor and its peripheral devices
   c An example of this type of configuration is a network
   d Configured mid–way between tightly coupled and loosely coupled systems
   b This type of configuration is also called a multiprocessor
   d All nodes reside in close proximity to each other

10. The b is an example of a tightly coupled system.

    a. MicroVAX II

    b. VAX 8820

    c. VAX 8600

    d. VAX–11/780

11. In the exercise below, match each network component with the most appropriate description.

    Network Components:

    a. Hardware communication devices

    b. Data transmission media

    c. Terminal servers

    d. DECnet software

    Descriptions:

    d Enables communication between networked systems
    b An example is Ethernet cable
    c Provides connections between user terminals and systems in a network
    a Necessary to achieve physical communication between computers
    c An optional component that enhances flexibility
    b Transmits data between communications devices

12. A __c__ is a network established within a limited geographical area.

    a. Loosely coupled system

    b. Wide area network

    c. Local area network

    d. Multiprocessor

13. In addition to providing the functions of a network, __a__ also provide higher availability of system resources, as well as faster and easier sharing of information and resources.

    a. VAXcluster configurations

    b. Wide area networks

    c. Tightly coupled systems

    d. Terminal servers

14. The major difference between a cluster and a network is __b__.

    a. VMS DECnet software

    b. VMS cluster software

    c. Communication devices

    d. Processors

MANAGING SYSTEM USERS

Written Exercises

Part I

Example 10–1 displays the characteristics of a process that has many privileges on your system. Using the information displayed in the example, determine the value of each of the following parameters:

Answers	Parameters
_____	Account Name
_____	Default Device and Directory Specification
_____	Interactive Terminal Specification
_____	Process Identification Number
_____	Process Name
_____	User Identification Code
_____	User Name
_____	Priority
_____	CPU Limit
_____	Open File Quota
_____	Working Set Limit
_____	Working Set Quota
_____	Privileges (list them)

Example 10–1: Process Parameters of a Sample Interactive Process

```
$ SHOW PROCESS/ALL

 5-FEB-1988 15:16:44.69   TXB0:              User: RUTHERFORD
Pid: 000000C2   Proc. name: SmallCreepsDay   UIC: [GENESIS,RUTHERFORD]
Priority:   4   Default file spec: DISK$COMPACT:[ATLANTIC]

Devices allocated: TXB0:

Process Quotas:
 Account name: MUSIC
 CPU limit:                        Infinite  Direct I/O limit:        18
 Buffered I/O byte count quota:        4000  Buffered I/O limit:      18
 Timer queue entry quota:                10  Open file quota:         19
 Paging file quota:                    9064  Subprocess quota:         2
 Default page fault cluster:             64  AST quota:               22
 Enqueue quota:                          30  Shared file limit:        0
 Max detached processes:                  0  Max active jobs:          0

Accounting information:
 Buffered I/O count:         791  Peak working set size:         393
 Direct I/O count:           106  Peak virtual size:            2264
 Page faults:               2916  Mounted volumes:                 0
 Images activated:            23
 Elapsed CPU time:       0 00:00:16.98
 Connect time:           0 00:09:28.01

Process privileges:
 CMKRNL                may change mode to kernel
 GRPNAM                may insert in group logical name table
 DETACH                may create detached processes
 DIAGNOSE              may diagnose devices
 GROUP                 may affect other processes in same group
 TMPMBX                may create temporary mailbox
 OPER                  operator privilege
 NETMBX                may create network device

Process rights identifiers:
 INTERACTIVE
 LOCAL

Process Dynamic Memory Area
    Current Size (bytes)          25600    Current Total Size (pages)      50
    Free Space (bytes)            21744    Space in Use (bytes)          3856
    Size of Largest Block         21712    Size of Smallest Block           8
    Number of Free Blocks             3    Free Blocks LEQU 32 Bytes        2

Processes in this tree:

Mike Rutherford
  SmallCreepsDay (*)
$
$ SHOW WORKING_SET
  Working Set      /Limit= 150   /Quota= 300    /Extent= 500
  Adjustment enabled    Authorized Quota= 300  Authorized Extent= 500
$
```

Solutions

Compare your answers with those shown below. For additional information, speak with your instructor.

Answers	Parameters
MUSIC	Account Name
DISK$COMPACT:[ATLANTIC]	Default Device and Directory Specification
TXB0	Interactive Terminal Specification
000000C2	Process Identification Code
SmallCreepsDay	Process Name
[GENESIS,RUTHERFORD]	User Identification Code
RUTHERFORD	User Name
4	Priority
Infinite	CPU Limit
19	Open File Quota
150	Working Set Limit
300	Working Set Quota
CMKRNL	Privileges (list them)
GRPNAM	
DETACH	
DIAGNOSE	
GROUP	
TMPMBX	
OPER	
NETMBX	

MANAGING SYSTEM USERS

Written Exercises

Part II

You have been told that Susan Jackson, who recently joined the finance group in your company, needs an account on your system. The account should have the same basic privileges as other accounts in the group. Members of the finance group have the group number 400 for their UIC. Each member has a default directory on the disk pointed to by the logical name FINANCE_DISK.

The password for Jackson should be her first name, and she should be required to change her password immediately upon logging in.

What steps should you follow?

Solutions

To allow a new user to access your system, you should create an account for the user. To allow the user to create and maintain files in his or her own area, you must also create a directory for the user. Finally, be sure the new user knows how to log in to the account and the procedures for using the system.

a. Creating an account

To create an account for Susan Jackson, use the AUTHORIZE utility, as follows:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF> SHOW /BRIEF [400,*]
    .
    .
    .
UAF> ADD JACKSON /PASSWORD=SUSAN /PWDEXPIRED /UIC=[400,116] -
_UAF> /DIRECTORY=[JACKSON]/DEVICE=FINANCE_DISK:
UAF> EXIT
```

The output of the SHOW /BRIEF [400,*] command listed all the accounts and their user numbers with a group number of 400. You need that information because the JACKSON account must have a unique user number. The user number chosen, 116, did not belong to anyone else having a group number of 400.

The /PWDEXPIRED qualifier assures that user Jackson must change her password immediately upon logging in to her account.

b. Creating a directory

Use the following commands to create a directory for user Jackson:

```
$ SET DEFAULT FINANCE_DISK:[000000]
$ CREATE/DIRECTORY/OWNER=JACKSON [JACKSON]
```

c. Instructing the user about system procedures

New users should receive information about logging in to, and appropriate use of, your system. This can be done by you, or by another appropriate person in your organization.

In this example, Susan Jackson should be told that her user name is JACKSON, her password is SUSAN, and she must use a new password when she logs in. In addition, she should be made aware of any procedures that are specific to your site.

You can use a command procedure to partially automate the process of adding new user accounts and creating directories. SYS$EXAMPLES:ADDUSER.COM is an example of such a procedure. To see how it works, type the following command:

```
$ @SYS$EXAMPLES:ADDUSER.COM
```

This procedure is included on your system and you can customize it to meet the requirements of your site.

MANAGING SYSTEM USERS

Laboratory Exercises

Part I

To practice controlling user processes, log in at two terminals or do the following with a partner:

1. Designate one terminal as the Manager terminal, and the other as the User terminal. Log in at both terminals. The process on the Manager terminal must have WORLD or GROUP privileges to affect the other processes.

2. The User should begin entering DCL commands that produce output, such as SHOW PROCESS and SHOW SYSTEM. (You can also create a command procedure to do this repeatedly.)

3. Find out the process ID number of the User process from the Manager terminal.

4. Suspend the User process from the Manager terminal (preferably while output is appearing on the User terminal).

5. Observe the User terminal. Notice the lack of response from the keyboard (if you press any key or combination of keys, the state of the User process does not change).

6. Allow the User process to continue.

7. Lower the priority of the User process to three from the Manager terminal. The User should continue to enter DCL commands and observe the response for several minutes.

8. Raise the priority of the User process to four from the Manager terminal. Continue to observe the response to various DCL commands.

9. Stop the User process from the Manager terminal. Can the Manager allow the process to continue?

Solutions

1. No solution needed.

2. A command procedure you might use is:

```
$LOOP:
$ SHOW PROCESS
$ SHOW SYSTEM
$ GOTO LOOP
```

   This command procedure produces an endless loop.

3. The commands SHOW SYSTEM and SHOW USERS display the PIDs of the users currently on the system. The PID is a hexadecimal number.

4. `$ SET PROCESS /SUSPEND /ID=21400FA1`

   Substitute the appropriate PID for 21400FA1.

5. No solution needed.

6. `$ SET PROCESS /RESUME /ID=21400FA1`

   Substitute the appropriate PID for 21400FA1.

7. `$ SET PROCESS /PRIORITY=3 /ID=21400FA1`

   If the process is in your UIC group, you need only specify the process name as a parameter instead of using the /ID qualifier. For example, if the process name is SMITH, and the UIC of the process has the same group number as yours, you could enter:

```
$ SET PROCESS/PRIORITY=3 SMITH
```

   Because the priority of the process is now lower than most of the priorities of the other processes on the system, you should observe a delay in the execution of the commands entered from the User terminal.

8. `$ SET PROCESS/PRIORITY=4/ID=21400FA1`

   The response to the DCL commands entered from the User terminal should be quicker.

9. `$ STOP/ID=21400FA1`

   If the process is in your UIC group, you can omit the /ID qualifier and supply the process name as a parameter.

   The process cannot be continued, since the STOP command deleted the process.

MANAGING SYSTEM USERS

Laboratory Exercises

Part II

NOTE: You cannot perform these exercises unless you have write access to the quota file on your class volume. Check with your instructor before you attempt the exercises.

1. Log in using your own account.
2. Run the SYSMAN utility (requires OPER privilege).
3. Enter the DISKQUOTA SHOW/DEVICE command, specifying your class volume name, and display all the quota records in the quota file for the class volume.
4. To observe the effects of disk quotas, perform the following:
   a. Display your current disk quota settings. Write them down.
   b. Delete your record from the quota file.
   c. Exit from the SYSMAN utility.
   d. Try to create a small text file by using either the CREATE command or a text editor.
   e. Reenter the SYSMAN utility and add a diskquota record for yourself. Specify the values you previously wrote down for permanent quota and overdraft. **Do not forget to specify your class volume name with the /DEVICE qualifier.**
   f. Exit from the SYSMAN utility.
   g. Enter the SHOW QUOTA command (from DCL level). Record the usage count. Create a small text file.
   h. Enter the SHOW QUOTA command again. Notice that your usage count has increased.
   i. Reenter the SYSMAN utility and display your diskquota record in the quota file for the class volume. Notice that your usage count has been increased here as well.
5. Modify your record to increase your permanent quota by 1000 blocks and your overdraft by 200 blocks.
6. Exit from the SYSMAN utility.
7. Set your default to SYS$SYSTEM.
8. Run the SYSMAN utility again. Enter the appropriate commands to display your record in the quota file for the class volume.

Solutions

1. No solution needed.

2. `$ RUN SYS$SYSTEM:SYSMAN`

3. `SYSMAN> DISKQUOTA SHOW /DEVICE=CLASS_DISK [*,*]`

   Substitute the name of your class volume for CLASS_DISK.

4. Enter the following commands to observe the effects of disk quotas.

   a. `SYSMAN> DISKQUOTA SHOW [320,10]`

      Substitute your UIC for [320,10]. If the utility displays the alphanumeric form of your UIC, substitute it for [320,10]. For example, if the string form of your UIC is [GRP320,SMITH], you should enter the following command:

      ```
      SYSMAN> DISKQUOTA SHOW [GRP320,SMITH]
      ```

   b. `SYSMAN> DISKQUOTA DELETE [320,10]`

   c. `SYSMAN>EXIT`

   d. Note that you are unable to create files on the volume because you do not have a disk quota.

   e. `$ RUN SYS$SYSTEM:SYSMAN`
      `SYSMAN> DISKQUOTA ADD /DEVICE=CLASS_DISK [320,10]`

      Substitute the name of your class volume for CLASS_DISK and your UIC for [320,10].

   f. `SYSMAN>EXIT`

   g. `$ SHOW QUOTA`

      The usage count should be 0. The following is a short text file you might create.

      ```
      $ CREATE FILE.TXT
      This is a short text file.
      It should take up at least one block of space.
      CTRL/Z
      ```

   h. `$ SHOW QUOTA`

i.
```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> SHOW /DEVICE=CLASS_DISK [320,10]
```

Substitute the name of your class volume for CLASS_DISK, and your own UIC for [320,10]. The utility modifies the quota file dynamically, so it recorded the blocks you used for the file when you created it.

5.
```
SYSMAN> DISKQUOTA MODIFY [320,10] /PERMQUOTA=2000 /OVERDRAFT=300
```

Substitute the appropriate values that correspond to your record.

6.
```
SYSMAN>EXIT
```

7.
```
$ SET DEFAULT SYS$SYSTEM:
```

8.
```
$ RUN SYSMAN
SYSMAN> SHOW /DEVICE=CLASS_DISK [320,10]
```

Substitute the name of your class volume for CLASS_DISK and your own UIC for [320,10]. If you do not enter the USE command, the utility uses the quota file for the current default disk (the system disk), if it exists. That file does not contain your quota record.

MANAGING SYSTEM USERS

Laboratory Exercises

Part III

You need write access to the MFD of the class disk and to the system authorization files to do this lab.

1. Create an account called PAYROLLn, where n is a number assigned by your instructor. The group number assigned to workers in Payroll is 322. Choose any member number from 60–377 (octal) that is not in use. Also create a directory and a disk quota entry for the account.

   Log in to the new account and create a small file in its directory to verify your work.

2. This account is only allowed to run a data entry program ENTRY.EXE. The name of the command procedure used to run the program is DATA.COM. Your instructor can tell you what directory these files are in.

   Modify the UAF record so that DATA.COM runs automatically when you log in to the PAYROLLn account, and so that you cannot reach the DCL prompt.

3. Log in as PAYROLLn. The DATA.COM procedure should execute automatically. The password for the procedure is GO. The procedure executes ENTRY.EXE. When ENTRY.EXE requests input, type in three numbers separated by commas.

   To test your work, log out and log in as SMITH several times. Each time, try to cause the DCL prompt to appear. (Suggestions: enter CTRL/Y, enter incorrect data at various points, enter the wrong password, etc.)

4. Do whatever is necessary to remove the account PAYROLLn from the system.

Solutions

1. To add an account, run the AUTHORIZE and DISKQUOTA utilities and create a UFD as shown below:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF>SHOW/BR [322,*]
(Output shows 63 has not been used as a member number yet)
UAF>ADD PAYROLL3 /PASSWORD=JOE-
_/UIC=[322,063]-
_/DEVICE=CLASS_DISK-
_/DIRECTORY=[PAYROLL3]
UAF>EXIT
$
$ SET DEFAULT CLASS_DISK:[000000]
$
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> DISKQUOTA ADD PAYROLL3 /DEVICE=CLASS_DISK
SYSMAN> EXIT
$
$ CREATE/DIRECTORY/OWNER=PAYROLL3 [PAYROLL3]
$
```

   You should have been able to create a test file successfully. If not, look at these solutions carefully, set up the account properly, and try again.

   Check the owner UIC of the directory if you receive protection errors. Check the contents of the QUOTA.SYS file on CLASS_DISK if you receive quota errors.

2. To modify the account so it can only run the DATA.COM procedure, use the AUTHORIZE utility to make the account captive.

```
UAF>MODIFY PAYROLL3 /LGICMD=CLASS_DISK:[PAYROLL]DATA.COM-
_/FLAGS=(CAPTIVE,LOCKPWD,DISCTLY)
```

3. If you enter incorrect answers to the DATA.COM procedure or the ENTRY program, you are logged out. If you enter letters as data to ENTRY, instead of numbers, you are logged out. If you enter a CTRL/Y key sequence, you are logged out.

4. To remove the account completely, you must delete all subdirectories and files in [PAYROLL3], remove PAYROLL3.DIR, remove the PAYROLL3 entry from the quota file, and remove the UAF record from the SYSUAF.DAT file.

MANAGING SYSTEM USERS

Laboratory Exercises

Part IV

You need two terminals to perform the following exercises, or, you can pair up with another student. Designate one terminal as an Operator terminal, and the other terminal as a User terminal. The process running on the Operator terminal must have OPER privilege.

1. Enable the Operator terminal so that it will receive messages sent to *one* of the operator classes OPER1 through OPER12. (Ask your instructor which operator class you should use.) Use the /TEMPORARY qualifier.

2. From the Operator terminal, use the REPLY command to send a message to all logged–in users, telling them that you are now on duty as an operator.

3. From the User terminal, use the REQUEST command to send a message that will be displayed on the Operator terminal. Observe the output on the Operator terminal.

4. Send another REQUEST message to OPER1 requiring a response. Observe the output on the Operator terminal.

5. Respond to the message from the Operator terminal, telling the User that you are working on the request but you do not have an answer yet.

6. Issue a REPLY command to display the status of outstanding requests made to your operator class.

7. After awhile, issue an appropriate message from the Operator terminal and abort the User request.

8. Log out from the Operator terminal.

9. Send a message to OPER1 from the User terminal that does not require a response.

10. Does the message appear on the Operator terminal? Why?

11. Log back in to the Operator terminal. Find out whether it is still enabled to receive operator messages.

Solutions

Substitute your assigned operator class for OPER1 in the following solutions.

1. `$ REPLY /ENABLE=OPER1 /TEMPORARY`

2. `$ REPLY /USERS /BELL "Janet S. is now on duty as OPER1."`

3. `$ REQUEST REPLY/TO=OPER1 "Please check LPA0."`

   You should see a message on the Operator terminal that includes this request.

4. `$ REQUEST /REPLY /TO=OPER1 "Is it OK to print from LPA0?"`

   You should see another message, but this one also contains a request identification number.

5. `$ REPLY /PENDING=123 "Still working on the problem."`

   (Substitute the appropriate request ID number for 123.)

6. `$ REPLY /STATUS`

7. `$ REPLY /ABORT=123 "LPA0 is disabled indefinitely. Use LPB0."`

8. No solution needed.

9. `$ REQUEST /TO=OPER1 "Please back up CLASS_DISK tonight."`

10. The message should not appear on the Operator terminal screen.

11. `$ REPLY /STATUS`

    You should see a message stating that an illegal operator request was made.

MANAGING QUEUES

Laboratory Exercises

Part I

1. Find out what batch queues are already on the system.
2. Create and start a batch queue with a job limit of two and a priority of one. Make sure the queue name is different from any existing queue names.

Solutions

1. `$ SHOW QUEUE /BATCH`

2. ```
   $ INITIALIZE /QUEUE /BATCH /START /JOB_LIMIT=2 -
   _$ /BASE_PRIORITY=1 SMITH_BATCH
   ```
   ```

MANAGING QUEUES

Laboratory Exercises

Part II

Team up with another student in the class or use two terminals to do the following exercises. Use one of the terminals as an output device (printer). This can be a video or a hard-copy terminal. You will enter queue manipulation commands from the other terminal.

1. Set up one terminal as an output device.

   a. Log in to the terminal.

   b. Use the SHOW TERMINAL command to list the current characteristics of the terminal.

   c. Write down the device name of the terminal for future reference.

   d. Modify the characteristics of the terminal with the SET TERMINAL/PERMANENT command so that it can be used as an output device. (You will need LOG_IO and OPER privileges to do this.)

   e. Log out from the terminal.

2. Enter the following commands from the other terminal:

   a. Spool the output device.

   b. Create and start a queue for the output terminal. Give the queue a name using the /ON qualifier.

   c. Print a file at the output terminal, using the queue you just created. Observe the output terminal to see if your job is displayed.

   d. Send a print job to your terminal execution queue, including the /HOLD qualifier in the command.

   e. Display the contents and characteristics of your terminal execution queue.

   f. Delete your job that is being held on the terminal execution queue.

Solutions

1. To set up a terminal as an output device, do the following.

   a. No solution needed.

   b. `$ SHOW TERMINAL`

   c. These solutions use the terminal name TTC5:

   d. `$ SET TERMINAL /PERMANENT /NOTYPE_AHEAD -`
   `_$ /NOBROADCAST /SPEED=2400`

      The /NOTYPE_AHEAD qualifier disables logins on this terminal. If you need to modify the terminal further after logging out, you can enter the SET TERMINAL/PERMANENT command at another terminal. For example, the following command modifies a characteristic of the TTC5 terminal:

      ```
      $ SET TERMINAL /PERMANENT /NOBROADCAST TTC5:
      ```

   e. `$ LOGOUT`

2. The following commands should eventually allow you to see your file displayed on the output terminal.

   a. `$ SET DEVICE/SPOOLED TTC5:`

   b. `$ INITIALIZE/QUEUE/START/ON=TTC5: MYQUEUE`

      Substitute the name of your output terminal for TTC5:, and substitute your own queue name for MYQUEUE.

   c. `$ PRINT/QUEUE=MYQUEUE FILE.TXT`

      If you followed the instructions in this lab carefully, you should see your file displayed on the output terminal. If your file does not appear, consult your instructor.

   d. `$ PRINT/QUEUE=MYQUEUE/HOLD FILE.TXT`

   e. `$ SHOW QUEUE/FULL/ALL MYQUEUE`

   f. `$ DELETE/ENTRY=369`

      Substitute the entry number of your job for 369.

MANAGING QUEUES

Laboratory Exercises

Part III

1. Modify the default characteristics of the terminal execution queue you created in the last laboratory exercise so that all jobs sent to the queue are printed with flag and trailer pages.
2. Send a short text file to the output terminal and observe the display.
3. Initialize and start a batch queue on the system with a job limit of two. Name it MULTIBAT.
4. Change the job limit on MULTIBAT to four.
5. Display the characteristics of the queue after changing the job limit to check that the command was executed properly.

Solutions

1. `$ SET QUEUE/SEPARATE=(FLAG, TRAILER) MYQUEUE`

   Substitute the name of your queue for MYQUEUE.

2. `$ PRINT/QUEUE=MYQUEUE FILE.TXT`

   Substitute the name of your file for FILE.TXT.

3. `$ INITIALIZE/QUEUE/BATCH/START/JOB_LIMIT=2 MULTIBAT`

4. `$ SET QUEUE/JOB_LIMIT=4 MULTIBAT`

5. `$ SHOW QUEUE/FULL MULTIBAT`

MANAGING QUEUES

Laboratory Exercises

Part IV

1. Send two separate print jobs to your terminal execution queue, holding them until you release them.
2. Release the first job, then abort it.
3. Display the contents of the queue immediately after aborting your job.
4. Stop the queue using the /RESET qualifier. You now have complete control over the queue.
5. Release the next job being held on the queue.
6. Display the contents of the queue and note the status of the jobs.
7. Start the queue and observe the output on your terminal.
8. Stop and delete your terminal output queue and your batch queue, MULTIBAT, that you created in the previous laboratory exercise.
9. Set the output terminal back to its original characteristics. (You will need OPER and LOG_IO privileges to do this.)

Solutions

Substitute your file names, entry numbers, and queue names in the commands listed below.

1.
```
$ PRINT/QUEUE=MYQUEUE/HOLD FILE1.TXT
$ PRINT/QUEUE=MYQUEUE/HOLD FILE2.TXT
```

2.
```
$ SET ENTRY 369/RELEASE
$ STOP/ABORT MYQUEUE
```

3.
```
$ SHOW QUEUE MYQUEUE
```

   Note the status of the jobs on the queue. Also note that trailer pages, if specified, will be printed for a job that was aborted.

4.
```
$ STOP/QUEUE/RESET MYQUEUE
```

5.
```
$ SET ENTRY 369/RELEASE
```

   Notice that the job did not print, as the queue is stopped.

6.
```
$ SHOW QUEUE MYQUEUE
```

7.
```
$ START/QUEUE MYQUEUE
```

8.
```
$ STOP/QUEUE/NEXT MYQUEUE
$ DELETE/QUEUE MYQUEUE
$ STOP/QUEUE/NEXT MULTIBAT
$ DELETE/QUEUE MULTIBAT
```

9.
```
$ SET DEVICE/NOSPOOL TTC5:
$ SET TERMINAL/PERMANENT/TYPE_AHEAD/BROADCAST TTC5:
```

   Substitute the device name of your output terminal for TTC5.

MANAGING DISK AND TAPE VOLUMES

Written Exercises

User PERDUTO has inadvertently deleted all of the files in his directory and subdirectories. Perduto's files were in directory [PERDUTO] and its subdirectories, located on device DRA1. Your system uses a command procedure that runs each night and backs up files from DRA1 to magnetic tape, using the following command:

```
$  BACKUP/VERIFY/IMAGE/JOURNAL=SYS$MANAGER:BACKUP.JOURNAL -
             DRA1:  MFA0:WORK_DISK.BACKUP/SAVE_SET/REWIND/INITIALIZE
```

a. What command would you use to recover the information that has seemingly been lost?

b. (OPTIONAL) What other method could you have used to recover the lost information?

Solutions

a. This command creates the subdirectory tree structure and restores all of the files in [PERDUTO] and its subdirectories.

```
$ BACKUP  MFA0:WORK_DISK.BACKUP/SAVE_SET/SELECT=[PERDUTO...]*.*;* -
  DRA1:[PERDUTO...]*.*;*
```

b. The most common way to recover (or partially recover) files that have been lost through inadvertent deletion is by restoring files that have previously been backed up. Using this process, you can restore those files that existed at the time of the most recent system backup. However, if your system has VAX RMS Journaling and if after–image journaling had been used for any data files that were deleted, you can use the RMS Recovery Utility to restore the files and include all modifications that were made up to the point where the file was deleted.

MANAGING DISK AND TAPE VOLUMES

Laboratory Exercises

Part I

You will need a disk drive and a disk volume or a tape drive and a tape volume to complete these exercises.

1. Do the following to initialize and mount a disk volume or a tape volume for public use.

   a. Allocate the device specified by your instructor to your process.

   b. Load your volume on the device. If you are not sure how to load the volume, ask your instructor.

   c. FOR DISK VOLUMES ONLY:
   Mount the disk volume using the /FOREIGN qualifier.

   d. FOR DISK VOLUMES ONLY:
   If your disk is one for which the BAD utility is valid (in other words, not an RA-series disk), invoke the BAD utility to search for bad blocks on the disk volume.

   e. FOR DISK VOLUMES ONLY:
   Dismount, but do not unload, the disk volume.

   f. Initialize the disk or tape volume so that all users will have access to it.

   g. Mount the disk or tape volume so that all users on the system will have access to it without having to mount it themselves.

2. Display the characteristics of the device on which you have just mounted the volume.

3. Dismount the volume and deallocate the device so that it is available for other users.

Solutions

Substitute the name of the device you are using for DRA1:, your own logical name for DRIVE, and your own label for MYVOL.

1. Enter the following commands to prepare a volume for public use:

   a. `$ ALLOCATE DRA1: DRIVE`

   b. No solution needed.

   c. `$ MOUNT/FOREIGN DRIVE`
   (For disk volumes only)

   d. `$ ANALYZE/MEDIA/EXERCISE DRIVE`
   (For disk volumes only)

   e. `$ DISMOUNT/NOUNLOAD DRIVE`
   (For disk volumes only)

   f. `$ INITIALIZE/SYSTEM DRIVE MYVOL`

   g. `$ MOUNT/SYSTEM DRIVE MYVOL`

2. `$ SHOW DEVICE/FULL DRIVE`

3. `$ DISMOUNT DRIVE`
   `$ DEALLOCATE DRA1:`

MANAGING DISK AND TAPE VOLUMES

Laboratory Exercises

Part II

You will need two disk drives and two disk volumes to complete these exercises. If you do not have the necessary materials to complete the lab, write out the answers on paper and compare them to the solutions.

1. Do the following to create a public volume set using two disk volumes.

   a. Allocate the devices you will be using.

   b. Load the volumes onto their respective drives. If you are not sure how to do this, ask your instructor.

   c. Initialize the volumes.

   d. Mount the volumes using the /SYSTEM and /BIND qualifiers.

2. Create a user directory on the volume set.

3. Copy a file from your default directory to your directory on the volume set.

4. Display the contents of your directory on the volume set.

5. Dismount the volume set.

Solutions

Substitute the names of your devices, and your own logical names and labels for the names and labels shown in these solutions.

1. Enter the following commands to create a volume set for public use:

   a.
```
$ ALLOCATE DRA1: DEV1
$ ALLOCATE DRA2: DEV2
```

   b. No solution needed.

   c.
```
$ INITIALIZE/SYSTEM DEV1 USER1
$ INITIALIZE/SYSTEM DEV2 USER2
```

   d.
```
$ MOUNT/SYSTEM/BIND=USER_SET DEV1,DEV2 USER1,USER2
```

2.
```
$ CREATE/DIRECTORY USER_DISK:[BROWN]
```

3.
```
$ SET DEFAULT SYS$LOGIN
$ COPY TESTFILE.LIS
_To: USER_DISK:[BROWN]*.*
```

4.
```
$ DIRECTORY USER_DISK:[BROWN]
```

5.
```
$ DISMOUNT DEV1
```

MANAGING DISK AND TAPE VOLUMES

Laboratory Exercises

Part III

You will need a tape drive and tape volume to complete these exercises.

1. Create three subdirectories of your default login directory. Call them [.A], [.B], and [.A.A2].

2. Copy at least two files into each of the above three subdirectories.

3. Allocate a tape drive. (If all drives are already allocated by other users, return to this exercise later.)

   Load your scratch tape into the drive. If you do not know how to load this drive, consult your instructor.

4. Initialize the scratch tape, then mount the tape as a foreign volume. Back up all the files in your directory and its subdirectories to a single save set on the tape.

5. List the contents of your new save set.

6. Delete one file in [.B] and selectively restore that file from your save set on tape.

7. Delete all files in [.A] and [.A.A2] and the subdirectory files themselves. Issue the following DCL commands:

```
$ DELETE [username.A]*.*;*,[username.A.A2]*.*;*
$ BACKUP  tape:save_set_name  disk:[username...]
```

   Note the results.

   What command would you use to correctly restore your subdirectories?

8. Dismount the tape drive. Issue a REPLY command to inform other students that the tape drive is now available.

Solutions

Substitute your own directory and device names for the ones shown in these solutions. You can also choose your own tape label and your own file name for the save set.

1.

```
$ CREATE/DIRECTORY [MYNAME.A]
$ CREATE/DIRECTORY [MYNAME.B]
$ CREATE/DIRECTORY [MYNAME.A.A2]
```

2.

```
$ COPY SYS$LOGIN:*.* [.A]
$ COPY SYS$LOGIN:*.* [.B]
$ COPY SYS$LOGIN:*.* [.A.A2]
```

3.

```
$ ALLOCATE WHOOSH$MUA0:
```

4.

```
$ INITIALIZE WHOOSH$MUA0: MYTAPE
$ MOUNT/FOREIGN WHOOSH$MUA0:
$ BACKUP [...]*.*;* WHOOSH$MUA0:MYFILES.BCK /LABEL=MYTAPE
```

5.

```
$ BACKUP /REWIND /LIST WHOOSH$MUA0:MYFILES.BCK
```

6.

```
$ DELETE [MYNAME.B]LOGIN.COM;*
$ BACKUP/LOG WHOOSH$MUA0:MYFILES.BCK/SELECT=[MYNAME.B]LOGIN.COM [MYNAME.B]
```

7. The commands shown cause BACKUP to create an extra level of subdirectories. The correct way to restore all your files is to use a command like the following:

```
$ BACKUP  WHOOSH$MUA0:MYFILES.BCK  STUDENT$DISK:[*...]
```

8.

```
$ DISMOUNT WHOOSH$MUA0:
$ DEALLOCATE WHOOSH$MUA0:
$ REPLY/USERS "Tape drive WHOOSH$MUA0 is now available"
```

CUSTOMIZING THE SYSTEM

Laboratory Exercises

1. Print the STARTUP.COM file and look at the output (this requires the SYSPRV privilege).
2. Create TERMINALS.COM in your own directory. This procedure should set the permanent characteristics of the terminals on your system according to the following information.
   a. Assume that you have eight terminals.
   b. Set up some fast terminals and some slow terminals.
   c. Protect at least three terminals from allocation by processes other than those with a [001,004] UIC.
   d. Four of the terminals are VT200–series terminals.
   e. Three are VT100s.
   f. One is a hard–copy terminal (LA120).
   g. The LA120 is attached through a modem.
   h. The other terminals are attached through direct lines.
   i. Include the name of the owner and office number where the terminal is located in a comment for each terminal.
3. Run the INSTALL utility:
   a. List the currently installed images.
   b. Choose one of the image names listed.
   c. Find out how many times the image has been invoked since it was installed.
   d. Send a message to all users indicating that you intend to remove that image from the installed list.
   e. Remove the image.
   f. List the installed images and see if it was removed.
   g. Reinstall the image as it was originally installed. (Display the contents of the SYS$MANAGER:VMSIMAGES.DAT file to see how it was originally installed.)
   h. Send a message to all users to inform them that the image has been reinstalled.

4. (ADVANCED EXERCISE) Create an alternate SYSTARTUP_V5.COM in your own directory. Do not execute this procedure to verify your work. Instead, look at the answers provided in this booklet. The procedure should include commands to do the following:

   a. Mount the class disk (label = CLASS) on the device DRA1: Be sure it will be accessible to all users. Assign the disk the logical name CLASS_DISK.

   b. Assume that a directory named PROGRAMS.DIR has been created on the system disk to contain site-specific programs. Define a system logical name for this directory.

   c. Restrict the number of interactive users to 35.

   d. Create and start the following queues (if possible). Be sure to start the queue manager first.

      - SYS$PRINT (generic queue)
      - LPA0 (print execution queue)

        – Make sure the printer supports lowercase characters and includes a flag page on each job printed.

      - FORM3 (logical queue)
      - SYS$BATCH (batch execution queue)

        – Set the job limit at two and the priority at three.

      - BIGJOB (batch execution queue)

        – Set the job limit at one and the priority at two.

        – Do not start this queue.

   e. Invoke SYS$MANAGER:TERMINALS.COM

   f. Define the logical names SYS$ANNOUNCE and SYS$WELCOME.

   g. Install the BACKUP.EXE program. Cause the directory information about the location of the program to remain permanently in memory. Cause the header of the BACKUP.EXE file to remain in memory.

   h. Change the default command interface for the INSTALL utility so the alternate command interface becomes the current interface.

   i. Rename the second highest version of the operator log to OPERATOR.OLD. Print OPERATOR.OLD, and have the system delete the file for you after it has been printed.

   j. Send a message to all terminals telling users that the system is now up and ready for use and log out.

Solutions

1. `$ PRINT SYS$SYSTEM:STARTUP.COM`

   The purpose of looking at the STARTUP.COM file is to familiarize yourself with the contents of this file as a whole.

2. A sample TERMINALS.COM file is shown below. This procedure meets the specifications of the problem, but it is not the only answer. If your procedure contains most of the same kinds of statements, it is probably also correct.

```
$!TERMINALS.COM
$!
$! This file sets up the permanent characteristics of the
$! terminals on this system.  This procedure is typically
$! invoked from SYSTARTUP_V5.COM.
$!
$!-------------------------------------------------------
$!
$ SET NOON
$!
$ SET TERM/PERM/NOMODEM-
/VT200/SPEED=300  TTA0:     !J.Smith   E15
$ SET TERM/PERM/NOMODEM-
/VT200/SPEED=2400 TTA1:     !N.Hae     E16
$ SET TERM/PERM/NOMODEM-
/VT200/SPEED=9600 TTA2:     !F.Chi     E17
$ SET TERM/PERM/NOMODEM-
/VT200/SPEED=9600 TTA3:     !P.Jones   E18
$ SET TERM/PERM/NOMODEM-
/VT100/SPEED=2400  TTA4:     !A.Steel   E19
$ SET TERM/PERM/NOMODEM-
/VT100/SPEED=9600  TTA5:     !J.Howland F01
$ SET TERM/PERM/NOMODEM-
/VT100/SPEED=9600  TTA6:     !M.Carter  F02
$ SET TERM/PERM/MODEM-
/AUTOBAUD/LA120    TTA7:     !D.Trevor  Dial-up
$!
$! NOTE: For the LA120 terminal, the speed is
$!  set automatically by /AUTOBAUD to 9600.
$!
$!Give [001,004] ownership of three terminals:
$!
$ SET PROT=(S,O:R,G,W)/DEVICE/OWNER_UIC=[001,004] TTA0:
$ SET PROT=(S,O:R,G,W)/DEVICE/OWNER_UIC=[001,004] TTA5:
$ SET PROT=(S,O:R,G,W)/DEVICE/OWNER_UIC=[001,004] TTA7:
$!
$! NOTE: All users can log in on these terminals, but users
$!       with a UIC of [001,004] can also allocate them.
```

3. `$ INSTALL`

   a. `INSTALL>LIST`

   b. Assume the DIRECTORY image is chosen.

   c. `INSTALL>LIST/FULL DIRECTORY`

      NOTE: The access count is initialized to 0 each time the image is installed. (Images are installed each time the system is started by the startup procedures.)

   d. `INSTALL>EXIT`
      `$ REPLY/BELL/ALL "I am about to de-install the DIRECTORY image"`

   e. `$ INSTALL/COMMAND_MODE`
      `INSTALL>DELETE DIRECTORY`

   f. `INSTALL>LIST`

      (You should not see the DIRECTORY image listed.)

   g. `INSTALL>CREATE/OPEN/HEADER/SHARED DIRECTORY`
      `INSTALL>LIST`

      (You should see the DIRECTORY image listed again.)

      `INSTALL>EXIT`

   h. `$REPLY/BELL/ALL "The DIRECTORY image has been re-installed"`

4. A sample SYSTARTUP_V5.COM procedure is shown below. This procedure meets the specifications of the problem, but it is not the only answer. If your procedure contains most of the same kinds of statements, it is probably also correct.

```
$! SYSTARTUP_V5.COM
$!
$!  This procedure sets up the system environment according
$!  to the resources available and functions performed on
$!  this system.
$!
$!-------------------------------------------------------------
$!
$ MOUNT /SYSTEM DRA1: CLASS CLASS_DISK
$!
$ ASSIGN/SYSTEM SYS$SYSDEVICE:PROGRAMS.DIR PROGRAMS
$!
$ SET LOGIN/INTERACTIVE=35
$!
$! NOTE: The SET LOGIN/INT command should be placed at the end
$!       of the procedure, so users cannot log in while the
$!       procedure is executing.
$!
$ START/QUEUE/MANAGER
$!
$!  Start up PRINT and BATCH queues
$!
$ INITIALIZE/QUEUE/GENERIC/START SYS$PRINT
$!
$ SET PRINTER LPA0:/LOWER
$ SET DEVICE LPA0:/SPOOLED
$!
$ INITIALIZE/QUEUE/FLAG/START LPA0
$!
$ INITIALIZE/QUEUE/NOENABLE_GENERIC FORM3   !You cannot start a
$!                                           logical queue until
$!                                           it has been assigned
$!                                           to a physical queue.
$!
$ INITIALIZE/QUEUE/BATCH/JOB=2/PRIORITY=3/START SYS$BATCH
$!
$ INITIALIZE/QUEUE/BATCH/JOB=1/PRIORITY=2 BIGJOB
$!
$ @SYS$MANAGER:TERMINALS.COM
$!
$ ASSIGN/SYSTEM "System number 239 - VAX/VMS " SYS$ANNOUNCE
$ ASSIGN/SYSTEM "Welcome to system 239" SYS$WELCOME
$!
$ RUN SYS$SYSTEM:INSTALL
SYS$SYSTEM:BACKUP /OPEN/HEADER
EXIT
$!
$ INSTALL :== $SYS$SYSTEM:INSTALL
$!
$ RENAME OPERATOR.LOG;-1 OPERATOR.OLD
$ PRINT/DELETE OPERATOR.OLD
$!
$ REPLY/BELL/ALL "System 239 is ready for use"
$!
$LOGOUT/BRIEF
```

STARTING UP AND SHUTTING DOWN THE SYSTEM

Laboratory Exercises

Part I

1. Under the supervision of your instructor, look at the front panels of as many VAX processors as are available to you. Become familiar with the items on them.

2. Your instructor will shut the system down for you and will turn the power off. (If he or she cannot shut the power off, go to problem 3. It is really not a good practice to shut the power off and on very often, as it can lead to hardware problems. However, if this is only done occasionally, it should not harm the system.)

3. Start the system from power on, following the instructions in the *VMS Installation and Operations* manual for this type of processor.

4. If your instructor allows you to, and if your system has battery backup, set up your system for an automatic restart and turn the power off. Start the system from power off. (The system should have been running before you began this problem and you should have been logged in.)

5. Your instructor will shut the system down for you. Start the system, specifying the system device explicitly.

Solutions

1. No solution needed.
2. No solution needed.
3. No solution needed.
4. No solution needed. If you set up your system properly, it should start and you should be able to log in.
5. No solution needed.

STARTING UP AND SHUTTING DOWN THE SYSTEM

Laboratory Exercises

Part II

Do each problem in this lab under the **direct** supervision of your instructor.

1. Your instructor will shut the system down for you. The console terminal is now in program mode. Log in at the console terminal.

2. While you are logged in at the console terminal, change to console mode. Do not enter any console commands.

3. Return to program mode and enter the DIRECTORY command.

4. Disable the use of console mode. Try to enter console mode at the console terminal.

5. If this system boots by means of boot files, copy the default boot file from the console volume to your directory and look at it. Notice that it boots the default system device on your system (or it should).

Solutions

1. No solution needed.

2. When you see the >>> prompt, you know you have been successful.

3. When you see the $ prompt and can enter the DIRECTORY command, you know you are in program mode.

4. You should not be able to enter console mode if you set the switch correctly.

   Enable console mode again when you are done with this problem.

5. You must be in program mode to copy a file from the console volume. Use the SYS$UPDATE:DXCOPY command procedure. If your default boot file, DEFBOO.CMD, does not boot your default system device, inform your instructor.

STARTING UP AND SHUTTING DOWN THE SYSTEM

Laboratory Exercises

Part III

1. Your instructor will create an alternate copy of the SYS$SYSTEM:SYSUAF.DAT file and will modify your account in the new file. The alternate copy is named SYS$SYSTEM:SYSUAFALT.DAT. Start the system conversationally, specifying the alternate UAF file. Log in and enter the SHOW PROCESS command to verify that the alternate file was used. The display should contain the new values entered by your instructor. Check with your instructor to verify this.

2. Your instructor will shut the system down for you. Start it conversationally, specifying the normal SYS$SYSTEM:SYSUAF.DAT file as the UAF file. Log in and enter the SHOW PROCESS command. The display should look normal.

Solutions

1. Boot the system conversationally. Enter the commands SET UAFALTERNATE 1 and CONTINUE at the SYSBOOT> prompt.

2. Boot the system conversationally. Enter the commands SET UAFALTERNATE 0 and CONTINUE at the SYSBOOT> prompt.

STARTING UP AND SHUTTING DOWN THE SYSTEM

Laboratory Exercises

Part IV

You must perform the following exercises on a standalone system under the supervision of your instructor. You must start the system up after each exercise.

1. Use the orderly shutdown method to shut down your system.
2. Use the emergency DCL shutdown method to shut down your system.
3. Force a shutdown using the CCL method.

Solutions

1. Enter the command:

   ```
   @SYS$SYSTEM:SHUTDOWN
   ```

   and respond to the prompts.

2. Follow these steps:

   a. Log in as SYSTEM on the console terminal.

   b. Issue the DCL command:

      $ RUN SYS$SYSTEM:OPCCRASH

   c. When the SYSTEM SHUTDOWN COMPLETE message is displayed, type:

      CTRL/P
      ```
      >>>H
      ```

3. See the *VMS Installation and Operations* manual for this type of system.

STARTING UP AND SHUTTING DOWN THE SYSTEM

Laboratory Exercises

Part V

Complete the following exercises only under the supervision of your instructor.

The purpose of this optional lab is to show you that different errors can occur at different points during the startup process. Some errors indicate a problem with the console volume. It could be corrupted, missing, or it could have been replaced by a scratch or data volume. Other errors indicate a problem with the system disk. It could be corrupted, off–line, write–locked, or certain files necessary for the startup process could be missing or corrupted. Finally, errors could indicate problems with data disks, buses, memory, or other parts of the system.

You can usually identify the problem by looking at the listing on the console terminal. Certain errors occur at certain points in the startup process. For example, if the console volume is missing, you will not even get to the part of the listing where the system volume is read. Or, if you cannot get to the part where the console volume is read, perhaps the problem lies in memory or some part of the CPU.

If you are having problems starting up your system, first check to be sure you are preparing your system properly for startup. If you still cannot start it, show the console terminal listings to a more experienced person or to a DIGITAL Field Service representative.

1. Shut the system down using the normal method.

2. Remove the console volume. (If this is not possible on your system, go to question 9. If you are working on a VAX–11/750 system, also set the BOOT DEVICE switch to A.)

NOTE

If your console volume is a diskette, do not remove it if you hear a clicking noise from the drive. The noise indicates that the system is using the volume. If you remove the volume while the system is using it, you can damage the volume.

3. Attempt to boot the system by using the boot switch on the front panel or by entering a command at the console terminal.

4. What are the results?

5. Place a scratch volume in the console drive.

6. Attempt to boot the system by using the boot switch on the front panel or by entering a command at the console terminal.

7. What are the results?

8. Replace the console volume in the console drive.

9. Spin down the system disk.

10. Attempt to boot the system by using the boot switch on the front panel or by entering a command at the console terminal.

11. What are the results?

12. Spin up the system disk and write protect it.

13. Attempt to boot the system by using the boot switch on the front panel or by entering a command at the console terminal.

14. What are the results?

15. Write protect a data disk that your site–specific command procedure normally loads.

16. Attempt to boot the system by using the boot switch on the front panel or by entering a command at the console terminal.

17. What are the results?

Solutions

1. Use the SHUTDOWN.COM procedure.

2. No solution needed.

3. No solution needed.

4. You should receive an error message at the console terminal. The content of this message differs between processors, but its meaning is the same. The meaning is that the system cannot boot without the console volume.

5. No solution needed.

6. No solution needed.

7. You should receive a different error message at the console terminal than the one you received when no volume was loaded. The meaning of this second message is that the system cannot boot without a proper system console volume.

8. No solution needed.

9. No solution needed.

10. No solution needed.

11. You should see more messages on the console terminal this time before you see an error message. The meaning of this third error message is that the system cannot read the system volume when it is not spun up. It must be able to read the system volume to complete the startup process.

12. No solution needed.

13. No solution needed.

14. You should see an error message on the console terminal. The meaning of this message is that the startup process cannot complete unless it can both read from and write to the system disk.

15. No solution needed.

16. No solution needed.

17. The startup process will complete if a data disk is not loaded, but you will receive error messages at the console terminal. You should always enter the SHOW DEVICES command after the system is up to check on the status of all disk drives, or you should read the console listing to be sure all were successfully mounted.

INSTALLING AND UPDATING SYSTEM SOFTWARE

Laboratory Exercises

1. Print the SYS$MANAGER:SYSTARTUP_V5.COM file on your system and look for the command that is used to start the network.

2. Obtain a copy of the set of instructions for installing Version 5.0 on your system. Under the supervision of your instructor, follow these instructions.

3. Under the supervision of your instructor, follow the instructions in the *VMS Release Notes* for Version 5.0 to upgrade your system (running Version 4.6 or later) to a Version 5.0 system.

4. Under the supervision of your instructor, install a maintenance update on your system volume by following the instructions in the release notes for the update. Be sure the update you are installing is the *next sequential* update for the volume, as you *must* apply them in sequential order.

5. After you have installed, upgraded, or updated your system, run the User Environment Test Package by following the instructions in the installation guide for your processor.

6. Under the supervision of your instructor, install one optional product on your system by following the instructions in the product's installation guide.

7. Under the supervision of your instructor, use the SYS$UPDATE:CONSCOPY.COM command procedure to create a spare copy of the console volume.

8. Perform a standalone BACKUP from disk to disk. (If your system has only one disk, perform a disk–to–tape BACKUP.) This exercise requires your exclusive use of the computer for 30 to 60 minutes. Inform the instructor that you want to perform this exercise. If time and circumstances are not favorable, you may have to perform it at a different time. If you are forced to delay this exercise until later, continue with the rest of the module, but return to this exercise as soon as possible.

    a. Obtain a standalone BACKUP kit on console media from your instructor. If no kit is available, create one. (If your instructor prefers, create a standalone BACKUP kit on the system disk instead. If you use a kit on the system disk, leave the normal console volume containing the boot software in the console drive and ignore the statements about the console volume in this exercise.)

    b. Ask your instructor for a scratch disk (or set of scratch tapes) and a sample data disk (or sample system disk) to back up. You will be copying up the entire data (or system) disk to the scratch disk (or set of tapes).

    c. At the time designated by the instructor, shut down the computer using the SYS$SYSTEM:SHUTDOWN.COM procedure. (The **Starting Up and Shutting Down the System** module discusses this command procedure.)

d. Once the system is fully shut down, boot the standalone BACKUP program.

e. Issue the command to perform the BACKUP. Be sure to include the /VERIFY qualifier.

NOTE

Before issuing any commands to the BACKUP utility, be sure you know which disk is to be copied (the input disk) and which is the output disk. It is a good idea to write-lock the input disk. Also, have the instructor check both volumes and your BACKUP command before you press the RETURN key to enter the command.

f. Once the BACKUP operation is finished, unload the two scratch and sample packs used for the exercise and reload (if necessary) the normal system and data packs.

g. Press the CTRL/P key sequence at the console terminal to prepare to boot the system. Halt the computer if necessary.

h. Replace the BACKUP kit console media with the console media containing the boot software.

i. Boot the VMS system from power on. (If you had copied the system disk to an output disk, you might want to load the output disk and try to boot from it. The **Starting Up and Shutting Down the System** module discusses how to boot the system from the default device or from a device you specify.)

Solutions

1. The command is @STARTNET.

2. If you followed the instructions carefully, you should be able to boot your system and log in. If you cannot do this, consult your instructor.

3. After you have completed the procedure, you should be able to boot the system and log in. If not, consult your instructor.

4. If you have followed the instructions carefully, you should be able to boot, log in to, and use your updated system. If you cannot do this, consult your instructor.

5. If you have followed the instructions carefully when installing, upgrading, or updating your system, the tests should be successful. If the tests are not successful, consult your instructor.

6. If you have followed the instructions carefully, the Installation Verification Procedure (IVP) done by the product's installation procedure should be successful. If the procedure does not include an IVP, test the product yourself. If the tests are not successful, consult your instructor.

7. Run the SYS$UPDATE:CONSCOPY.COM command procedure to save the contents of the console volume. Then load a scratch volume in the console device and use the same procedure to restore the contents from the system disk to the scratch volume.

   If you followed the instructions carefully, you should be able to boot the system using your copy of the console volume.

8. If you performed a disk–to–disk backup of a data disk, and you can mount the output copy and reference files and directories, you were successful.

   If you did a disk–to–disk backup of a system disk, and you can boot the new disk, you were successful.

   If you did a disk–to–tape backup, you should be able to restore the tapes to a scratch disk and reference files on it or boot it (depending on the contents of the original disk). The **Managing Disk and Tape Volumes** module discusses how to restore tapes to a scratch disk.

MAINTAINING SYSTEM INTEGRITY

Laboratory Exercises

Part I

1. Set your default to your login directory.
2. Copy the file SYS$SYSTEM:SYSUAF.DAT to your own directory.
3. Run the AUTHORIZE utility. Using AUTHORIZE, perform the following exercises:
   a. Display in brief format all the records in the UAF that have user names beginning with the letter S.
   b. Display in brief format all the records in the UAF that have UIC group number 20.
   c. Display the contents of your own UAF record.
   d. Display the contents of the DEFAULT record.
   e. Modify the DEFAULT record so that the GROUP and GRPNAM privileges are automatically given to all new accounts.
   f. Modify the DEFAULT record to increase the ENQLM limit to 30.
   g. Modify the DEFAULT record so that the value of the disk device in the default field is the same as the name of your default disk device. This can be either a real device name or a logical name equating to the real device name. (You can display the name of your default device by entering the SHOW DEFAULT command at the DCL prompt.)
   h. Display the contents of the DEFAULT record to verify your changes.
   i. Create a new UAF record with the user name SMITH and a group number of 322. Remember to specify a password for SMITH, but do not specify any other values.

      NOTE

      Ignore any informational or error messages you receive about the rights database.

   j. Display the contents of the new record.
   k. Compare the values of the fields in the new record with the values of the fields in the DEFAULT record. What are your conclusions?
   l. Create a new UAF record in group 322 for user JONES. Use the COPY command to create a record for user DOE. Assign each user a unique member number, password, owner, and default directory.

m. Display all the UAF records in JONES' group using the brief format. Use a wildcard in the command.

n. Modify account JONES to limit the login time for the primary days MON, TUE, WED, THU from 8 a.m. to 4 p.m. and secondary days FRI, SAT, SUN from 4 pm to 11 pm. Examine the record.

o. Modify account JONES to limit dial–in capability from 6 p.m. to 10 p.m. for secondary days. Establish default privileges of TMPMBX, NETMBX, SYSPRV and an additional authorized privilege of VOLPRO. Examine the record.

p. Disable user SMITH from logging in.

q. Delete the UAF record for user SMITH.

r. Generate a brief listing and a full listing of all user records. Print the listing file on a printing device.

Solutions

1. `$ SET DEFAULT SYS$LOGIN`

2. `$ COPY SYS$SYSTEM:SYSUAF.DAT *`

3. `$ RUN SYS$SYSTEM:AUTHORIZE`

   a. `UAF> SHOW /BRIEF S*`

   b. `UAF> SHOW /BRIEF [20,*]`

   c. `UAF> SHOW username`

   d. `UAF> SHOW DEFAULT`

   e. Enter the following commands at the UAF> prompt:

      ```
      UAF>MODIFY DEFAULT/PRIVILEGE=(GROUP,GRPNAM)-
      _UAF>/DEFPRIVILEGE=(GROUP,GRPNAM)
      ```

      You can record two sets of privileges in each UAF record. One set contains the authorized privileges (PRIVILEGE); the other set contains the default privileges (DEFPRIVILEGE). Users receive the default privileges when they log in. Users can give themselves any of their authorized privileges by using the SET PROCESS/PRIVILEGE command.

   f. `UAF>MODIFY DEFAULT /ENQLM=30`

   g. `UAF>MODIFY DEFAULT /DEVICE=CLASS_DISK`

      Substitute the name of the device you are using for CLASS_DISK.

   h. `UAF> SHOW DEFAULT`

   i. `UAF>ADD SMITH /PASSWORD=ENIGMA /UIC=[322,1]`

      You can change your password and UIC if you choose.

   j. `UAF>SHOW SMITH`

   k. The values of the fields in SMITH should be identical to the values of the same fields in DEFAULT, with the exception of the user name and UIC fields. Notice, also, that SMITH has the GROUP and GRPNAM privileges.

l.
```
UAF>ADD JONES /UIC=[322,2] /OWNER="Jane Jones" -
_UAF>/DIRECTORY=[JONES]
UAF>COPY JONES DOE /UIC=[322,3] /OWNER="Ken Doe" -
_UAF>/DIRECTORY=[DOE]
```

The password for these accounts is USER, since no password was indicated when the records were created. Note that each user has a unique member number, but that all are members of the same group.

m.
```
UAF>SHOW/BRIEF [322,*]
```

n.
```
UAF>MODIFY JONES /PRIMEDAYS=(NOFRI,NOSAT,NOSUN) -
_UAF>/ACCESS=(PRIMARY, 8-3, SECONDARY, 16-22)
UAF>SHOW JONES
```

o.
```
UAF>MODIFY JONES /DIALIN=(SECONDARY, 18-21) /DEFPRIVILEGES= -
_UAF>(TMPMBX,NETMBX,SYSPRV) -
_UAF>/PRIV=(TMPMBX,NETMBX,SYSPRV,VOLPRO)
UAF>SHOW JONES
```

p.
```
UAF>MODIFY SMITH /FLAG=DISUSER
```

q.
```
UAF>REMOVE SMITH
```

r.
```
UAF> LIST/BRIEF*
UAF> LIST/FULL*
UAF> EXIT
$ PRINT SYSUAF.LIS;*
```

MAINTAINING SYSTEM INTEGRITY

Laboratory Exercises

Part II

Pair up with another student to do this exercise. You must have write access to the user authorization file to do this exercise. You may also need the SYSPRV privilege.

1. Log in using your own account. Set your default to SYS$SYSTEM and run the AUTHORIZE utility. Display a list of all identifiers in the system rights list.

2. Create an identifier called READERn, where n is a number assigned to you by your instructor.

3. Exit from the utility. Set your default back to your login directory.

4. Create a file called TEXT.TXT. With the ACL editor, create an ACE for the identifier READERn in the ACL of your file. In the ACE, specify that holders of the READERn identifier will be allowed to read the file. Also set the UIC protection code on the file to allow Group and World users no access.

5. Type the file TEXT.TXT on your terminal screen. Note that you have access to the file even though you do not hold the identifier READERn.

6. Ask your partner to try to type the file.

7. Run the AUTHORIZE utility. Grant the READERn identifier to your partner's account. Show the UAF record for your partner's account and note the identifers that have been assigned to it.

8. Ask your partner to log out and then log back in, and to try typing the file again. Now have he or she try to edit the file.

9. Display a list of all the identifiers held by your account.

10. Create a file containing a list of all the identifiers in the system rights list.

Solutions

Substitute your assigned identifier for READER1 in the following solutions.

1.
```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF>SHOW /IDENTIFIER *

BATCH
LOCAL
INTERACTIVE
    .
    .
    .
```

2.
```
UAF>ADD /IDENTIFIER READER1
```

3.
```
UAF>EXIT
$ SET DEFAULT SYS$LOGIN
```

4. You may create the text file in the following manner:

```
$ CREATE TEXT.TXT
This is a test file.
CTRL/Z
```

   To invoke the ACL editor, type the following:

```
$ EDIT/ACL TEXT.TXT
```

   You should receive a prompt of the form (IDENTIFIER= . Enter the following after this prompt:

```
READER1,OPTIONS=NONE,ACCESS=READ)
```

   Then, type CTRL/Z to exit from the editor. (Pressing RETURN will prompt you for another identifier.)

   Finally, to set the protection on the file so that Group and World users have no access, enter the command:

```
$ SET PROTECTION=(G,W) TEXT.TXT
```

5. You have access to the file because you are the owner, and System and Owner UIC–based protection overrides ACL–based protection.

6. Your partner will get an error message when trying to type the file, since your partner's account does not hold the READER identifier, and Group and World users have no access to the file according to the UIC–based protection code that you defined.

7.
```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF>GRANT/IDENTIFIER READER1 STUDENT5
UAF>SHOW STUDENT5
```

   The utility lists the identifiers held at the bottom of the display.

   (Substitute your partner's user name for STUDENT5 in this example.)

8. Your partner should be able to read the file but should not be able to edit the file because neither the identifier nor UIC–based protection allow he or she to have write access to the file.

9.
```
UAF> SHOW /RIGHTS /USER=username
```

10.
```
UAF> LIST /IDENTIFIER *
```

MAINTAINING SYSTEM INTEGRITY

Laboratory Exercises

Part III

NOTE

You must have write access to the system authorization file and the SECURITY privilege to do this lab.

1. Create a secondary password for your UAF record. Log out and log in. What has changed?

2. Create a system password and set the appropriate characteristics of your terminal so it is a system–password–required terminal. Log out and log in. What has changed?

3. Make your terminal more secure by setting the SECURE_SERVER characteristic on it. Log out and log in. What has changed?

4. Create another UAF record. Set the expiration date on the new UAF record to be five minutes after the current time. Wait at least five minutes. Attempt to log in to the new account. What happens? Log in to your own account and delete the extra UAF record.

5. Change the minimum length of the password to 16 on your own account, set a flag to require password generation, exit from the utility, log out and log in.

6. Change your password with a DCL command that uses one of the choices listed by the generator (or generate a new list).

7. Use the SET AUDIT and SHOW AUDIT commands to define and display various types of audit.

8. Log out and attempt to log in several times, specifying an incorrect password each time. Observe the output on the console terminal.

Solutions

1.
```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF> MODIFY JONES/PASSWORD=(" ",SECRET)
```

   The system displays a second password prompt when you log in.

2. Use the SET PASSWORD/SYSTEM command to create a system password. Use the SET TERMINAL command to modify your terminal characteristics as shown in the module. When you log in to your account after setting these up, you must enter the system password before you see the prompt for your user name.

3. Use the SET TERMINAL/SECURE_SERVER command as shown in this module to make your terminal more secure. This time, you must press the BREAK key and enter a system password before you see the prompt for your user name.

4. You will not be able to log in to the new account because it has expired.

5. Use the /FLAGS=GENPWD qualifier to the MODIFY command in the AUTHORIZE utility to do this.

6. When you enter the SET PASSWORD command, it automatically generates a list of passwords for you to choose from.

7. No answer needed.

8. No answer needed.

MAINTAINING SYSTEM INTEGRITY

Laboratory Exercises

Part IV

1. Use the SDA utility to generate a report at your terminal about the last system failure.

   Find out and record the following:

   a. The name of the process whose state was CUR

   b. How many processes were active

   c. The reason for the failure (or recorded reason).

   d. The contents of general register 6

   e. The user names of processes that were in the COM state

   f. The name of the image file that was executing

2. Look in the correct system directory (use a logical name) to see if ERRLOG.OLD exists.

   If it does not exist, rename the current error log file to ERRLOG.OLD.

3. Use the ERF utility to examine the contents of ERRLOG.OLD. If the file is very large, use the /SINCE qualifier to examine only the section for the last few days. If no entries are listed for the requested event, enter a less recent date or examine the entire file.

   a. List all memory error entries in brief format at your terminal.

   b. List all CPU–related entries in full format at your terminal.

   c. List all errors recorded for the device where your class disk is loaded in summary format.

   d. List all entries recorded because of device mounts and dismounts. Send the listing to the system line printer.

4. If you have never seen an SPR form, and the instructor has made one available, take the time to look at it during this lab period. Take note of the types of information requested, so if a problem occurs, you will be more likely to remember to obtain the needed information.

5. Is it necessary to complete the entire form for all problems?

6. What information should be sent with the form to DIGITAL? And why?

7. List the test files used by the UETP. Use a system logical name in the command.

Solutions

1.
```
$ SET PROCESS /PRIVILEGE=SYSPRV
$ ANALYZE /CRASH_DUMP SYS$SYSTEM:SYSDUMP
```

   a.
```
SDA> SHOW PROCESS/ALL
```

   b.
```
SDA> SHOW SUMMARY
```

   c.
```
SDA> SHOW CRASH
```

   d.
```
SDA> SHOW CRASH
```

   e.
```
SDA> SHOW SUMMARY
```

   f.
```
SDA> SHOW CRASH
```

   The stated reason for the system failure is not always the real reason. Another event may have actually caused the failure. Also, the system manager cannot routinely conclude that the current process at the time of the failure caused the failure.

   A system programmer can use the information gathered from the SDA utility to discover the cause of the problem. The information for a particular failure may not be very helpful, but when combined with other information collected over a period of time, the problem may be found.

   The job of the system manager is not to interpret the output to the SDA utility, but to collect the information. If the system fails repeatedly with the same stated reason, this MAY indicate either a software or hardware problem. The software problem may be caused by site-specific software or by system software. In either case, the output to the SDA utility should always be saved and stored in a central location. If the failures continue, and the DIGITAL representative has access to these output records, the cause for the failure will be easier to isolate and solve.

2. If ERRLOG.OLD did not exist in SYS$ERRORLOG: $RENAME ERRLOG.SYS ERRLOG.OLD NOTE: After this command has been entered, ERRLOG.SYS does not exist. If an event occurs that needs to be recorded, the ERRFMT process will create a new ERRLOG.SYS file.

3. Use the ANALYZE/ERROR_LOG command with the following qualifiers to generate the reports:

   a. /INCLUDE=MEMORY/BRIEF

   b. /INCLUDE=CPU_ENTRIES

      Full format is the default format.

   c. /INCLUDE=CLASS_DISK/SUMMARY

      Substitute the name of your class disk for CLASS_DISK.

   d. /INCLUDE=VOLUME_CHANGES/OUTPUT=LPA0:

4. No answer needed.

5. Most of the information requested on the form should be supplied for problems and errors.

6. Information such as listings, programs, related files, data files, number of users, version of software, etc. should be provided. Problems should be presented in a concise and complete manner. This will help the DIGITAL software engineer solve the problem.

7. `$ DIRECTORY SYS$TEST`

MAINTAINING SYSTEM INTEGRITY

Laboratory Exercises

Part V

These exercises are optional.

1. Modify your SYSTARTUP_V5.COM file to run the SDA utility and analyze the system failure. Send the output from the utility to a file. Include a command to print the file.

2. Create a command procedure to rename the error file every day at midnight. The procedure should save old versions. Old versions should have names that correspond to the day they were created, such as FEB23. The procedure should run the ERF utility to generate a summary report of all errors on a daily basis and print the report.

Solutions

1. See the SDA VMS V5 dump file in Module 8.

2. A suggested command procedure follows. Log in as SYSTEM or be sure you have SYSPRV before executing this. To be sure this is run daily, modify the SYSTARTUP_V5.COM file and include the line:

```
$ SUBMIT TRACKERR /AFTER="23:59:59.99"
```

If an error occurs during the execution of this procedure, a mail message is automatically sent to the SYSTEM account.

```
$!
$!TRACKERR.COM
$!
$!-----------------------------------------------
$ON ERROR THEN GOTO ERROR_HANDLER
$!
$TIME = "''F$TIME()'"
$!
$SET DEFAULT SYS$ERRORLOG
$!
$RENAME ERRLOG.SYS ERRLOG.OLD
$!
$ ANALYZE/ERROR_LOG/SUMMARY/OUTPUT=REPORT.DAT
$!
$DAY = F$EXTRACT (0,2,TIME)
$!
$MONTH = F$EXTRACT (3,3,TIME)
$!
$FILE = "''MONTH'''DAY'.DAT"
$!
$RENAME ERRLOG.OLD 'FILE'
$!
$REPORT = "''MONTH'''DAY'.RPT"
$!
$RENAME REPORT.DAT 'REPORT'
$!
$PRINT 'REPORT'
$!
$GOTO RESUBMIT_AND_EXIT
$!
$ERROR_HANDLER:
$!
$MAIL/SUBJECT="TRACKERR.COM has failed" NL: SYSTEM
$!
$RESUBMIT_AND_EXIT:
$!
$SUBMIT/AFTER="23:59:59.99" TRACKERR.COM
$!
$EXIT
```

MONITORING THE SYSTEM

Laboratory Exercises

Part I

1. Using a video terminal, run the MONITOR utility to display the MONITOR> prompt. At the prompt, enter a command to the utility to display the PAGE class with an update interval of ten seconds. Allow the utility to run for five to ten minutes. Watch the display and notice how (if you are on an active system) the display changes with time.

2. Return to the MONITOR> prompt and display the SYSTEM class.

3. Return to the MONITOR> prompt and display the IO class.

4. Use MONITOR to create a file called SUMMARY.DAT that contains summary information about the PROCESSES class. Allow the utility to write several screens of information to the file before you enter the CTRL/Z key sequence to return to the DCL prompt. (Each time the values in the display change, the utility writes a copy of the display to the file.) Display the file on your terminal screen after you exit from the utility.

5. Return to the MONITOR> prompt and display any other classes that you want. Use the HELP facility in the MONITOR utility to list the names of the classes and possible qualifiers.

6. Use the SHOW USERS command to obtain a list of the interactive users on the system. Use the SHOW SYSTEM command to obtain a list of the processes on the system. Compare the names of the interactive users with the list of process names. Note which processes appear in one but not the other.

7. Use the SHOW PROCESS/CONTINUOUS command to examine your own process. What image does the program indicate you are running?

Solutions

1.

```
$ MONITOR
MONITOR>  MONITOR PAGE
```

2.

```
CTRL_C
MONITOR>  MONITOR SYSTEM
```

3.

```
CTRL_C
MONITOR>  MONITOR IO
```

4.

```
CTRL_C
MONITOR>  MONITOR PROCESS/SUMMARY=SUMMARY.DAT
CTRL/Z
$ TYPE SUMMARY.DAT
```

5. Enter the CTRL/C key sequence to cancel any display and return to the MONITOR> prompt.

```
$ MONITOR
MONITOR>  HELP MONITOR
```

6. There should be more processes listed by SHOW SYSTEM than by SHOW USERS. SHOW SYSTEM lists the system processes, interactive processes, processes from batch queues, and subprocesses. SHOW USERS lists interactive users attached to terminals.

7. SHOW PROCESS/CONTINUOUS shows that you are executing the image SYS$SYSROOT:[SYSEXE]SHOW.EXE.

MONITORING THE SYSTEM

Laboratory Exercises

Part II

In these exercises, you use a command procedure called HARMLESS_AUTOGEN.COM. Your instructor should tell you in what directory to find this procedure.

HARMLESS_AUTOGEN mimics the actions of SYS$UPDATE:AUTOGEN.COM, except that it does not create or modify any system files or shut down the system. Instead, it creates files in your default login directory (SYS$LOGIN:).

1. Create a file MODPARAMS.DAT in your default directory SYS$LOGIN: (not in SYS$MANAGER:). Have it contain values for the following:

   - Preserve the values of parameters SCSNODE and SCSSYSTEMID for this system (use the values that appear in SYS$SYSTEM:MODPARAMS.DAT).
   - Set LRPSIZE so that each large request packet (LRP) of nonpaged pool is 512 bytes.
   - Set VIRTUALPAGECNT to allow a program to use 20 megabytes (40960 pages) of virtual memory.
   - Set MAXSYSGROUP to decrease the maximum UIC group number for a system user to four.
   - Set RJOBLIM to allow only four remote terminals to log in concurrently.
   - Increase the number of global sections by six, and the number of global pages by 300.

2. Run HARMLESS_AUTOGEN so that it starts at the earliest possible phase and ends with the phase that reports on system file sizes.

3. Examine SYS$LOGIN:AGEN$FEEDBACK.REPORT and SYS$LOGIN:SETPARAMS.DAT. (You may want to print them.) Verify that AUTOGEN generated correct settings for the parameters that you specified in MODPARAMS.DAT. If settings are incorrect, edit MODPARAMS.DAT and perform the previous step again.

4. Run HARMLESS_AUTOGEN so that it starts by generating system parameter values and ends by rebooting the system. (It will not cause the system to reboot, but will display a message at the point where AUTOGEN would have rebooted the system.)

5. Use the SYSGEN utility to examine SYS$LOGIN:AUTOGEN.PAR. For the parameters that you specified in MODPARAMS.DAT, verify that AUTOGEN set them correctly in AUTOGEN.PAR.

6. Still using this AUTOGEN.PAR file, generate a listing file showing the values of the MAJOR parameters. Call the file MAJOR.LIS. Print the file.

Solutions

1. Your file should be similar to the following:

```
! This is a sample MODPARAMS.DAT
!
SCSNODE = "PANAMA"
SCSSYSTEMID = 2278
LRPSIZE = 512
MIN_VIRTUALPAGECNT = 40960 ! Use MIN in case AUTOGEN calculates a higher value
MAXSYSGROUP = 4
RJOBLIM = 4
ADD_GBLSECTIONS = 6
ADD_GBLPAGES = 300
```

2. `$ @dir:HARMLESS_AUTOGEN SAVPARAMS TESTFILES`

3. No solution needed.

4. `$ @dir:HARMLESS_AUTOGEN GENPARAMS REBOOT`

5.

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> USE AUTOGEN.PAR
SYSGEN> SHOW SCSNODE
SYSGEN> SHOW SCSSYSTEMID
SYSGEN> SHOW LRPSIZE
   .
   .
   .
```

6.

```
SYSGEN> SET /OUTPUT=MAJOR.LIS
SYSGEN> SHOW /MAJOR
SYSGEN> EXIT
$ PRINT MAJOR.LIS
```

MONITORING THE SYSTEM

Laboratory Exercises

Part III

1. To observe how the ACCOUNTING utility works, perform the following:
   a. Find out what classes of accounting have been enabled.
   b. Disable the recording of accounting data on print jobs.
   c. Send a file to the printer.
   d. Enable the recording of accounting on print jobs.
   e. Send a file to the printer.
   f. Generate an ACCOUNTING report in brief format that shows all print jobs completed within the last hour. Observe the contents of the report.
2. Create an accounting report in full format that shows all print jobs completed within the last hour. Send the report directly to the printer. Collect the output and observe the format and contents of the report.
3. Use the ACCOUNTING utility to examine the following record types:
   a. Login failures (in brief format)
   b. Interactive job terminations (in brief format)
   c. Process terminations (in brief format)
   d. System initializations (in full format)
4. Use the ACCOUNTING utility to display a summary report of the accounting records that have your user name.
5. Create a report in summary format containing the data collected on direct I/O from interactive processes today. List the data in order by UIC, and send the report directly to the printer. (This can all be done in the same command.)
6. Create a summary report that shows how many batch jobs have completed in each hour.
7. Find out whether anyone has tried and failed to log in to the SYSTEM account.

Solutions

1. Enter the following commands to observe how the ACCOUNTING utility works:

   a. `$ SHOW ACCOUNTING`

   b. `$ SET ACCOUNTING/DISABLE=(PRINT)`

   c. `$ PRINT FILE.TXT`

      Substitute the name of your file for FILE.TXT.

   d. `$ SET ACCOUNTING/ENABLE=(PRINT)`

   e. `$ PRINT FILE.TXT`

   f. `$ ACCOUNTING/TYPE=PRINT/SINCE=14:00`

      Substitute an appropriate time for 14:00. Note that the report contains a brief description of your print jobs (if they have already completed) as well as descriptions of other jobs.

2. `$ ACCOUNTING/FULL/TYPE=PRINT/SINCE=14:00/OUTPUT=LPA0:`

   Substitute an appropriate time for 14:00 and the device name of your line printer for LPA0:. The colon (:) indicates that LPA0: is a device and not a file specification. If the colon is omitted, the output from the report is stored in a file named LPA0.LIS.

3. Enter the following commands to use the ACCOUNTING utility:

   a. `$ ACCOUNTING/TYPE=LOGFAIL`

   b. `$ ACCOUNTING/PROCESS=INTERACTIVE`

   c. `$ ACCOUNTING/TYPE=PROCESS`

   d. `$ ACCOUNTING/FULL/TYPE=SYSINIT`

4. `$ ACCOUNTING /SUMMARY=USER /REPORT=RECORDS /USER=SMITH`

   Substitute your user name for SMITH.

5. ```
   $ ACCOUNTING /SUMMARY=UIC -
   _$ /REPORT=DIRECT_IO /SINCE=00:00 /OUTPUT=LPA0:
   ```

   Substitute the device name of your line printer for LPA0: The information in the report is organized in order by UIC.

6. `$ ACCOUNTING /TYPE=BATCH /SUMMARY=HOUR`

7. `$ ACCOUNTING /USER=SYSTEM /TYPE=LOGFAIL`
   ```

MODULE 11
POST-TEST

TEST

Write the letter of the best answer in the blank next to each question.

1. _____If you have installed the VMS operating system, what do you have to do to install a new version containing minor changes and corrections to the system code?

   a. Upgrade the VMS operating system

   b. Install the VMS operating system from scratch

   c. Update the VMS operating system

   d. Reboot the system

2. _____Which of the following is stored in the User Authorization File?

   a. Process disk quotas

   b. Process device drivers

   c. Process privileges

   d. Process library names

3. _____Which utility is used to locate defective blocks on a disk pack?

   a. ANALYZE/DISK_STRUCTURE

   b. BAD

   c. SYSMAN

   d. EXCHANGE

4. _____Which of the following can you create using the AUTHORIZE utility?

   a. ACL

   b. File protection code

   c. Identifier

   d. ACE

5. _____Which of the following would you use to cause the VMS system to record events such as process deletion, print job completion, login failure and batch job completion?

   a. SET ACCOUNTING

   b. SET ACCOUNTING/RECORD

   c. ACCOUNTING

   d. ACCOUNTING/RECORD

6. _____Which of the following should be used to upgrade or update system software?

   a. SYS$UPDATE

   b. VMSINSTAL

   c. SYSGEN

   d. INSTALL

7. _____Which of the following should be used to examine a system failure dump file?

   a. ERF

   b. SYSGEN

   c. SDA

   d. UETP

8. _____Which of the following commands displays (on one screen) several of the most important classes of information a system manager can monitor?

   a. $ MONITOR SCS

   b. $ MONITOR SYSTEM

   c. $ MONITOR STATES

   d. $ MONITOR ALL_CLASSES

9. _____Which of the following control break-in detection?

   a. Symbols

   b. Identifiers

   c. Logical names

   d. System parameters

10. _____Which of the following commands creates a user record and an identifier?

   a. UAF> ADD/IDENTIFIER

   b. UAF> ADD

   c. $ EDIT/ACL

   d. $ CREATE/DIR/IDENTIFIER

11. _____Which of the following commands is recommended to stop a running system?

   a. $ RUN OPCCRASH

   b. >>>@CRASH

   c. $ @SHUTDOWN

   d. $ RUN SYSGEN

12. _____What file do you edit to make parameter changes permanently known to AUTOGEN?

   a. SYS$MANAGER:VMSIMAGES.DAT

   b. SYS$UPDATE:AUTOGEN.DAT

   c. SYS$SYSTEM:SETPARAMS.DAT

   d. SYS$SYSTEM:MODPARAMS.DAT

13. _____If you receive a Files–11 tape from a user who requests you to print one of the files on it, which of the following DCL commands must you issue first?

    a. MOUNT

    b. INITIALIZE

    c. ASSIGN

    d. COPY

14. _____Which utility is used to make shareable files available to all users?

    a. SYSGEN

    b. SDA

    c. ERF

    d. INSTALL

15. _____Which DCL command is used to create a queue?

    a. CREATE/QUEUE

    b. START/QUEUE

    c. ASSIGN/QUEUE

    d. INITIALIZE/QUEUE

16. _____When installing a system, which of the following files can be of any size, regardless of the amount of memory on the system?

    a. Page file

    b. Swap file

    c. Dump file

    d. Parameter file

17. _____What is the function of a Software Performance Report (SPR)?

a. Collect and display software performance information

b. Analyze software performance information

c. Report software problems to DIGITAL

d. Report software problems to system users

18. _____Which utility is invoked during a conversational startup?

a. SYSGEN

b. SYSBOOT

c. VMSINSTAL

d. SDA

19. _____Which of the following accounts cannot be used to log in on a newly installed VMS system?

a. SYSTEM

b. FIELD

c. DEFAULT

d. SYSTEST

20. _____In which directory would you normally find on-line diagnostic files?

a. SYS$MANAGER

b. SYS$MAINTENANCE

c. SYS$UPDATE

d. SYS$SYSTEM

21. _____Which utility is used to generate an error log report?

    a. SDA

    b. MONITOR

    c. ERF

    d. UETP

22. _____In which file are terminal speeds, queues, and other system management related operations usually specified?

    a. SYLOGIN.COM

    b. SYSTARTUP_V5.COM

    c. STARTUP.COM

    d. SYLOGICALS.COM

23. _____Which procedure should mount site–specific volumes?

    a. SYSTARTUP_V5.COM

    b. STARTUP.COM

    c. SYLOGICALS.COM

    d. SYCONFIG.COM

24. _____Which procedure should define standard logical names such as SYS$LOGIN, SYS$ANNOUNCE, and SYS$WELCOME?

    a. SYLOGICALS.COM

    b. SYCONFIG.COM

    c. SYSTARTUP_V5.COM

    d. STARTUP.COM

25. _____In which table are shareable logical name tables cataloged?

a. LNM$SYSTEM_TABLE

b. LNM$GROUP

c. LNM$JOB

d. LNM$PROCESS_TABLE

26. _____Which DCL command is used to enable an operator's terminal?

a. REPLY

b. SET TERMINAL

c. SET PROCESS

d. REQUEST

27. _____Before an active queue can be deleted, it must first be:

a. Initialized

b. Stopped

c. Spooled

d. Started

ANSWERS

1. __c__ If you have installed the VMS operating system, what do you have to do to install a new version containing minor changes and corrections to the system code?

   a. Upgrade the VMS operating system

   b. Install the VMS operating system from scratch

   c. Update the VMS operating system

   d. Reboot the system

2. __c__ Which of the following is stored in the User Authorization File?

   a. Process disk quotas

   b. Process device drivers

   c. Process privileges

   d. Process library names

3. __b__ Which utility is used to locate defective blocks on a disk pack?

   a. ANALYZE/DISK_STRUCTURE

   b. BAD

   c. SYSMAN

   d. EXCHANGE

4. __c__ Which of the following can you create using the AUTHORIZE utility?

   a. ACL

   b. File protection code

   c. Identifier

   d. ACE

5. a Which of the following would you use to cause the VMS system to record events such as process deletion, print job completion, login failure and batch job completion?

   a. SET ACCOUNTING

   b. SET ACCOUNTING/RECORD

   c. ACCOUNTING

   d. ACCOUNTING/RECORD

6. b Which of the following should be used to upgrade or update system software?

   a. SYS$UPDATE

   b. VMSINSTAL

   c. SYSGEN

   d. INSTALL

7. c Which of the following should be used to examine a system failure dump file?

   a. ERF

   b. SYSGEN

   c. SDA

   d. UETP

8. b Which of the following commands displays (on one screen) several of the most important classes of information a system manager can monitor?

   a. $ MONITOR SCS

   b. $ MONITOR SYSTEM

   c. $ MONITOR STATES

   d. $ MONITOR ALL_CLASSES

9. __c__ Which of the following control break-in detection?

   a. Symbols

   b. Identifiers

   c. Logical names

   d. System parameters

10. __b__ Which of the following commands creates a user record and an identifier?

   a. UAF> ADD/IDENTIFIER

   b. UAF> ADD

   c. $ EDIT/ACL

   d. $ CREATE/DIR/IDENTIFIER

11. __c__ Which of the following commands is recommended to stop a running system?

   a. $ RUN OPCCRASH

   b. >>>@CRASH

   c. $ @SHUTDOWN

   d. $ RUN SYSGEN

12. __d__ What file do you edit to make parameter changes permanently known to AUTOGEN?

   a. SYS$MANAGER:VMSIMAGES.DAT

   b. SYS$UPDATE:AUTOGEN.DAT

   c. SYS$SYSTEM:SETPARAMS.DAT

   d. SYS$SYSTEM:MODPARAMS.DAT

13. __a__ If you receive a Files–11 tape from a user who requests you to print one of the files on it, which of the following DCL commands must you need to issue first?

    a. MOUNT

    b. INITIALIZE

    c. ASSIGN

    d. COPY

14. __d__ Which utility is used to make shareable files available to all users?

    a. SYSGEN

    b. SDA

    c. ERF

    d. INSTALL

15. __d__ Which DCL command is used to create a queue?

    a. CREATE/QUEUE

    b. START/QUEUE

    c. ASSIGN/QUEUE

    d. INITIALIZE/QUEUE

16. __d__ When installing a system, which of the following files can be of any size, regardless of the amount of memory on the system?

    a. Page file

    b. Swap file

    c. Dump file

    d. Parameter file

17. __c__ What is the function of a Software Performance Report (SPR)?

a. Collect and display software performance information

b. Analyze software performance information

c. Report software problems to DIGITAL

d. Report software problems to system users

18. __b__ Which utility is invoked during a conversational startup?

a. SYSGEN

b. SYSBOOT

c. VMSINSTAL

d. SDA

19. __c__ Which of the following accounts cannot be used to log in on a newly installed VMS system?

a. SYSTEM

b. FIELD

c. DEFAULT

d. SYSTEST

20. __b__ In which directory would you normally find on-line diagnostic files?

a. SYS$MANAGER

b. SYS$MAINTENANCE

c. SYS$UPDATE

d. SYS$SYSTEM

21. __c__ Which utility is used to generate an error log report?

a. SDA

b. MONITOR

c. ERF

d. UETP

22. __b__ In which file are terminal speeds, queues, and other system management related operations usually specified?

a. SYLOGIN.COM

b. SYSTARTUP_V5.COM

c. STARTUP.COM

d. SYLOGICALS.COM

23. __a__ Which procedure should mount site-specific volumes?

a. SYSTARTUP_V5.COM

b. STARTUP.COM

c. SYLOGICALS.COM

d. SYCONFIG.COM

24. __a__ Which procedure should define standard logical names such as SYS$LOGIN, SYS$ANNOUNCE, and SYS$WELCOME?

a. SYLOGICALS.COM

b. SYCONFIG.COM

c. SYSTARTUP_V5.COM

d. STARTUP.COM

25. __a__ In which table are shareable logical name tables cataloged?

a. LNM$SYSTEM_TABLE

b. LNM$GROUP

c. LNM$JOB

d. LNM$PROCESS_TABLE

26. __a__ Which DCL command is used to enable an operator's terminal?

a. REPLY

b. SET TERMINAL

c. SET PROCESS

d. REQUEST

27. __b__ Before an active queue can be deleted, it must first be:

a. Initialized

b. Stopped

c. Spooled

d. Started