Educational Services

VMS System Management I Student Workbook Volume II

EY-3505E-SB-0003

9.10 Laboratory Exercises

In these exercises, you use a command procedure called HARMLESS_AUTOGEN.COM. This file can be found in a directory pointed to by the logical name COURSE$V5SYSMGT.

HARMLESS_AUTOGEN mimics the actions of SYS$UPDATE:AUTOGEN.COM, except that it does not create or modify system files or shut down the system. Instead, it creates files in your default login directory (SYS$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.
   - Run HARMLESS_AUTOGEN so that it starts at the earliest possible phase and ends with the phase that reports on system file sizes.

July 1989

Contents

CUSTOMIZING THE SYSTEM

5.1 Introduction

Essentially, every VMS system is a customized system. Not only does the choice of hardware vary between installations, but this hardware may be configured many ways. Module 1 covers many of these hardware options, including many of the software choices for VMS systems. From these choices, you configure your system and install software to meet user needs. Because standard hardware and software may not entirely satisfy user needs, you must also customize the software on your system by:

- Creating user accounts
- Establishing disk quotas
- 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 cover many of the skills you use to customize the VMS system 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 the user. Using all of the skills you learn in this course, you can also customize the system to improve its performance and efficiency. Module 9 discusses performance improvement in greater detail.

5.2 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:

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

5.3 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 Guide* for your particular VAX system
5. *VMS DCL Dictionary*

5.4 The User Environment

Two principal methods establish the default environment for all processes:

- Setting system parameters
- Issuing DCL commands

System parameters, which affect all processes, are usually set by the system manager. These parameters control the structure of the VMS operating system. The system generally uses the parameter values assigned before or during the startup procedure. You can change some parameters while the system is running, and the new values will be used. However, to use a new value for most parameters, you must reboot the system. More information about modifying system parameters is presented in Modules 6 and 9.

DCL commands are issued from the system startup files each time the system is initialized. Like system parameters, they also affect all processes. The system manager controls the contents of the system startup files. This module discusses commands typically executed in the system startup files.

Other DCL commands are issued from the system and user login command files each time a user logs in or when a batch job runs. These commands further customize the process environment. The system manager controls the contents of the system login files. Users control the contents of their own login file. (Login files are discussed in Module 2.)

You can issue DCL commands to affect all processes interactively (such as initializing queues or creating logical names). Unless you include the DCL commands in the startup or login files, you must repeat them each time the system is started and/or each time a user logs in. Table 5–1 describes the files that automatically establish the VMS system environment at system startup and at process creation.

Table 5–1 Files Controlling the VMS System Environment

System File	Information	When Used
SYS$SYSTEM:VAXVMSSYS.PAR	System parameters	System startup
SYS$SYSTEM:STARTUP.COM	DCL commands	System startup
SYS$MANAGER:SYPAGSWPFILES.COM	DCL commands and possibly SYSGEN commands	System startup
SYS$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

5.5 System Startup Files

When the VMS operating system is initialized, five startup command procedures execute in the following order:

1. SYS$SYSTEM:STARTUP.COM (site-independent)
2. SYS$SYSTEM:SYPAGSWPFILES.COM (site-specific)
3. SYS$SYSTEM:SYCONFIG.COM (site-specific)
4. SYS$SYSTEM:SYLOGICALS.COM (site-specific)
5. SYS$MANAGER:SYSTARTUP_V5.COM (site-specific)

The VMS operating system automatically invokes the command procedure STARTUP.COM at system initialization. STARTUP.COM then invokes the other procedures. All site-specific files are located in SYS$MANAGER and have a prefix of SY to distinguish them from other system startup files.

5.5.1 Site-Independent Startup File: STARTUP.COM

The startup files listed previously are distributed as part of the VMS operating system. The file STARTUP.COM contains commands necessary to initialize any VMS system. The other files, which are initially empty, are intended to contain site-specific commands (commands that depend upon your particular system configuration). Later sections of this module contain detailed descriptions of the site-specific command procedures.

All major releases, and some minor releases of the VMS operating system delete and replace STARTUP.COM. If a command in STARTUP.COM does not create the environment you want, it is better to override that command with a command in SYCONFIG.COM, or SYSTARTUP_V5.COM, than to change the command in STARTUP.COM itself.

STARTUP.COM uses a series of component files that accomplish many functions, such as:

- Assigning logical names required by certain VMS system software
- Assigning logical names to the VMS system directories
- Starting up system processes such as JOB_CONTROL, OPCOM, and ERRFMT
- Installing known images
- Connecting all standard devices
- Calling the site-specific startup command procedures

These component files are located in a set of directories associated with the system-wide logical name SYS$STARTUP.

STARTUP.COM starts up the system in four basic phases. Three data files located in SYS$STARTUP are involved in the phased startup:

- VMS$PHASES.DAT (**do not modify**)
- VMS$VMS.DAT (**do not modify**)
- VMS$LAYERED.DAT

Never modify VMS$PHASES.DAT or VMS$VMS.DAT. These files contain important site-independent information that can change from one VMS system release to the next. The data file VMS$LAYERED.DAT, managed through the SYSMAN utility, contains information about the various layered software products purchased for your system. For more information on these component files, see the *Guide to Setting Up a VMS System.*

5.5.2 Installing Paging and Swap Files: SYPAGSWPFILES.COM

SYS$MANAGER:SYPAGSWPFILES.COM is the first site-specific startup file invoked by STARTUP.COM. It is used to install page and swap files on disks other than the system disk.

Before invoking SYPAGSWPFILES.COM, STARTUP.COM first activates the following files (if they exist) in SYS$SYSTEM:

- PAGEFILE.SYS
- SWAPFILE.SYS
- SYSDUMP.DMP

After these files are activated, SYPAGSWPFILES.COM is invoked, allowing the system manager to direct the system to activate other paging and swap files on other disks. You might consider creating additional paging and swap files if you have a large system with lots of memory and/or processes, or if you want to distribute paging and swap file activity across multiple disks.

To activate additional paging and swap files, place whatever commands are needed for the installation, including:

- INITIALIZE
- MOUNT
- SYSGEN

When SYPAGSWPFILES.COM completes execution, control is returned to STARTUP.COM.

5.5.3 Configuring Devices: SYCONFIG.COM

SYS$MANAGER:SYCONFIG.COM is the second site-specific startup file invoked by STARTUP.COM. SYCONFIG.COM is used to connect special devices to the system and load their I/O drivers.

You can configure some devices or all devices from this file. Generally you configure only nonstandard (special) devices within this command file. SYSGEN commands are typically placed within this file, although you might also use MOUNT and SET DEVICE commands to completely configure devices.

After SYCONFIG.COM completes execution, STARTUP.COM executes the SYSGEN command AUTOCONFIGURE ALL which automatically configures all known devices attached to the system according to parameters set in the SYSGEN parameter file. You can suppress this autoconfiguration by setting the following symbol within SYCONFIG.COM:

```
$ STARTUP$AUTOCONFIGURE_ALL == 0
```

If you set this symbol, the CONFIGURE section of STARTUP.COM does not execute. You must ensure that all devices are properly configured within SYCONFIG.COM.

5.5.4 Defining System-Wide Logical Names: SYLOGICALS.COM

SYS$MANAGER:SYLOGICALS.COM, the third site-specific command file invoked by STARTUP.COM, is used to define system-wide logical names. STARTUP.COM also defines system-wide logical names, so you only have to define those logical names specific to your particular site. Table 5–2 lists some of the logical names commonly defined in SYLOGICALS.COM for any given VMS operating system site.

Table 5–2 Some Standard Logical Names to Define in Site-Specific Startup File

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 V5.2`")

To create system-wide logical names, use one of the DCL commands shown in Table 5–3.

Table 5–3 Assigning System Logical Names

Operation	Command Format/Example†

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`

†Requires SYSNAM privilege or a system UIC

Typically you would use the /NOLOG qualifier with either DEFINE or ASSIGN to reduce the amount of printout on the console terminal during system startup.

Security considerations require you to define some system-wide logical names as **executive-mode** logical names. Whenever you decide to define a system-wide logical name, consider its use by privileged images, or in situations when the integrity of the system could be compromised by incorrect logical names.

To define an executive-mode logical name, use the following command format:

```
$ DEFINE/SYSTEM/EXECUTIVE  logical-name  equivalence-name
$ DEFINE/SYSTEM/EXECUTIVE  CLASSIFICATIONS  SECURE:[ADMIN]CLASS.DAT
```

It is **strongly** recommended that executive-mode logical names be used for system components and files. For example:

- Public disks and directories
- SYSUAF.DAT
- RIGHTSLIST.DAT
- VMSMAIL.DAT
- NETPROXY.DAT

One executive-mode logical name commonly defined within SYLOGICALS.COM is SYSUAF, a pointer to the site user authorization file (SYSUAF.DAT):

```
$ DEFINE/SYSTEM/EXECUTIVE/NOLOG  SYSUAF  SYS$SYSTEM:SYSUAF.DAT
```

Defining the SYSUAF logical name this way allows you to access SYSUAF.DAT (using the Authorize utility) without having to set SYS$SYSTEM as your default directory. Other examples of logical name definitions are shown in Example 5–1.

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

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

For a complete discussion of logical names and the commands DEFINE and ASSIGN, refer to the *VMS DCL Dictionary*.

5.5.5 General Site-Specific Startup Functions: SYSTARTUP_V5.COM

The final site-specific command procedure invoked by STARTUP.COM is SYS$MANAGER:SYSTARTUP_V5.COM.

> NOTE
> Prior to VMS Version 5, this site-specific command file was called STARTUP.COM. SYSTARTUP_V5.COM performs the same functions as STARTUP.COM, with the exception of those functions performed by the site-specific command files.

SYSTARTUP_V5.COM is used to accomplish functions not covered by the other site-specific command files, such as:

- Mounting public disks
- Setting device characteristics
- Initializing and starting batch and print queues
- Installing known images
- Starting up DECnet software
- Analyzing the most recent system failure
- Purging unwanted operator log files
- Starting up the LAT network

- Defining the maximum number of interactive users
- Announcing that the system is up and available
- Allowing users to log in

Each of these functions is discussed later in this module.

Many different DCL and utility commands are used to perform the functions listed above. A common mistake made by a new system manager is to place all the necessary commands into SYSTARTUP_V5.COM. This results in a single, lengthy command file. It is better to create SYSTARTUP_V5.COM as a "shell" procedure that invokes one separate, independent command file for each startup function. The benefits of this startup approach are:

- You only need to edit a small command file to change the effects of any startup function.
- Accidental editing is significantly minimized.
- Modifications are easier to document. (Include an "edit history" in the associated function command file.)
- Functions can be separately executed after the system is started.

There are times during normal system operation when you need to execute the commands found within a particular function command file. One example of this is when, for some reason, the queue manager function of the JOB_CONTROL process must be restarted. When this happens, you need to restart the print and batch queues. If you put all the initialization and startup functions for system queues in a separate command file, you only execute that particular command file to restart your entire queue system. (This subject is discussed in Module 3.)

Example 5–2 shows a sample SYSTARTUP_V5.COM procedure. The sections following the example discuss the procedures invoked from this sample SYSTARTUP_V5.COM file. For more information on system startup files, see the *Guide to Setting Up a VMS System.*

5.5.5.1 Mounting Public Disks

Mount the volumes that should be brought on-line each time the system is initialized (using the DCL command MOUNT) in the site-specific startup procedure. These volumes usually include all public volumes (volumes accessible to all users and controlled by the system manager). Module 4 describes public volume management.

In Example 5–3, commands from the sample MOUNTDSK.COM procedure mount two public volumes. The commands include the /SYSTEM qualifier to make these volumes available to all users and to store the specified logical name in the system logical name table. By default, logical names for devices are created with the concealed and terminal attributes. Because of the concealed attribute, DCL commands display the logical name for the device, not the real device name. This gives system managers the ability to load a volume in any available device, without confusing the users, who simply specify the logical name.

For example, users might be accustomed to seeing the device name WORK1 displayed as their default device name. If the manager moves their volume to another drive, but still assigns it the name WORK1, the users do not know that it was moved. They can continue to refer to the volume using the same name as before. Because of the terminal attribute, the system does not try to translate the real device name when it is returned to the equivalence string. Instead, it terminates translation activities, thereby saving time. More information on the MOUNT command can be found in Module 4 or in the *VMS DCL Dictionary*.

When you issue the MOUNT command in SYSTARTUP_V5.COM (or a procedure invoked from SYSTARTUP_V5.COM), the qualifier /NOASSIST is the default, rather than the usual /ASSIST default. This change enables the system to start up even if you have not physically mounted all public volumes.

```
$ 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
$ !
$ ! Initialize and start print and batch queues
$ @STARTUP_PROCS:QUEUES.COM
$ !
$ ! Install known images
$ @STARTUP_PROCS:INSTALL.COM
$ !
$ ! Start DECnet
$ @SYS$MANAGER:STARTNET.COM
$ !
$ ! Start LAT network
$ @SYS$MANAGER:LTLOAD.COM
$ !
$ ! Create reports about the last system failure
$ @STARTUP_PROCS:REPORT_FAILURE.COM
$ !
$ ! Purge old versions of operator log files
$ PURGE/KEEP=3 SYS$MANAGER:OPERATOR.LOG
$ !
$ ! 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:SYSTARTUP_V5.COM
$ EXIT
```

Example 5–2 SYSTARTUP_V5.COM Command Procedure

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

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

5.5.5.2 Setting Device Characteristics

The VMS operating system supports terminals that have a variety of physical characteristics. When STARTUP.COM configures the VAX system hardware, it determines the number of terminal lines the system can support, then sets each terminal's permanent characteristics according to the defined terminal system parameters. Modules 6 and 9 discuss system parameter modifications. Detailed information about terminal system parameters can be found in *VMS System Generation Utility Manual.*

DCL commands must be added to the SYSTARTUP_V5.COM file to alter the permanent characteristics associated with particular terminals. Table 5–4 shows how to use the DCL command SET TERMINAL/PERMANENT to set permanent terminal characteristics.

For more information on setting terminal characteristics, see the command description for SET TERMINAL/PERMANENT in the *VMS DCL Dictionary*. Although users without special privileges can change the characteristics of their own terminals, the VMS operating system resets the permanent characteristics again when they log out.

Table 5–4 Setting Permanent Characteristics of Terminals

	SET TERMINAL/PERMANENT	
Terminal Characteristic	**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`
		The VMS operating system 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 of which specifies one characteristic.

†Requires LOG_IO or PHY_IO privilege.
‡To list the available device types, enter the HELP SET TERMINAL/DEVICE_TYPE command.

Terminals are not the only devices whose characteristics can be set in SYSTARTUP_V5.COM. For example, printers, tape drives, and card readers also have device characteristics. More information on setting device characteristics can be found in the command descriptions for SET PRINTER, SET MAGTAPE, and SET CARD_READER in the *VMS DCL Dictionary*.

Devices that are not shareable, such as terminals, have an owner UIC and a protection code that determine what processes can allocate the device. Table 5–5 shows how to use the DCL command SET PROTECTION/DEVICE to control device ownership and protection.

At most sites, the system manager owns all terminals (UIC=[SYSTEM]) and the protection code is typically set so that only users in the SYSTEM or OWNER protection category can allocate terminals.

Users in the SYSTEM category are those whose UIC is a system UIC. Users in the OWNER category are those whose UIC is [SYSTEM]. Users with other UICs cannot allocate these protected terminals. By restricting the ability to allocate a terminal, you prevent users from displaying the Username: and Password: prompts from a program to collect user passwords and violate the security of the system. However, this protection code does not prevent users from logging in to these terminals.

For more information on setting device protection, see the command description for SET PROTECTION/DEVICE in the *VMS DCL Dictionary*.

In Example 5–4, commands from the sample TERMINALS.COM file set the characteristics of terminals and other devices.

Table 5–5 Establishing Ownership and Protection of Terminals and Other Non-Shareable Devices

Operation†	**Command Examples (Require 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.

†By default, all terminals have the owner specified by the system parameter TTY_OWNER, and the protection specified by the system parameter TTY_PROT.

```
$ 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
```

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

5.5.5.3 Initializing and Starting Queues

Create a separate, site-specific procedure containing commands to initialize and start all print and batch queues for your system. Then, as shown in Example 5-2, invoke that procedure from SYSTARTUP_V5.COM.

5.5.5.4 Installing Known Images

When you install an image, the VMS operating system stores information about the image in system memory. The reasons for installing images include:

- Reducing the time required to activate the image
- Enabling the image to execute with privileges that the process running the image lacks
- Promoting the sharing of physical memory
- Adding a user-written system service

Installing an image is a totally different task from installing the VMS operating system and its optional software products. When you install the VMS operating system and optional products, you copy files to a disk. When you install an image, you copy information to memory about a file on a disk. An installed image is called a **known image** because the system knows information (stored in memory) about the image that it usually would not know until it is activated.

NOTE
Do not confuse the two different uses of the word "install": to install known images (discussed here) or to install VMS operating system or optional software, (see Module 7).

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

You install images using commands in the Install utility. To invoke the Install utility, enter the following command:

```
$ INSTALL
```

Example 5-5 shows a sample installation procedure, INSTALL.COM. Table 5-6 shows how to use the Install utility to install and manage known images. Table 5-7 lists the qualifiers used to specify image installation parameters.

If an image you want to install is stored on a disk volume other than the system disk, mount the volume that contains the image first. For more information on installing known images, see the section on the Install utility in the *VMS Install Utility Manual.*

The standard shutdown procedure removes all files installed by STARTUP.COM. You must explicitly remove any other files by invoking the Install utility from the site-specific SYSHUTDWN.COM procedure. Otherwise, you will not be able to dismount the disks that contain the installed files. Module 6 contains detailed information on the SYSHUTDWN.COM procedure.

```
$!
$! 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
!
```

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

Table 5–6 Functions of the Install Utility

Operation	Command Format	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

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`

†Specifying a version number will produce unpredictable results when you use the known image. Therefore, never specify a version number in the file specification.

Table 5–7 INSTALL Command Qualifiers

Qualifier	Description
/ACCOUNTING	Activates image level accounting for the installed image. (Discussed in Module 9).
/EXECUTE_ONLY	Installs 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	Lists 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 kernel 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, so that only one copy of those sections has to be in physical memory. The image is implicitly declared permanently open (/OPEN).
/WRITEABLE	Shared 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 to it). The image must also be declared /SHARED.

5.5.5.5 Starting Local DECnet

The following command starts local DECnet (if you have purchased DECnet for your system):

```
$ @SYS$MANAGER:STARTNET.COM
```

STARTNET.COM is a standard command procedure provided with the VMS operating system. Include this command in the SYSTARTUP_V5.COM procedure. (See Example 5–2). After you establish local DECnet communications in this manner, users can execute the command SET HOST on your system if they specify your system as the target host.

If your system is part of a large DECnet network, you can significantly decrease the time it takes to start up your system by executing the STARTNET.COM procedure as a batch job (instead of invoking it directly from within SYSTARTUP_V5.COM). Place the following command line in SYSTARTUP_V5.COM (instead of the command shown above):

```
$ SUBMIT SYS$MANAGER:STARTNET.COM
```

5.5.5.6 Starting the LAT Network

The following command starts the LAT terminal server network software:

```
$ @SYS$MANAGER:LTLOAD.COM
```

LTLOAD.COM is a command file provided with the VMS operating system. Include this command in the SYSTARTUP_V5.COM procedure, as Example 5–2 shows. After you establish LAT network communications, users can connect to your system through LAT servers, or access other services you have defined for your LAT network. For more information on starting up a LAT network on a VMS system, see the *Guide to Setting Up a VMS System.* Detailed information about LAT software and networks can be found in the *VMS LAT Control Program (LATCP) Manual.*

5.5.5.7 Creating Reports About the Last System Failure

Sudden system failures can result from a power loss, hardware errors, or software errors. You can also force a system failure by using the console subsystem. (Module 6 discusses how to intentionally cause the system to fail.)

Regardless of the reasons for system failure, the contents of memory are written to the dump file before the system stops running. You can analyze the contents of this file when you restart the system.

Analyze the dump file with the System Dump Analyzer utility (SDA). Invoke the utility with the command ANALYZE/CRASH_DUMP, and create a listing of the information in the file using SDA commands. (Module 8 discusses the SDA utility in more detail.)

Since the system may fail again during, or immediately after the startup procedure, you should include SDA commands to create the listing file when SYSTARTUP_V5.COM executes. Example 5-6 lists the command file invoked by the SYSTARTUP_V5.COM command file in Example 5–2.

```
$ !
$ !  User-written procedure to
$ ! analyze the most recent crash.
$ ! Invoked from SYSTARTUP_V5.COM.
$ !
$ ! Copy the dump file and create listing
$ !
$ ANALYZE/CRASH_DUMP SYS$SYSTEM:SYSDUMP.DMP
   COPY LIB$DISK:[SYSMGR.CRASH]SYSDUMP.DMP        ! Save dump file
   SET OUTPUT LIB$DISK:[SYSMGR.CRASH]SYSDUMP.LIS  ! Create listing from:
    SHOW CRASH                        ! Display crash information
    SHOW STACK/ALL                    ! Show current stack
    SHOW SUMMARY                      ! List active processes
    SHOW PROCESS/PCB/PHD/REGISTERS    ! Display all current processes
    SHOW SYMBOL/ALL                   ! Display system symbol table
   EXIT
$ !
$ ! Get rid of old system dump files
$ PURGE/KEEP=2 LIB$DISK:[SYSMGR.CRASH]*.*
$ PRINT LIB$DISK:[SYSMGR.CRASH]SYSDUMP.LIS
$ EXIT
```

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

5.5.5.8 Purging Unwanted Versions of Operator Log Files

Each time the system is rebooted, a new version of OPERATOR.LOG is created. It is usually unnecessary to keep more that one version of this log file on-line. Determine how many versions to keep on-line and purge the remaining versions. As shown in the sample SYSTARTUP_V5.COM (Example 5–2), use the PURGE/KEEP command to delete unwanted versions:

```
$ PURGE/KEEP=3 SYS$MANAGER:OPERATOR.LOG
```

5.5.5.9 Setting the Maximum Number of Interactive Logins

The VMS operating system is distributed with a default interactive login value of 64. This means that up to 64 users can be logged in to your system. You can change this value by setting a special symbol within SYSTARTUP_V5.COM (or in any command procedure called by it). For example, to set the maximum number of interactive logins to 40, use the following command:

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

5.5.5.10 Announcing System Availability

Use the DCL REPLY command to announce the availability of the system. Include various qualifiers depending on who you want to notify. Typically, as in Example 5-7, you want to notify all users, so you would include the /BELL and /ALL qualifiers. (Module 2 contains a detailed description of the REPLY command.) REPLY is usually the last command in the site-specific startup file. After this command is executed, you will see a logout message on the console terminal. At this point, you can log in.

Some system managers like to log in, before other users have access to the system. They check logical names and system devices to be sure everything is set up correctly. For this reason, the sample SYSTARTUP_V5.COM procedure in Example 5–2 executes a procedure containing the command REPLY as a batch job. Example 5-7 displays the contents of a sample procedure. The WAIT command gives the system manager time to log in at the console terminal before the other users are aware that the system is available. The system executes the REPLY command after 45 seconds.

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

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

5.6 Laboratory Exercises

1. Print the SYSTARTUP_V5.COM file and look at the output.
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 VT300-series terminals.
    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.
    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 to 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 directory. Do not execute this procedure to verify your work. Instead, look at the answers. The procedure should include commands to do the following:

   a. Mount the class disk (label = CLASS) on the device DRA1: Be sure it is 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. 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 2 and the priority at 3.
      - BIGJOB (batch execution queue)

        - Set the job limit at 1 and the priority at 2.

        - 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 after it has been printed.

j. Send a message to all terminals telling users that the system is up and ready.

5.7 Solutions to Laboratory Exercises

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

   Looking at the SYSTARTUP_V5.COM file familiarizes you with the contents of this file as a whole. You may want to look at a copy of SYS$SYSTEM:STARTUP.COM as well.

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 similar 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-
/VT300/SPEED=2400  TTA4:    !A.Steel   E19
$ SET TERM/PERM/NOMODEM-
/VT300/SPEED=9600  TTA5:    !J.Howland F01
$ SET TERM/PERM/NOMODEM-
/VT300/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 at 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. The following example is a sample SYSTARTUP_V5.COM. This procedure meets the specifications of the problem, but it is not the only answer. If your procedure contains similar 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$SYSTEM:INSTALL
CREATE  SYS$SYSTEM:BACKUP /OPEN/HEADER
EXIT
$!
$ 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

6.1 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

The VMS operating system 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 ways to start 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 the VMS system reads on startup.

This module assumes that you have already configured VMS system software to run on your system, as outlined in Module 7 and Module 5.

6.2 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

6.3 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 Operation Guide* for your specific VAX processor

6.4 Starting Up a VMS System

The system manager stores the image files, data files, and other files that the VMS operating system requires for its own operation on a disk volume known as the **system volume**. This volume is mounted on a disk drive known as the **system device**.

To start up a VMS system, system software must be moved from the system volume into memory to be executed. Also, the VMS database, which reflects the system hardware and software configuration, must be built. The system accomplishes this entire procedure, called **system initialization** or **system startup**, in several phases, summarized below.

Phase 1: The **primary bootstrap program**, VMB.EXE, is moved into memory and executed. At this phase, you can request a **conversational startup** where the system prompts you at Phase 3 for input about the system hardware and software configuration.

Phase 2: The primary bootstrap program moves the **secondary bootstrap program**, SYS$SYSTEM:SYSBOOT.EXE, into memory and executes it.

Phase 3: The secondary bootstrap program loads the **executive images**, including SYS$SYSTEM:SYS.EXE, into memory and executes them. If you requested a conversational startup at Phase 1, the system accepts input interactively.

Phase 4: The executive images initialize the VMS operating system database and prepare it to create processes. A message is printed on the console terminal stating the VMS version and the date. Then the **system initialization process**, which runs the image SYS$SYSTEM:SYSINIT.EXE, is created.

Phase 5: The system initialization process continues to initialize and build the VMS system software. It creates a detached **system startup process,** which executes the DCL commands in the system startup command procedures.

Phase 6: The system startup process completes the construction of the VMS system database and the configuration of the full VMS system. It also executes the site-independent command procedure, SYS$SYSTEM:STARTUP.COM. This procedure, provided with the system, contains commands common to all VMS systems. It executes several site-specific command procedures, including SYS$MANAGER:SYCONFIG.COM and SYS$MANAGER:SYSTARTUP_V5.COM. These procedures contain commands that the system manager enters. Module 5 discusses these procedures and their customization.

Phases 1, 2, and 3 are called **bootstrapping** the system. At the beginning of phase 1, the contents of VAX memory are unknown. By the end of Phase 3, enough software is running in memory for it to be called a VMS system.

Phases 2 through 6 are essentially the same for any VAX system processor. However, each of the major kinds of VAX system processors accomplishes Phase 1 differently, because each is hardware-dependent. This module covers the basic, original VAX system processors, specifically the VAX-11/780, VAX-11/750, and VAX-11/730. (The procedures described in this module are quite similar to those used with other VAX processors. Detailed information about startup procedures and options can be found in the *VMS Installation and Operation Guide* for your particular processor.) Figure 6–1 illustrates the sequence of startup phases and the image or command procedure executed at each phase.

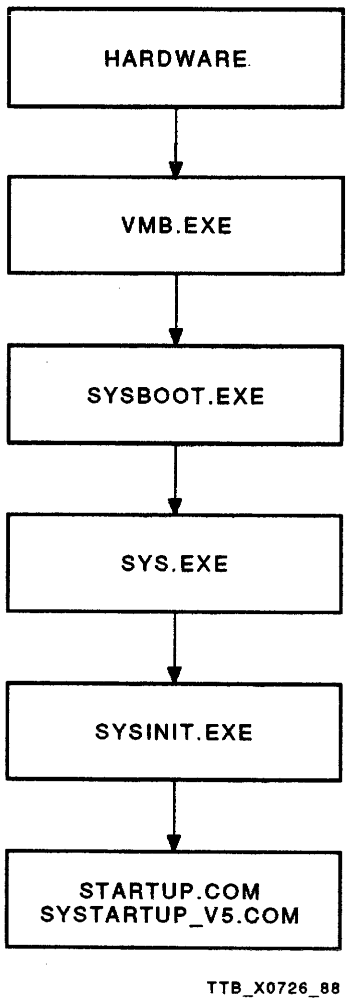


Figure 6–1 VMS System Startup Phases

6.4.1 The VAX Console Subsystem

The component of VAX system hardware that enables you to exercise control over the rest of the hardware is the **console subsystem**. Use it to accomplish Phase 1 of system startup. The console subsystem consists of the following parts, which are described in Table 6-1.

- Console processor
- Console command language (CCL)
- Console terminal
- Console device
- Front panel switches and lights
- Remote diagnosis port (optional)

Table 6–1 VAX Console Subsystem

Component	Function
Console processor	Interprets and executes console commands affecting VAX system hardware. It does not require the presence of VMS software.
Console command language (CCL)	Expresses console commands affecting VAX hardware.
Console terminal	Enters CCL commands and displays responses (console I/O mode). When the VMS system 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. Updates the VMS operating system, installs optional VAX software, and runs diagnostics.
Front panel switches and lights	Controls system startup, recovers from system failure, displays system status, and performs remote diagnosis. Keyswitch turns power on and off.
Remote diagnosis port	Enables a terminal at the Digital Diagnosis Center to function as a remote console terminal.

6.4.2 The Front Panel Switches

Switches and lights on the VAX system front panel control the processor and indicate its status. To start up a VMS system, you must set the front panel switches correctly.

Table 6–2 describes the switches for three different processors. Notice that even though the names of some switches vary between processors, their function is the same.

Table 6–3 describes the lights for these processors. Learn the light colors for your processor so you can see at a glance whether your system is running or not. Check the lights occasionally to catch problems that no one has reported.

Figure 6–2 shows the front panels of these processors. For more information see the *VMS Installation and Operation Guide* for your particular processor.

Table 6–2 VAX System Front Panel Switches

	Switch		
Function	**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 6–3 VAX System Front Panel Lights

	Panel Light		
Function When Lit	**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 up. (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 6–2 shows these block letter labels.

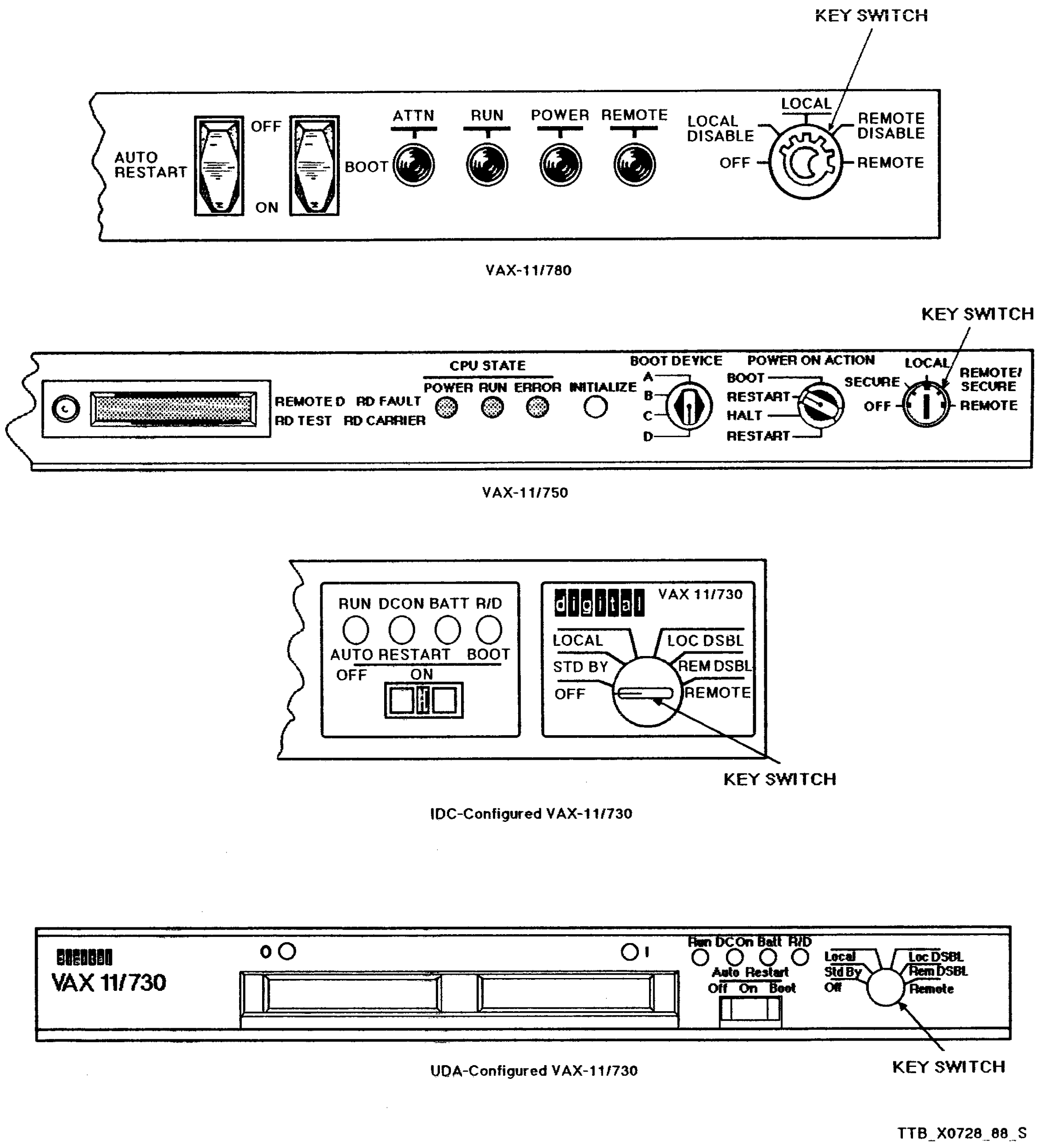


Figure 6–2 VAX Console Control Panels

6.4.3 The Console Processor and Console Device

The VAX system console processor accomplishes the first phase of system startup. VAX processors have a separate console processor, such as the VAX 8600, the VAX-11/780, and the VAX-11/730. For others, such as the VAX-11/750, the VAX CPU is used as a console processor as well as a system processor. However, the CPU does only one function at a time.

During the first phase, the VAX-11/780 and 11/730 require access to certain files on the **console volume**. This volume, which must be physically mounted in the console device for startup to succeed, may contain:

- Processor microcode
- Primary bootstrap program
- CCL command files

The VAX-11/780 console volume is a diskette. The diskette loads the RX01 disk drive inside the VAX processor cabinet. The drive swings out on a pivot for easy access. Because the console volume is necessary for startup and powerfail recovery, it should remain in the console device, unless a procedure (such as software installation) calls for removing it and loading another volume.

The VAX-11/730 console volumes are TU58 cartridge tapes. There are two console devices on the VAX-11/730, both TU58 cartridge drives. For one type of VAX-11/730, these cartridge drives are on the front panel of the processor. For the other type of VAX-11/730, one is inside the processor cabinet and the other is on the front panel. One TU58 drive always contains the console volume used to start the system. The second drive is available for installing updates and optional software.

For some other processors, such as the VAX-11/750, you can use the console volume (in a TU58 device) during the startup procedure, but it is not required. Generally, you set the BOOT DEVICE switch on the front panel to boot the system automatically using microcode stored in read-only memory (ROM).

Each VAX-11/750 has at least two ROMs. Set the BOOT DEVICE switch on the front panel to connect one at a time. This controls which device the system boots when you press the INITIALIZE button. If you set the switch to the "A" position, the system boots the console device. If you set the switch to another position, the system boots the system device associated with that position. Your Digital Field Service representative establishes which setting boots which devices by installing ROMs that match your system in the front panel. ROMs are normally provided for positions "A" (the console device) and D (the default system device).

Booting the console volume starts the BOOT58 program on the volume. When the BOOT58> prompt appears, you may enter BOOT58 commands. The BOOT58 command language is an extension of CCL. Booting with the console volume is slower than booting with a ROM because the console device is slow, but it is more versatile. For example, you can use BOOT58 commands to boot system devices for which you do not have ROMs if the console volume contains a command procedure for booting that device. Table 6-4 lists the names and locations of the VAX console devices.

For more information about the BOOT58 program and commands, consult the *VMS Installation and Operation Guide* for the VAX-11/750.

> NOTE
> The TU58 hardware records data on its tape cartridge in such a manner that VMS software treats the cartridge as if it were a disk. The storage capacity of the TU58 "disk" is approximately the same as the capacity of an RX01 diskette. Thus, there are few differences in the two console device types, other than that the TU58 is slower than the RX01.

Table 6–4 VAX Console Devices

Processor	Console Device	Location	Device Name
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. 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.	CSA2:

6.4.4 The Default System Device

Before you can start a VMS system, you must physically load the system volume into the disk drive that will be the system device. VMS systems support a variety of system devices. When you start the system, you must specify the system device, or use the default.

At any VMS system installation, the same device usually serves as the system device during each startup. For this reason, each VAX processor supports a method of specifying a **default** system device. These methods are covered later in this module. Once you specify the default system device, system startup is almost automatic.

There are many ways to start a VMS system whose default device has been specified. Table 6–5 describes one in detail for the three representative processors. (For other VAX processors, a similar, if not identical method is used.) There are three steps to start the processor:

1. Prepare the system for booting (by setting switches)
2. If the power is off, turn it on
3. Boot the system

You can include one final step to protect the console terminal and prepare the system for automatic restart after a power failure, but you do not need it to start the system. See Table 6-5, Step 4.

Examples 6–1, 6–2, and 6–3 show some console terminal listings that result from startups described in Table 6–5. Most VAX processors perform self-checking and memory tests before allowing user input. When the processor is ready, you will see the ">>>" prompt on the console terminal. In each sample listing, two asterisks mark this ready prompt (">>>"). See Table 6-5, Step 3.

> NOTE
> When you start your system, your listing may contain more informational messages than any of these sample listings. The number and content of the messages depend on the configuration of your system and the content of your site-specific startup files.

Table 6–5 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 usually D, but this depends on the 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	Toggle BOOT switch when the >>> prompt appears on console terminal.	Set POWER ON ACTION switch to BOOT when the >>> prompt appears on console terminal. Press INITIALIZE button.	Set POWER ON ACTION switch to BOOT when the >>> prompt appears on console terminal. Press INITIALIZE button when the BOOT58> prompt appears, type BOOT then RETURN.	Toggle AUTORESTART/ BOOT switch from OFF to BOOT when the >>> prompt appears on console terminal .
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.

```
        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=OE-10 FPLA-OE CON=V07-00-L
** >>>
        CPU HALTED
        INIT SEQ DONE
        HALT INST EXECUTED
        HALTED AT 200034F9

        G 0000000E 00000200
        LOAD DONE, 00005000 BYTES LOADED
VAX/VMS Version 4.6 23-JUN-1986 10:00

%%%%%%%%%%%  OPCOM 3-JUN-1987 08:33:03  %%%%%%%%%%%
Logfile has been initialized by operator _OPAO:
Logfile is SYS$SYSROOT:[SYSMGR]OPERATOR.LOG;9

%SET-I-INTSET, login interactive limit = 64, current interactive value = 0
  SYSTEM       job terminated at 3-JUN-1987 08:33:40.72

@
```

Example 6–1 Default Startup of a VAX-11/780 from Power Down

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

   VAX/VMS Version 4.6 23-JUN-1986 10:00

   %%%%%%%%%%%  OPCOM  11-JUL-1987 10:21:34.36  %%%%%%%%%%%
   Logfile has been initialized by operator _OPAO:
   Logfile is SYS$SYSROOT:[SYSMGR]OPERATOR.LOG;2

   %SET-I-INTSET, login interactive limit = 50, current interactive value = 0
    SYSTEM       job terminated at 11-JUL-1987 10:21:53.85

$
```

Example 6–2 Default Startup of a VAX-11/750 Using the Default System Disk from Power Down

```
        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 4.6 23-JUN-1986 10:00

PLEASE ENTER DATE AND TIME (DD-MMM-YYYY HH:MM) 07-JUL-1988 18:55
%%%%%%%%%%%  OPCOM  7-JUL-1987 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-1987 18:57:40.34
$
```

Example 6–3 Default Startup of a VAX-11/730 from Power Down

The startup method is only slightly different if the power is already supplied to the processor (keyswitch is not set to OFF). The VAX-11/780 and VAX-11/730 startup is somewhat faster in this case, because these processors already have certain files in memory. They do not have to read the files from the console volume. Table 6–6 summarizes this startup method. Note that the two startup methods are the same except for Step 3.

Table 6–6 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 usually D, but this depends on the 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. If nothing happens at console terminal, type CTRL/P. When >>> prompt appears, type HALT then RETURN. When >>> prompt appears again, type BOOT then 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 then RETURN.	Toggle AUTORESTART/ BOOT switch from OFF to BOOT. If nothing happens at console terminal, type CTRL/P. When >>> prompt appears, type B then 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.

6.4.5 The Automatic Restart

When a VAX processor is equipped with optional battery backup, the battery power can maintain the contents of memory during a temporary loss of power. When power is restored, the system automatically restarts itself. User processes, batch jobs, and print jobs will resume at the point where they were interrupted.

Each VAX processor supports a way to specify the system device for automatic restart.

You can turn automatic restart off and on by using the console switches. Table 6–7 shows how to set the console switches to choose the automatic restart response of a VAX processor.

Remember that Step 4 in Table 6-5 and Table 6-6 prepares the system for automatic restart.

Table 6–7 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	—

6.4.6 Alternate System Devices

If the console volume is not customized for default startup, or if you want to override the customized default, you can specify (at startup) the system device you want to use.

To specify the system device explicitly when you start the system, follow the steps in Table 6–8 for the processors listed. (The steps for other VAX processors are similar, if not identical.) In Step 3, you must supply a code that specifies the device type and unit number of the system device.

> NOTE
> These codes are not the standard VMS system device codes for interactive use. For example, although there are several standard VMS system device codes (DB, DR) for different MASSBUS disks, a single code (DB) applies to all MASSBUS disks at startup.

Table 6–9 lists some of the startup device codes for VMS system devices. The letter S in the table indicates the devices each VAX processor supports as a system device.

Table 6–8 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 then RETURN. Then, type BOOT ddu then 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 then 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 then 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 then 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 6–9 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
RA80			S	DQ
Console Device				
RX01	D			CS
TU58		D	D	CS[4]

[1] 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 **ddu**, 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.

Examples 6–4, 6–5, and 6–6 show the console terminal listings that result from three of the four startups described in Table 6–8. In these examples, the initial setting of the power switch is not OFF.

```
>>>BOOT DB0

        CPU HALTED
        INIT SEQ DONE
        HALT INST EXECUTED
        HALTED AT 20003552

        G  0000000E  00000200
        LOAD DONE,  00005000 BYTES LOADED

  VAX/VMS Version 4.6 23-JUN-1986 10:00

%%%%%%%%%%%  OPCOM  23-MAY-1987 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-MAY-1987 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-MAY-1987 08:39:48.60  %%%%%%%%%%%
Message from user DECNET
DECnet starting

   SYSTEM       job terminated at 23-MAY-1987 08:39:48.90
```

Example 6–4 Startup of a VAX-11/780 Specifying an RM05 as System Device from Power On

```
>>>B DBA0
%%

  VAX/VMS Version 4.6 23-JUN-1986 10:00

%%%%%%%%%%%  OPCOM  3-JUN-1987 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 3-JUN-1987 09:30:12.52
%%%%%%%%%%%%  OPCOM  3-JUN-1987 09:30:34.07  %%%%%%%%%%%%
Message from user DECNET on MOTHER
DECnet starting
```

Example 6–5 Startup of a VAX-11/750 Specifying an RM03 as System Device from Power On

```
>>B DM0
>>>@DD1:D,0B00.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 4.6 23-JUN-1986 10:00

%%%%%%%%%%%  OPCOM  3-JUN-1987  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 3-JUN-1987 09:21:24.59
```

Example 6–6 Startup of a VAX-11/730 Specifying an RL02 as System Device from Power On

6.4.7 The Standalone Utilities and Diagnostics

You can run certain hardware diagnostics and VMS system utilities on a VAX processor without running the VMS system. This is called running the diagnostic or utility **standalone**. For example, you can run the Backup utility either under the VMS system or standalone. You must run standalone BACKUP when the VMS system is being installed for the first time. Module 7 includes an example of starting standalone BACKUP.

Start standalone utilities and diagnostics, like the VMS system, by following the steps in Table 6–8. Specify the system device, and the processor starts using information from a volume loaded on that device. Volumes other than those that can be loaded in a system device can store standalone utilities and diagnostics. For example, the console device is not suitable as a system device. However, you can boot standalone BACKUP from a volume loaded in the console device. You can also start standalone utilities and diagnostics from devices supported only for data storage.

When you start a standalone utility or diagnostic by following the steps listed in Table 6–7, use the code for the device that Table 6–8 shows.

Table 6–10 VAX Processor and CCL Commands

Processor	CCL Command
VAX-11/780	`>>>BOOT CS1`
VAX-11/750	`>>>B DDA0`
VAX-11/750 (BOOT58)	`BOOT58>BOOT CSA1`
VAX-11/730	`>>>B CS1`

6.4.7.1 Additional References

- For more information on the VAX front panel lights and switches and the VAX console subsystems, consult the *VMS Installation and Operation Guide* for your particular processor.
- For more information on the controls and indicators on typical system disk drives, consult the *VMS Installation and Operation Guide* for your particular processor.
- For more information about the BOOT58 program and commands, consult the *VMS Installation and Operation Guide* for the VAX-11/750.
- For more information about device names to use during startup procedures, see the *VMS Installation and Operation Guide* for your particular processor.

6.5 Laboratory Exercises

1. Under the supervision of your course administrator, look at the front panels of as many VAX processors as are available to you. Become familiar with them.

2. Your course administrator will shut the system down for you and 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 Operation Guide* for this type of processor.

4. If your course administrator agrees, 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 course administrator will shut the system down for you. Start the system, specifying the system device explicitly.

6.6 Solutions to Laboratory Exercises

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.

6.7 Console Commands

The command language of the VAX console subsystem is the **console command language** (CCL). Each VAX processor has its own CCL, that reflects its hardware characteristics.

A discussion of CCL and its syntax is beyond the scope of this course. Table 6–11 lists the functions of some of the most useful CCL commands for the representative processors. Typically, you will use only the BOOT, HALT, and CONTINUE commands. See the *VMS Installation and Operation Guide* for your particular system.

6.7.1 Issuing Console Commands at System Startup

All the startup methods covered in this module allow you to enter CCL commands in Step 3. If the console terminal does not display the CCL prompt (>>>), you can obtain it by typing CTRL/P. After entering CCL commands, you can continue Step 3 of startup. The console terminal serves as a CCL command terminal while the system is starting up.

As already mentioned in this module, you can also enter commands to the VAX-11/750 at startup using the BOOT58 command language, an extension of CCL. Use the console volume for system startup. In this case, you can enter BOOT58 commands in Step 3 of startup. See the *VMS Installation and Operation Guide for VAX-11/750.*

Table 6–11 Typical Console Commands

Command Function	VAX-11/780	VAX-11/750	VAX-11/750 (BOOT58)	VAX-11/730
Starting a 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	N/A	I
Executing commands in a file	@file-spec	N/A	@file-spec	@file-spec
Reloading and restarting the console processor	REBOOT	N/A	N/A	N/A

6.7.2 Issuing Console Commands While the VMS System is Running

When VMS system startup is complete, you can use the console terminal as an interactive user terminal. Any user can log in at this terminal. For an interactive user, the main difference between this terminal and any other terminal is that the console terminal also displays messages from users and VMS to system operators. However, this feature does not restrict its use to system operators alone.

The SYSTEM process performs the last step in the VMS startup procedure by executing the SYS$SYSTEM:STARTUP.COM command procedure. The last command in this procedure is LOGOUT. When you see the logout termination message on the console terminal for the SYSTEM process, you know that the startup procedure is over (See Example 6-7). At this point, you may log in at the console terminal.

```
❶ %SET-I-INTSET, login interactive limit = 64, current interactive value = 0
    SYSTEM       job terminated at 18-MAY-1989 20:16:46.67

❷ Username:  SYSTEM
❸ Password:

               Welcome to VAX/VMS Version 5.2
  $
```

Example 6–7 Using the Console Terminal Interactively After Startup

Notes on Example 6–7:

❶ Terminal displays SYSTEM logout message

❷ User presses RETURN and receives the Username: prompt

❸ User logs in

When the console terminal functions as an interactive VMS system terminal, it is said to be in **program mode**.

You can also use the console terminal to issue CCL commands while the VMS system is running. At the console terminal, you can obtain the CCL prompt by typing CTRL/P, as Example 6–8 shows. When you use the console terminal as a CCL command terminal, it is said to be 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 Eightbit
  ^P
❶ >>>
❷ >>>SET TERMINAL PROGRAM
     Broadcast          No Readsync        No Form            Fulldup
     No Modem           No Local_echo      No Autobaud        No Hangup
     No Brdcstmbx       No DMA             No Altypeahd       Set_speed
     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        No DEC_CRT3
  $
  ^P
❸ >>>SET TERMINAL PROGRAM
  $
```

Example 6–8 Using the Console Terminal in Console Mode (VAX-11/780)

Notes on Example 6–8:

❶ The user is logged in at the console terminal and is entering DCL commands. At any time, the user can press CTRL/P to enter console mode.

❷ Each VAX processor has its own CCL command for returning from console mode to program mode, as Table 6–12 shows. If you are logged in at the console terminal, changing to console mode does not log you out. When you return to program mode, you are still logged in.

❸ The user has typed CTRL/P and the console terminal is now in console mode. The console command SET TERMINAL PROGRAM returns the console terminal to program mode (for VAX-11/780 processors). Other VAX processors typically use the CONTINUE command to enter program mode.

When the console terminal is in console mode, the VAX-11/750 and VAX-11/730 processors halt. The VAX-11/780 processor, however, continues to run. (To halt the VAX-11/780 processor, you must enter the HALT command at the CCL prompt.)

To prevent unauthorized users from issuing CCL commands at the console terminal, you can use the keyswitch to disable CTRL/P, as Table 6–13 shows. See Table 6-5, Table 6-6, Step 4.

Table 6–12 Issuing Console Commands While the VMS System 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 6–13 Disabling 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

6.7.3 Customizing Startup and Automatic Restart

On the VAX-11/750, the BOOT DEVICE switch selects the default system device at startup and automatic restart. If you follow the optional startup method by using the console volume, the file DEFBOO.CMD on the console volume specifies the default system device.

On the VAX-11/780 and VAX-11/730, the file DEFBOO.CMD on the console volume specifies the default system device. On the VAX-11/780, the file RESTAR.CMD on the console volume enables automatic restart after a power failure.

There are other files stored on the console volume besides DEFBOO.CMD. The contents of these files are similar to the contents of DEFBOO.CMD. You can use any of these other files as the default, or you can start the system by referring to one of them directly. Each file contains CCL or BOOT58 commands. You can invoke any file with the BOOT command while you are in console mode. Table 6–14 lists the names of some of these files. Notice that the name of a file corresponds to the system device name. For example, if you enter the command BOOT DB0, the system executes the DB0BOO.CMD file. Example 6–9 shows the contents of the DB0BOO.CMD file stored on a typical VAX-11/780 console volume.

Table 6–14 CCL Command Files Used at Startup

Device Code Used at System Startup (See Table 6–9)	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

```
$ @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 press 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 6–9 File DB0BOO.CMD from a VAX-11/780 Console Volume

NOTE
The contents of DB0BOO.CMD on the VAX-11/780 console volume are not the same as the contents of DB0BOO.CMD on the console volume of another processor. See *The VMS Installation and Operation Guide* for your particular processor.

To use a file as the default boot file, invoke the DCL command procedure, SYS$UPDATE:SETDEFBOO.COM. This procedure copies the contents of a specific file to the DEFBOO.CMD file on the console volume. (See Example 6-10).

You should change the contents of the default boot file whenever you change the system configuration to use a different system device. If you do not change the default procedure, you must always include the name of the system device in the boot command to avoid errors.

Unlike DEFBOO.CMD, you usually do not have to customize the RESTAR.CMD file. RESTAR.CMD restarts the system after a power failure or when other conditions occur. Example 6–11 shows the file RESTAR.CMD as supplied on the VAX-11/780 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:  DB0BOO.CMD

%EXCHANGE-S-MOUNTED, the RT-11 volume _CSA1:  has been mounted
%EXCHANGE-S-DELETEPREV, previous copy of _CSA1:DEFBOO.CMD deleted
%EXCHANGE-S-COPIED, _CSA1:DB0BOO.CMD copied to _CSA1:DEFBOO.CMD, 2 blocks

Default boot command file now replaced with DB0BOO.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 6–10 Customizing the 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 press 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
$
```

Example 6–11 File RESTAR.CMD from a VAX-11/780 Console Volume

6.7.3.1 Additional References

- For more information on the console command language for the processors discussed, read the command descriptions in the chapters on the Console Subsystem for your type of processor in the *VMS Installation and Operation Guide*.
- For more information on the BOOT58 program, read the section on Using BOOT58 on a VAX-11/750 in the *VMS Installation and Operation Guide* for the VAX-11/750.
- For more information on files of CCL commands on the VAX console volume, read the section on Selecting a Bootstrap Command Procedure in the *VMS Installation and Operation Guide* for your particular processor.

6.8 Laboratory Exercises

(Do each problem in this lab under the direct supervision of your course administrator.)

1. Your course administrator 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 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 (it should).

6.9 Solutions to Laboratory Exercises

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 course administrator.

6.10 System Parameters

The VMS operating system builds its data structures, configures its peripheral devices, and customizes its user interface each time the system starts up. **System parameters** stored in the default parameter file control much of this customization. The values of these parameters are established as part of system installation. You can modify them after the system is installed by using the SYSGEN utility or the AUTOGEN.COM command procedure. (Both are discussed further in Module 9.) You can also modify them during the startup procedure with the SYSBOOT utility.

6.10.1 The Conversational Startup

The system parameters stored in SYS$SYSTEM:VAXVMSSYS.PAR are called the **current** system parameters. The current values initialize the system at startup. Unsuitable values can prevent the VMS operating system from starting. Values set for normal startup can not meet your current needs. Therefore, the VMS operating system includes the option of examining and modifying the system parameters in Phase 3 of startup, before the processor uses them to customize the system. This option is called a **conversational startup**.

When you start up a VMS system conversationally, the secondary bootstrap program (SYS$SYSTEM:SYSBOOT.EXE) displays the SYSBOOT> prompt, allowing you to affect the system parameters by entering SYSBOOT commands. After entering these commands, you can complete Phase 3 of startup by issuing the SYSBOOT command CONTINUE.

SYSBOOT supports a subset of the SYSGEN command language. Table 6–15 lists the principal SYSBOOT commands and their functions. In particular, the SET command allows you to specify parameter values that override the current system parameters. See the *Guide to Setting Up a VMS System,* or the *VMS Installation and Operation Guide* for your particular processor.

Table 6–15 Using SYSBOOT During Conversational Startup

Function	Command Format

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 6-3 illustrates the effect of system startup on system parameters.

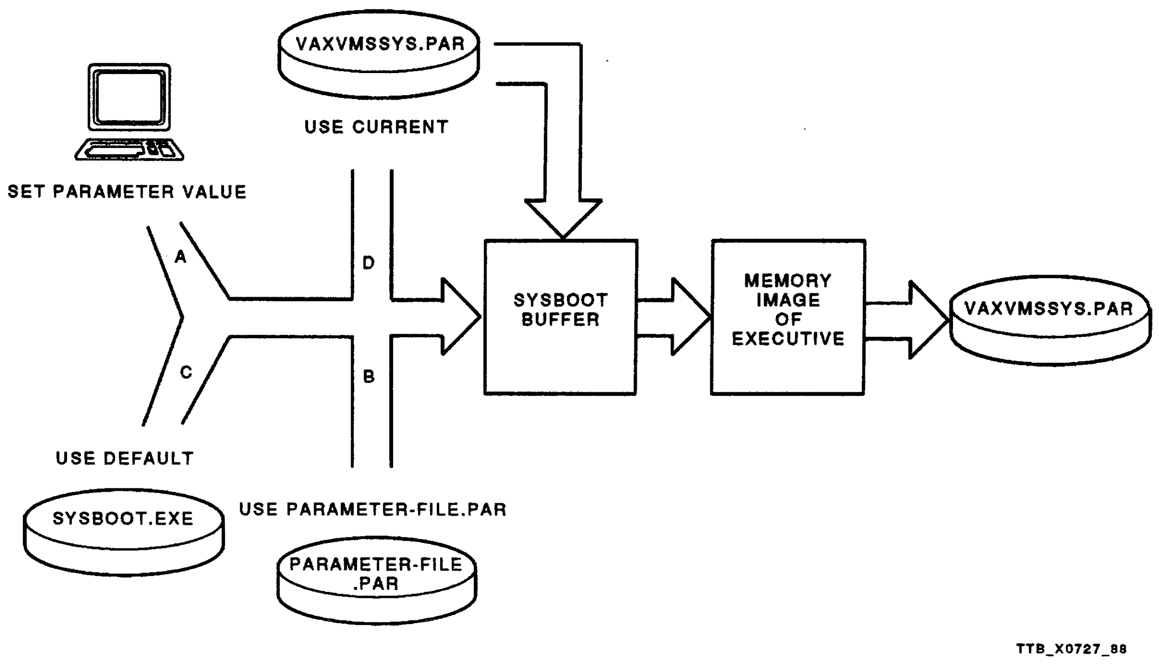


Figure 6–3 Effect of VMS System Startup on System Parameters

Notes on Figure 6-3:

At Phase 3 of startup, SYSBOOT copies the current system parameter values from SYS$SYSTEM:VAXVMSSYS.PAR to the SYSBOOT buffer.

If the startup is conversational, you can use the SYSBOOT utility to modify the parameters in the SYSBOOT buffer.

- You can modify the values of individual parameters. SET commands specify the parameters and their new values.
- You can replace all parameter values with values stored in a .PAR file in SYS$SYSTEM. You create such files, which contain values for every system parameter, by using the SYSGEN utility while the VMS system is running. Module 9 discusses the SYSGEN utility further.
- You can replace all parameter values with default values stored in the SYSBOOT image.
- You can replace all parameter values with the current system parameters stored in SYS$SYSTEM:VAXVMSSYS.PAR. (This is the default action SYSBOOT takes initially.)

After you issue the SYSBOOT command CONTINUE, the system copies the system parameters from the SYSBOOT buffer to the executive image in memory. This copy of the system parameters configures the system during the remaining phases of startup.

At Phase 5 of startup, the processor copies system parameters from the executive image in memory back to SYS$SYSTEM:VAXVMSSYS.PAR. Therefore, after startup completes, the parameter values customized by SYSBOOT become the current system parameters. Thus, they affect not only this startup but every subsequent startup, unless you modify them. You can modify these parameters in another conversational startup with SYSBOOT or with the SYSGEN utility or AUTOGEN.COM while the system is running.

Table 6–16 lists the steps you use to start a VMS system conversationally. The console volume supplies files required for conversational startup.

Table 6–16 Starting Up a VMS System Conversationally

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.	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	If >>> prompt is not displayed on console terminal, obtain it by typing CTRL/P. Type HALT then RETURN. Then, type @dduGEN then 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 then RETURN.	Set POWER ON ACTION switch to BOOT. Press INITIALIZE button. When the BOOT58> prompt appears on console terminal, type D/G 3 u then RETURN, @ddcGEN then 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 then RETURN, (where ddu is the startup code for the system device).
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.

```
>>>@DB0GEN
!
!       DB0 CONVERSATIONAL BOOT COMMAND FILE - DB0GEN.
!       BOOT FROM DB0 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>
❶ SYSBOOT>  USE ALTPARAM.PAR
❷ SYSBOOT>  SET UAFALTERNATE 1
❸ SYSBOOT>  SET/STARTUP SYS$SYSTEM:ALTSTART
❹ SYSBOOT>  CONTINUE

     VAX/VMS Version 5.2 3-JUN-1989 08:31
```

Example 6–12 (Cont'd on next page.) Typical Conversational Startup

```
%%%%%%%%%%%  OPCOM  26-JUN-1989 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-1989 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
The 13th VAX -- THE SUPERSTITION -- Ready for use.

%SET-I-INTSET, login interactive limit = 64, current interactive value = 0
   SYSTEM      job terminated at 26-JUN-1989 20:18:59.26
%%%%%%%%%%%  OPCOM  26-JUN-1989  20:19:05.73  %%%%%%%%%%%
Message from user DECNET
DECnet starting
```

Example 6–12 Typical Conversational Startup

Notes on Example 6–12:

❶ Specifies the alternate system parameter file, ALTPARAM.PAR. This file must already exist in the SYS$SYSTEM directory. Module 9 discusses how to create parameter files.

❷ Specifies the use of SYS$SYSTEM:SYSUAFALT.DAT, the alternate user authorization file, instead of SYS$SYSTEM:SYSUAF.DAT, the standard UAF. This alternate file must exist already. You can create it by copying and modifying the standard file.

❸ Specifies the use of SYS$SYSTEM:ALTSTART.COM, an alternate DCL startup file, instead of SYS$SYSTEM:STARTUP.COM, the standard, site-independent startup file. This alternate file must already exist before you can specify it in SYSBOOT.

❹ The CONTINUE command causes the startup to continue and finally complete.

Note that the system uses alternate parameter values and files defined in this example at all subsequent startups of this system, unless SYSGEN, AUTOGEN.COM, or SYSBOOT subsequently alters them.

6.10.1.1 Additional References

- For more information on the conversational bootstrap method of startup, consult the *Guide to Setting Up a VMS System*, or the *VMS Installation and Operation Guide* for your particular processor.
- To learn more about the commands and qualifiers that you can enter to the SYSBOOT utility, consult the *VMS System Generation Utility Manual*.

6.11 Laboratory Exercises

1. Your course administrator 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 course administrator. Check with your course administrator to verify this.
2. Your course administrator 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.

6.12 Solutions to Laboratory Exercises

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.

6.13 Specifying the System Configuration

As you have learned in this chapter, the VMS system configuration is customized each time you start the system. In particular, at startup:

- The system device is selected.
- System parameters determine how the VMS system database is constructed.
- DCL commands create the system environment.

By default, the processor takes most of the information required to start up the system from files on the system volume. The name of the default system device is included in a default command file on the console volume for the VAX-11/780 and VAX-11/730. For the VAX-11/750, define the default device using the BOOT DEVICE switch on the front panel or by specifying it in the default boot file on the console volume (not used unless you start the system using the BOOT58 program on the console volume). To customize the default system configuration, you must customize the files on the system volume and the default boot file on the console volume. Table 6–17 lists the principal files you must customize.

The following modules cover the ways to customize the files listed in Table 6–17:

- Module 5

  (SYSTARTUP_V5.COM, SYCONFIG.COM)
- Module 6

  (DEFBOO.CMD and VAXVMSSYS.PAR)
- Module 9

  (VAXVMSSYS.PAR)

Table 6–17 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, SYSGEN, or AUTOGEN
SYSTARTUP_ V5.COM SYCONFIG.COM	DCL commands	SYS$MANAGER	Text editor (such as EDT)

6.14 Multiprocessing Systems

Until now, this course has described procedures for managing VAX systems with a single VAX system processor. A system with only one processor is called a **uniprocessor** system.

Systems comprising two or more processors within a single cabinet are called **multiprocessing systems**. Older multiprocessing systems, such as the VAX-11/782, are **asymmetric multiprocessors**. In these systems, one processor acts as a **primary** processor and the other as an **attached** processor. (There are only two processors in older, asymmetrical multiprocessing systems. The primary processor in asymmetric multiprocessing systems is sometimes called the **master**, and the attached processor is called the **slave**.)

In asymmetric multiprocessing systems, the primary processor schedules jobs to run on either processor and performs all I/O tasks. If a job executing on the attached processor requests an I/O function, the attached processor suspends execution of the job and allows the primary processor to perform the I/O function.

Another kind of multiprocessing system is called **symmetric multiprocessing** (SMP). In SMP systems, there is one primary processor and one or more **secondary** processors. The term **symmetric** comes from the fact that each processor has equal access to the VMS system code and resources and can perform all I/O functions.

The primary processor is the processor that is logically or physically connected to the console device. The primary processor performs initialization activities at system startup and also serves as the system "timekeeper."

> NOTE
> VMS Version 5 (and later versions) does **not** support asymmetric multiprocessing systems, only symmetric multiprocessing systems.

6.14.1 Starting Up Multiprocessing Systems

When you install the VMS system on an SMP system, you customize the SYSGEN parameter file for multiprocessing startup. Certain system parameters for SMP systems declare which image is used to synchronize the system processors, and which processors are to be brought in to the multiprocessing environment at boot time (see Table 6–18).

To start up an SMP system, follow the same procedure that you use for a uniprocessor system. Refer to Tables 6–5, 6–6, 6–8, and 6–16.

Table 6–18 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.

6.14.2 Controlling Multiprocessing

Three DCL commands control the primary processor's use of the attached processor:

- START/CPU
- STOP/CPU
- SHOW/CPU

Table 6–19 explains the functions of these commands. Each command requires a CPU identifier, a unique number assigned to each processor. The primary processor is usually assigned the value 0. The secondary processors are assigned sequential numbers (starting with 1), depending on their positions on the VAXBI bus.

Table 6–19 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

When an SMP system is first powered up, processors that pass the initial hardware power-up diagnostics become part of the system's **available set**. Processors that are directed to participate in system operations become the **active set**. A processor can be in the available set, but not in the active set. The START/CPU command places a processor in the active set. The STOP/CPU command takes a processor out of the active set. The SHOW CPU command can be used to inquire about the status of a particular processor.

6.15 Shutdown

The introduction to this module lists typical reasons for shutting down a VMS system. This section discusses the following three methods for system shutdown:

- Orderly shutdown
- Emergency shutdown from DCL
- Emergency shutdown from CCL

6.15.1 Orderly Shutdown

When you use the orderly shutdown method, the VMS system attempts to avoid interrupting any critical system operations. It warns users of the approaching shutdown, and makes every attempt to preserve all system and user data. For this reason, you should always use the orderly shutdown procedure if circumstances permit.

You accomplish an orderly shutdown by executing the command procedure SYS$SYSTEM:SHUTDOWN.COM. You should log in as SYSTEM to execute this command procedure, but this is not required. However, you must have either SETPRV privilege or the privileges CMKRNL, SYSNAM, OPER, WORLD, SYSPRV, and EXQUOTA to execute this procedure. Although you can log in on any terminal, try to use the console terminal. (The console terminal is typically a hard-copy terminal. Use it for all operator functions including startup and shutdown procedures so you will have a paper copy of any problems that might occur.)

SYS$SYSTEM:SHUTDOWN.COM is provided with your VMS system software. An empty file, SYS$MANAGER:SYSHUTDWN.COM, is also provided. If you want to issue site-specific DCL commands during an orderly shutdown, add them to SYSHUTDWN.COM. Typically, you issue commands to turn off site-specific devices and remove images you installed in the SYSTARTUP_V5.COM file. You can also send extra messages to users, close files, or do other site-specific jobs.

During orderly shutdown, SYS$SYSTEM:SHUTDOWN.COM performs the following functions:

- Displays a notice on all terminals that a shutdown is planned, and reminds interactive users periodically as shutdown time approaches
- Prevents users from logging in (unless they have OPER privilege)
- Shuts down DECnet software
- 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 system paging mechanism to record on disk any file changes that have occurred in memory
- Writes a dump header and error log allocation buffers to the dump file
- For a multiprocessing system, shuts down the secondary processor(s)
- Dismounts the system disk

Example 6–13 shows how to perform an orderly shutdown on a VAX-11/780 system.

```
❶ Username:  SYSTEM
  Password:
          Welcome to VAX/VMS version V5.2 on node SUPER
      Last interactive login on Sunday, 23-APR-1989 08:20
      Last non-interactive login on Sunday, 23-APR-1989 01:24

              You have 7 new Mail messages.

❷ $ @SYS$SYSTEM:SHUTDOWN

          SHUTDOWN -- Perform an Orderly System Shutdown

❸ How many minutes until final shutdown [0]:  10
❹ Reason for shutdown [Standalone]:  PREVENTIVE MAINTENANCE
❺ Do you want to spin down the disk volumes [NO]?  YES
❻ Do you want to invoke the site-specific shutdown procedure [YES]? YES
❼ Should an automatic system reboot be performed [NO]?  NO
❽ When will the system be rebooted [later]:  AT 10:00
  Shutdown options:
❾  REBOOT_CHECK        Check existence of basic system files

  Shutdown options [NONE]:  NONE

  SHUTDOWN message from user SYSTEM at _OPA0:     08:23:36
❿ SUPER will shut down in 10 minutes; back up AT 10:00.
  PREVENTIVE MAINTENANCE

  44 terminals have been notified.
  %SHUTDOWN-I-OPERATOR, This terminal is now an operator's console.
  %%%%%%%%%%%  OPCOM  23-APR-1989 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.
⓫ %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
⓬ SUPER will shut down in 5 minutes; back up AT 10:00.  Please log out.
  PREVENTIVE MAINTENANCE

  3 terminals have been notified.

  SHUTDOWN message from user SYSTEM at _OPA0:   08:31:09
  SUPER will shut down in 3 minutes; back up AT 10:00.  Please log out.
  PREVENTIVE MAINTENANCE

  3 terminals have been notified.

  SHUTDOWN message from user SYSTEM at _OPA0:   08:32:15
  SUPER will shut down in 2 minutes; back up AT 10:00.  Please log out.
  PREVENTIVE MAINTENANCE
```

Example 6–13 (Cont'd on next page.)   Performing an Orderly System Shutdown on a VAX-11/780 System

```
3 terminals have been notified.
%SHUTDOWN-I-STOPQUEMAN, The queue manager will now be stopped.

SHUTDOWN message from user SYSTEM at _OPA0:   08:33:24
SUPER will shut down in 1 minutes; back up AT 10:00.  Please log out.
PREVENTIVE MAINTENANCE

3 terminals have been notified.

SHUTDOWN message from user SYSTEM at _OPA0:   08:34:30
SUPER will shut down in 0 minutes; back up AT 10:00.  Please log out.
PREVENTIVE MAINTENANCE

3 terminals have been notified.
13 %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-APR-1989  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-APR-1989 08:35:03.13  %%%%%%%%%%%
Message from user SYSTEM
_OPA0:, SUPER shutdown was requested by the operator.

%%%%%%%%%%  OPCOM  23-APR-1989 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

14        SYSTEM SHUTDOWN COMPLETE - USE CONSOLE TO HALT SYSTEM

>>>H

        HALTED AT 80008CE8
>>>
```

Example 6–13 Performing an Orderly System Shutdown on a VAX-11/780 System

Notes on Example 6–13:

❶ Log in as SYSTEM.

❷ Execute the command procedure SYS$SYSTEM:SHUTDOWN.COM.

❸ Your response to this question determines when the final phase of shutdown begins.

❹ Your response to this question becomes part of the shutdown notices displayed on user terminals.

❺ Your response to this question determines whether the system spins down disk volumes other than the system volume when they are dismounted. The VMS system will not spin down the system volume.

❻ Your response to this question affects whether the site-specific shutdown command procedure, SYS$MANAGER:SYSHUTDWN.COM, will be executed or not.

❼ If your response to this question is YES, the system starts up again automatically as soon as the shutdown is complete. The next section discusses automatic restart after a normal shutdown.

❽ Your response to this question appears in the shutdown notices displayed on user terminals. It has no effect on when the system actually starts up again.

❾ If you specify the REBOOT_CHECK option, the system will check the system disk to be sure the basic files used for rebooting are available and readable. This may take some time to do, so you may press the RETURN key to avoid using this option.

❿ The system sends a message to all terminals, specifying when the system will be shut down, why it will be shut down, and when it will be available again.

⓫ About five minutes before the shutdown completes, the system disables all interactive logins. Users with the OPER privilege can still log in. Before the terminal displays this message, you can completely cancel the shutdown by typing CTRL/Y. You can still cancel the remaining shutdown operations with CTRL/Y, but you must manually reverse any operation that the system has already performed.

⓬ The system sends out messages periodically to remind the users that a shutdown is in progress. It sends the first message when you invoke the procedure, then it divides the time left in half and sends the next message at the halfway point. It repeats this action until there are only three minutes left, when it sends a message every minute.

⓭ The system performs these operations during the final phase of shutdown.

14 This message indicates that shutdown is complete except for one final step. Go to the console terminal and type:

```
CTRL/P
>>>H
```

No matter what terminal SYS$SYSTEM:SHUTDOWN.COM is executed from, the message SYSTEM SHUTDOWN COMPLETE appears on the console terminal. You can only use the console terminal to issue the CCL HALT command. (HALT is not valid for some processors. See Table 6–12.)

Example 6–14 shows the shutdown notices that a user terminal displays during orderly shutdown.

```
Username:  JONES
Password:
            Welcome to VAX/VMS version 5.2 on node SUPER
          Last interactive login on Friday, 21-APR-1989 08:47
          Last non-interactive login on Sunday, 23-APR-1989 00:01
$
$ SHOW DEFAULT
  WORK1:[JONES]
$
❶ SHUTDOWN message from user SYSTEM at _OPA0:     08:23:36
SUPER will shut down in 10 minutes; back up AT 10:00.
PREVENTIVE MAINTENANCE      ❷                 ❸
           ❹
❺ $ SHOW DEFAULT
  WORK1:[JONES]

$
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.
PREVENTIVE MAINTENANCE

$
SHUTDOWN message from user SYSTEM at _OPA0:      08:31:09
SUPER will shut down in 3 minutes; back up AT 10:00.  Please log out.
PREVENTIVE MAINTENANCE

$
SHUTDOWN message from user SYSTEM at _OPA0:      08:32:15
SUPER will shut down in 2 minutes; back up AT 10:00.  Please log out.
PREVENTIVE MAINTENANCE

$
SHUTDOWN message from user SYSTEM at _OPA0:      08:33:24
SUPER will shut down in 1 minutes; back up AT 10:00.  Please log out.
PREVENTIVE MAINTENANCE

$
SHUTDOWN message from user SYSTEM at _OPA0:      08:34:30
❻ SUPER will shut down in 0 minutes; back up AT 10:00.  Please log out.
PREVENTIVE MAINTENANCE
❼ $ SHO
```

Example 6–14 User View of Orderly System Shutdown

Notes on Example 6–14:

❶ All terminals display one shutdown notice when you first execute SYS$SYSTEM:SHUTDOWN.COM. Periodic shutdown reminders are displayed only on logged in terminals.

❷ This is the number of minutes specified in Example 6–13, number 3.

❸ This is when you expect the system to be up again, specified in Example 6–13, number 8.

❹ This is the reason you are shutting the system down, as specified in Example 6–13, number 4.

❺ The user can continue working until the final phase of shutdown begins.

❻ After this message, the final phase of shutdown begins.

❼ In the beginning of the final phase, the system stops and deletes all user processes. This user attempted to enter another command after the final message, but was not allowed to complete it.

You can cancel the orderly shutdown at any time by pressing CTRL/Y at the terminal where you executed it. However, you must manually reverse any shutdown operations that the command procedure has already performed.

6.15.1.1 Automatic Reboot After Normal Shutdown

You can modify some system parameters while the system is running. The system will use the new values immediately. You can modify other parameters, but the system will not use the new values until you reboot the system. Therefore, a typical reason for shutting the system down in an orderly manner is to establish new values for the latter type of system parameter.

When you shut down the system in an orderly manner, you can request that the system reboot automatically. This automatic reboot will occur if you have done the following:

1. Set the AUTORESTART switch to ON. (On the VAX-11/750, set the POWER ON ACTION switch to RESTART/BOOT.)
2. Specified the system device in the default boot command procedure for the VAX-11/780 and the VAX-11/730. For the VAX-11/750, set the BOOT DEVICE switch to the system device setting (B, C, or D).
3. Answered YES to the automatic reboot question in the shutdown procedure.

NOTE
The automatic reboot after a shutdown does not restore the system to the state it was in prior to the shutdown (as it does in an automatic reboot after a power failure). Instead, the system is simply booted using the default system device. To create their interactive processes, users must log in as they would on any system that has just been booted.

6.15.2 The Emergency Shutdown

You can execute SYS$SYSTEM:SHUTDOWN.COM, specifying the number of minutes until shutdown as 0, to produce a fast orderly shutdown.

If some emergency makes an even faster shutdown necessary, or if the orderly shutdown procedure fails, you can run the image SYS$SYSTEM:OPCCRASH. Since this is a program, you can execute it from any terminal where you are logged in. This image, which requires CMKRNL privilege, performs only the following functions:

- Writes error log entries to disk
- Causes the VMS system paging mechanism to record on disk any file changes that have occurred in memory
- Writes a dump header and error log allocation buffers to the dump file
- Dismounts the system disk

OPCCRASH does not perform most of the cleanup performed during an orderly shutdown. For this reason, data may be lost if you use this method.

When OPCCRASH finishes running, it displays the following message on the console terminal:

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

When this message is displayed, use the console terminal to type:

```
CTRL/P
```

When the terminal displays the console prompt, enter the H (HALT) command as follows (not needed for processors that use the VAX CPU as a console processor):

```
>>>H
```

6.15.2.1 Automatic Reboot After Emergency Shutdown

You should only use the emergency shutdown in emergencies, such as a head crash or a memory failure. The system does not provide for an automatic reboot in this case.

6.15.3 Forcing a Shutdown with CCL Commands

If your system will not respond to terminal input, you will have to shut it down using CCL at the console terminal. The VAX-11/780 and VAX-11/730 console volumes include the command file CRASH, which contains CCL shutdown commands. On the VAX-11/750, you must enter the CCL commands manually. Table 6–20 shows how to force a shutdown using CCL commands on these processors.

No cleanup is performed during a CCL shutdown. Use this method only if DCL shutdown methods fail.

A CCL shutdown writes a copy of physical memory to the crash dump file and displays the contents of certain processor registers on the console terminal. This file and listing are important in isolating and solving system problems, as explained in Module 8.

Table 6–20 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`	

6.15.3.1 Automatic Reboot After Forcing a Shutdown

Generally, you force a system to shut down from the console terminal (using CCL commands) when it no longer responds to user input from other terminals. Since there are many reasons why a system will not respond, you can force a system failure and request an automatic reboot. Again, the system does not return to the state it was in at the time of the forced system failure. Instead, it just reboots for you automatically.

To request an automatic reboot after a forced system failure:

1. Set the AUTORESTART switch to ON. (For the VAX-11/750, set the POWER ON ACTION switch to RESTART/BOOT.)
2. For the VAX-11/780 and VAX-11/730, specify the system device in the default boot command procedure. For the VAX-11/750, set the BOOT DEVICE switch to the system device setting (B, C, or D).

For more information on shutting down a VMS system, consult the *Guide to Setting Up a VMS System*.

6.16 Laboratory Exercises

You must perform the following exercises on a standalone system, under the supervision of your course administrator, and then 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. Use the CCL method to force a shutdown.

6.17 Solutions to Laboratory Exercises

1. See Example 6–13 and its keyed comments.
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 Operation Guide* for the system you are using.

6.18 Optional Laboratory Exercise

Complete the following exercise only under the supervision of your course administrator. The purpose of this optional exercise is to show you that different errors can occur at different points during the startup process. 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 be able to get to the part of the listing where the system volume is read. If you cannot get to the part where the console volume is read, the problem might be in memory or in some area of the CPU.

If you are having problems starting up your system, first check to be sure you are preparing it 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.

Use the following steps to force some common startup problems:

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.)

   CAUTION
   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?

6.19 Solutions to Optional Laboratory Exercise

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 according to processors, but its meaning is the same: The system cannot boot without the console volume.
5. No solution needed.
6. No solution needed.
7. You should receive an error message at the console terminal that is different than the one you received when no volume was loaded. This second message means 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. This third error message means 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. This message means 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

7.1 Introduction

The manager of a VMS operating system must install and customize the system software 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 operating 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 your system, customize it using the skills discussed in the other modules in this course.

Finally, run the User Environment Test Package (UETP) and make backup copies of your system disk and console volume.

7.2 Objectives

To install and test a system, a system manager should be able to:

- 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

7.3 Resources

1. *VMS Installation and Operation Guide* for each processor
2. *VMS Release Notes* for the latest version VMS
3. *VMS License Management Utility Manual*
4. *VMS Backup Utility Manual*

7.4 Managing Product Licenses Using the License Management Facility (LMF)

7.4.1 Overview of the LMF

Most Digital products now require installation of a **license key** in order to operate on VMS V5.0 and later systems. 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, 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 specific use of the product is licensed.

There are two basic licenses, with a third license possible when the two are combined.

- The **availability license** allows the use of a product on a host node.
- The **activity license** allows a specific number of concurrent users to access a product.
- The **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 several products.

7.4.2 LMF Features and Benefits

Features and benefits that support the goals of the LMF:

- Centralized license management, administration, and tracking
- Management of products by the manipulation of license keys
- INCLUDE or EXCLUDE nodes for subcluster licensing
- Concurrent user licensing distributed over the cluster
- Focuses on usage of product, rather than on the number of systems
- Spans the VAX system family
- Allows combination of license keys
- Designed to support new technologies (such as distribution of software and documentation on CD-ROM)

The command procedure SYS$UPDATE:VMSLICENSE.COM assists you in registering your keys in the LICENSE database.

The license management facility provides a convenient and extensible means for ensuring that unlicensed Digital software, or other vendor's software, will not run on the VMS system.

Licensing information distinguishes between kinds of software (for example, system or application) and the amount of usage units permitted (based, essentially, on processor power). Usage may be permitted on a particular host machine, or restricted to a certain number of usage units. These units are like a pooled resource on the system, taken and returned dynamically as users invoke and exit the software.

Although it is convenient to say that an activity license, with a centain number of units, allows a specific number of concurrent users to use a product, it is more correct to say that, based on the number of users and their processors, a specific level of product usage is permitted.

The LICENSE database is used only when the product (layered or otherwise) makes an inquiry. Most people have the impression that the LMF can automatically stop software from running. This is not true. The software asks for information from the database. Then the software runs or does not run on behalf of a particular user. If the product (whether produced by Digital or another vendor) does not query the database, LMF does not prevent the product from running.

7.4.3 Components of the License Management Facility (LMF)

There are four basic components of the LMF:

1. The License Unit Requirement Tables (LURTs):
   - 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, the activity field, or both.
2. 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.
3. The LICENSE database holds all information about keys. The database is managed by the License utility:
   - SYS$COMMON:[SYSEXE]LMF$LICENSE.LDB is the default location of the database.
   - If there is more than one system disk in the cluster, the logical name LMF$LICENSE should point to a single database for the cluster containing all product licenses for all cluster hosts.
   - If there is no disk accessible to all hosts, separate databases should be maintained **identically** (see the *VMS License Management Utility Manual.*)
4. SYS$UPDATE:VMSLICENSE.COM:
   - A command procedure to assist in registering keys in the LICENSE database.

Table 7–1 Six License Unit Requirement Tables (LURTs)

Code	LURT Type
A	VMS Capacity
B	VMS Server
C	VMS Concurrent User (for **activity** licenses)
D	VMS Workstation
E	System Integrated Products
F	Layered Products

7.4.3.1 License Units and License Unit Requirement Tables

It is helpful to know how to move software between processors during upgrades or make changes in VAXcluster configurations.

Features of the **license unit** that provide ways to move software:

- 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 available units is greater or equal to the units required, then the product may be used.

7.4.3.2 License Unit Requirement Tables (LURTs)

- A LURT is a list of unit requirements for each supported processor for specific 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 SORT on a VAX 6230 system might require 1500 license units, while unlimited use of SORT on a MicroVAX II system might require only 200 units.
- A LURT for activity licenses specifies the number of units required for each concurrent user of the product on a specific VAX system model. Activity licenses are useful when the product is used infrequently or by a small number of users on a large machine or cluster. Additional activity licenses can be purchased, if the number of users increases.
  - 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 are 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 return to the general pool of available units.
- Products licensed by activity licenses can also be licensed by availability licenses if the use of the product increases beyond the point where an activity license is cost effective.

Table 7–2 Values for an Activity License LURT

VAX Model	Number of License Units Required per User
VAX 8650	75
VAX 8350	60
VAXstation 2000	30

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. Assume that the activity license LURT has the values shown in Table 7–2 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.

Eight users on the VAX 8650 require 600 units (8x75). If, at the same time, there are five users on the VAX 8350, that would require an additional 300 units (5x60). This leaves only 100 available units. Each of the three VAXstation 2000 systems can now have one QUERY user, requiring 90 units (3x1x30). The next user attempting to access QUERY on the cluster would receive an error message, because the 10 units remaining are not enough for any users.

The three VAXstation 2000 users cannot access QUERY from either the VAX 8650 or the VAX 8350. With 100 units remaining, only one additional user can use the product from either of the larger VAX system processors.

7.4.3.3 The License Management Utility (LICENSE)

LICENSE is a DCL-level interface to the License Management Facility (LMF) on the VMS operating system. The following format is used to invoke the LMF:

```
$ LICENSE  subcommand  parameter
```

To use the License Management utility (LICENSE), enter the LICENSE command and the desired LICENSE subcommand and qualifiers at the DCL prompt ($). For example, you can enter short commands on one line:

```
$  LICENSE LIST VAX-VMS/AUTHORIZATION=USA12345
```

Or, you can enter long command strings followed by a a hyphen (-) before pressing each RETURN:

```
$  LICENSE REGISTER FORTRAN  /ISSUER=DEC  /AUTHORIZATION=USA-1957 -
_$ /PRODUCER=DEC /UNITS=1200 /VERSION=4.6 -
_$ /AVAILABILITY=F /CHECKSUM=1-GEAD-MIDJ-IDNC-DLAC
```

Because LMF may return only a checksum error when you omit or incorrectly enter information, carefully check the characters you entered before pressing RETURN. The command procedure SYS$UPDATE:VMSLICENSE.COM eliminates much of the typing needed for the LICENSE REGISTER and LICENSE AMEND commands.

The LICENSE commands (except LICENSE START, LICENSE LOAD, and LICENSE UNLOAD) manipulate only the LMF database and require only the privileges required to access the LICENSE database. Since the LICENSE database is file-based, normal VMS file protection applies. The LMF provides the database with a default file access of read and write privileges to system-level processes (S:RW).

To enter the LICENSE START, LICENSE LOAD, and LICENSE UNLOAD commands, you need the privileges CMKRNL, SYSNAM, and SYSPRV.

To display a listing of licenses registered in the LICENSE database, use the LICENSE LIST command. To display the licenses active on the current node, use the DCL command SHOW LICENSE.

In the future, some products may automatically register a license at installation. Although in this case you do not need to use the LICENSE REGISTER command, you can use other LICENSE commands with these automatically registered licenses. The automatically registered licenses appear in the LICENSE database along with other licenses.

Table 7-3 lists the license subcommands used to invoke the License Management utility.

7.4.3.4 LICENSE Subcommand Overview

Table 7–3 LICENSE Subcommands

Subcommand†	**Function**
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.
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.
REGISTER	Adds a new license to the LICENSE database.
START	Sets up a 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.

†To use LICENSE subcommands, enter the LICENSE command followed by the appropriate subcommands and qualifiers.

AMEND

AMEND changes a license currently in the LICENSE database.

Use LICENSE AMEND only when the software vendor provides amendment information. (Currently, Digital does not issue amendments to licenses.)

Format:

```
LICENSE AMEND product-name
```

Example:

```
$ LICENSE AMEND XAMPLE /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 XAMPLE. Entering this command upgrades the existing XAMPLE license to Version 9.3. The producer name, issuer name, authorization number, and checksum number are entered exactly as they appear in the amendment information.

CANCEL

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 this license. Note that the product name is entered with a hyphen (-) character as it was specified on the PAK.

CREATE

CREATE creates a LICENSE database with no license records. LMF creates a default LICENSE database in SYS$COMMON:[SYSEXE]LMF$LICENSE.LDB. Thus you do not typically need to specify this command.

Format:

```
LICENSE CREATE
```

Example:

```
    $  LICENSE CREATE/DATABASE=SYS$MANAGER:LMF$LICENSE.LDB
```

DISABLE

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. Because no database is specified, LMF uses the default database.

ENABLE

ENABLE enables an existing license in the LICENSE database. The enabled license can then 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

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 LICENSE ENABLE.

Format:

```
LICENSE ISSUE product-name
```

Example:

```
$  LICENSE ISSUE /OUTPUT=SYS$MANAGER:FORTRAN.PAK -
_$    /PRODUCER=DEC FORTRAN
```

LIST

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 CTRL/Z to exit, use arrow keys to scroll.
.
.
.
-----------------------------------

 FORTRAN                  DEC
 COBOL                    DEC
 PASCAL                   DEC
[End of List]
```

Example 7-1 illustrates the /FULL and /HISTORY qualifiers to display additional information for the license database.

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.

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
```

Example 7-1 shows a license history that has been registered and subsequently modified.

```
    $  LICENSE LIST /FULL /HISTORY FORTRAN
Press CTRL/Z to exit, PF3-PF4 for Previous-Next Screen and 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
   Version:                     V4.7
   Date:                        (none)
   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]
```

Example 7–1 $ LICENSE/LIST/FULL/HISTORY Output

REGISTER

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. See Example 7-2 for an illustration of a Product Authorization Key.

Often the command procedure SYS$UPDATE:VMSLICENSE.COM is used to register a new product license. This provides a prompt-based interface to LICENSE REGISTER.

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

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 need this only if startup fails.

To use START 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 LURTs for your system, and activates all the licenses that are registered and enabled in the LICENSE database.

UNLOAD

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 affect 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
```

```
 - - - - - - -
| | | | | | | |            LICENSE SOFTWARE PRODUCT        ____________________
|d|i|g|i|t|a|l|            PRODUCT AUTHORIZATION KEY     | DOCUMENT ISSUE DATE |
| | | | | | | |                                          |     18-MAY-1989     |
 - - - - - - -                                           |_____________________|
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: 18-APR-1989

 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-CCLB-MNBO-KNNG-CBEH
********************************************************************************
```

Example 7–2 Product Authorization Key

Examples 7-3 and 7-4 illustrate an actual installation of the VMS License Management utility, using the PAK information from Example 7-2.

```
$ 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. Cancel an existing Product Authorization Key
        4. List Product Authorization Keys
        5. Modify an existing Product Authorization Key

        9. Exit this procedure

    Type '?' at any prompt for a description of the information
    requested.

Enter one of the above choices [1]: 1
Do you have your Product Authorization Key? [YES]: y

   The REGISTER option allows you add a new license to a license
   database.  A Product Authorization Key (PAK) provides the product
   name and information you need to register the license.  You must
   enter all the information provided by your PAK exactly as specified.

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 []: 18-APR-1989

KEY TERMINATION DATE:
          Key Termination Date []:

RATING:
       Availability Table Code []: F
           Activity Table Code []:
```

Example 7–3 VMSLICENSE Session - Part I

```
MISCELLANEOUS:
                  Key Options []: MOD_UNITS
                Product Token []:
                  Hardware-Id []:
                     Checksum []: 1-CCLB-MNBO-KNNG-CBEH

      License Database File:  SYS$COMMON:[SYSEXE]LMF$LICENSE.LDB
                     Issuer:  DEC
              Authorization:  USA000877
                   Producer:  DEC
               Product Name:  DOCUMENT
                      Units:  5000
                       Date:  18-APR-1989
                    Version:  5.0
           Termination Date:
               Availability:  F
                   Activity:
                    Options:  MOD_UNITS
                      Token:
                Hardware ID:
                   Checksum:  1-CCLB-MNBO-KNNG-CBEH

Is this information correct? [YES]:
Registering DOCUMENT license in SYS$COMMON:[SYSEXE]LMF$LICENSE.LDB...

Do you want to LOAD this license on this system? [YES]:
%LICENSE-I-LOADED, DEC DOCUMENT was successfully loaded with 0 units

    VMS License Management Utility Options:

        1. Register a Product Authorization Key
        .
        .  <other options omitted from this listing>
        .
        9. Exit this procedure

Enter one of the above choices [1]: 9
$
```

Example 7–4 VMSLICENSE Session - Part II

7.4.4 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
```

7.5 Software Installation

The discussion of system startup in Module 6 assumes that the system you are starting already has a VMS system volume customized for your site. The process of creating a VMS Version 5.2 system volume and starting up VMS with it for the first time is called installing the VMS Version 5.2 (V5.2) system.

If your system is already running an updated V5.0 system, then you have probably already customized your system disk. As you create your V5.2 system disk from your current system disk, you can preserve many customized files, such as the user authorization file and the site-specific startup file. The process of creating a VMS V5.2 system volume and starting up VMS V5.2 for the first time, while preserving as many customized files on the current system volume as possible, is called upgrading the VMS operating system volume.

Maintenance updates are distributed periodically to correct VMS system problems discovered in the current release. Applying these maintenance updates to the system volume is called updating the VMS system volume. Note that you must always apply maintenance updates in sequential order.

After the system installation, upgrade, or update is complete, you should run the User Environment Test Package (UETP) to confirm that the system volume was successfully built.

If you want to run optional Digital software on your system (such as compilers and database managers) you must transfer this software from distribution kits to your system volume. This process is called installing VMS system optional software.

The VMS installation and upgrade procedures that create or modify a VMS Version 5.0 system volume are highly automated. They require very little user intervention. These procedures are included in the distribution kit you receive to install, upgrade, update, or add optional software to the system.

Each distribution kit contains documentation to show you how to install the software. A kit may also contain the distribution software on a distribution medium (tape or disk volume). See the *VMS Installation and Operation Guide* for your processor.

7.5.1 Installing a Major Release of the VMS Operating System

There is one guide 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*

7.5.2 Installing a Major Release

The distribution kit for a major release contains detailed installation instructions tailored to your system. They discuss your processor, your system disk device, and the device where you will load the distribution media by name.

Each set of instructions covers the same basic steps:

1. Build a VMS system volume on a scratch disk:

   a. Load the scratch system volume in a disk device for data volumes and run the Bad utility on the scratch volume if you have a VMS system running a previous version of VMS. Module 4 discusses the Bad utility in more detail.

   b. Load the scratch system volume in the system device. See the *VMS Installation and Operation Guide* for your processor.

   c. Use standalone BACKUP to copy a portion of the VMS system from the distribution medium to form a **kernel** VMS system. (A later section of this module discusses standalone BACKUP in more detail.)

   d. Start the kernel VMS system. (Module 6 discusses how to start a VMS operating system.)

   e. Watch the procedure use the Backup utility to copy the remainder of VMS from the distribution medium to the scratch volume. The result is a complete VMS system on the scratch volume.

   f. Boot the system using the new system disk. You may want to load it in the normal system drive first, or you can specify the name of it in the boot command. (See Module 6 for more information on how to boot using a specific device.)

2. Customize the new system volume and the console volume for your site:
   a. Use the Authorize utility to add UAF records, proxy records, and rights database information to the user authorization files. (Module 2 details how to do this.)
   b. Use the SYSMAN utility to set up disk quotas on data volumes as desired. (Module 2 details how to do this.)
   c. Modify the site-specific startup command procedure file so the appropriate queues will be created, volumes will be mounted, messages will be sent, and other startup chores will be done. (Module 5 discusses this in more detail.)
   d. Change the passwords of the default accounts, especially the SYSTEM account. (Module 8 discusses how to change passwords.)
   e. Use the SYS$UPDATE:SETDEFBOO.COM command procedure to change the contents of the default boot file on the console volume.
   f. Do any other jobs that make your system volume useful to you and to the users of your system.
3. Run the User Environment Test Package (UETP). You can run UETP before or after you customize the system.
4. Use the SYS$UPDATE:CONSCOPY.COM command procedure to copy the files from the console volume to a directory on the system volume. Usually you create a spare console volume with CONSCOPY.COM at this point as well.
5. Use standalone BACKUP to copy the system volume to another disk, to a multivolume disk set, or to a multivolume tape set so you will have a backup copy in case there is a problem with the system volume.

The *VMS Installation and Operation Guide* has informative sections on:

- Contents of the distribution kits for Version 5.0.
- How to use the disk devices on your system
- Installing a base VMS system

7.6 Optional Laboratory Exercise

Obtain a copy of the set of instructions for installing Version 5.0 (or later) on your system. Under the supervision of your course administrator, follow these instructions.

7.7 Solution to Optional Laboratory Exercise

If you followed the instructions carefully, you should be able to boot your system and log in. If you cannot do this, consult your course administrator.

7.7.1 Upgrading the VMS Operating System to Version 5.0

If your system is running Version 4.6 or higher, you can upgrade it to Version 5.0 by using the software in your distribution kit and the command procedure SYS$UPDATE:VMSINSTAL.COM. Before you begin the upgrade, back up your system disk using the standalone Backup utility. You should also read the *VMS Release Notes* and Upgrading to Version 5.0 in the *VMS Installation and Operation Guide* for your processor.

This section discusses how to prepare for and perform the upgrade by following a set of steps. There are several phases of the upgrade procedure, the first of which is slightly different for each processor. Therefore, be careful to follow the steps listed for your processor.

> NOTE
> The VMS upgrade procedure is restartable. The system is started up several times as part of the upgrade. If a power failure or other system interruption occurs, the *VMS Installation and Operation Guide* for each processor explains how to resume the upgrade from the last startup you performed.

After you have completed the upgrade procedure, run the User Environment Test Package (UETP) to test the installation. Then make a backup copy of the new system volume. If you modified the console volume during the upgrade procedure, use the SYS$UPDATE:CONSCOPY.COM command procedure to make new copies of the console volume.

Read the section on upgrading the VMS operating system in the *VMS Installation and Operation Guide* for your processor.

7.8 Optional Laboratory Exercise

Under the supervision of your course administrator, follow the instructions in the *VMS Installation and Operation Guide* to upgrade your system to a Version 5.0 (or later) system.

7.9 Solution to Optional Laboratory Exercise

After you have completed the procedure of upgrading your system to Version 5.0 (or later), you should be able to boot the system and log in. If not, consult your course administrator.

7.9.1 Installing Maintenance Updates

Maintenance updates are distributed periodically to correct problems discovered in a major release. You must use the SYS$UPDATE:VMSINSTAL.COM command procedure to copy the update software from the distribution medium to your system disk. *VMS Release Notes* accompany each update and discuss the VMSINSTAL procedure and the changes that were made to the software to create an update. You should always read the release notes carefully, as commands or methods of doing certain tasks may change.

NOTE
You must install maintenance updates in the proper sequence. For example, it is normally not possible to update VMS directly from V5.0 to V5.2. First, you must update V5.0 to 5.1; then you can update 5.1 to 5.2.

Be sure to follow all of the listed instructions in the installation guide for the update. When you finish the installation, run the User Environment Test Package (UETP) to check the integrity of the system.

NOTE
When you build a new system volume, begin by installing the latest major release. Each major release contains all of the VMS software for that version. Then update the volume with each maintenance update in **sequential order**.

7.10 Optional Laboratory Exercise

Under the supervision of your course administrator, 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. You **must** apply them in sequential order.

7.11 Solution to Optional Laboratory Exercise

If you have followed the instructions carefully, you can boot, log in to, and use your updated system. If you cannot do this, consult your course administrator.

7.11.1 Running UETP

Run the User Environment Test Package (UETP) after installing the system to demonstrate that VMS hardware and software are working correctly. The UETP tests system components including:

- All standard peripheral devices
- Various commands and operating system functions
- The system's multiuser capability
- DECnet-VAX interface

The command procedure SYS$TEST:UETP.COM controls the action of the UETP. Log in using the UAF record SYSTEST (supplied with the system) because UETP will run properly only in a process created with the values in the SYSTEST record.

Steps to follow when running UETP:

1. Log in, using the user name SYSTEST.
2. Prepare all devices for testing, as desribed in the *VMS Installation and Operation Guide* for your processor.
3. Execute the UETP with the DCL command @UETP.
4. Each time the UETP prompts you to input information about running the test, press the RETURN key. This tells the UETP to use the default response for the question. (The UETP prompts for four responses, so you will press the RETURN key a total of four times.)
5. If the UETP completes successfully, run the Errorlog Report Formatter (ERF) using the ANALYZE/ERROR_LOG command to check for hardware problems that may have occurred during the UETP run. Module 8 discusses this utility.
6. If the UETP does not complete successfully, follow the troubleshooting instructions described in the section on Running the UETP After Installation in your *VMS Installation and Operation Guide.*

The UETP consists of a number of phases. Each phase consists of a number of tests. See the *VMS Installation and Operation Guide* for more information on the organization of the UETP.

> NOTE
> The UETP does not test any layered products (optional software).

7.12 Optional Laboratory Exercise

After you have installed, upgraded, or updated your system, run the User Environment Test Package by following the instructions in the section on Running the UETP After Installation, in the *VMS Installation and Operation Guide*.

7.13 Solution to Optional Laboratory Exercise

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 course administrator.

7.13.1 Installing Optional (Layered) Products

A variety of optional software products are available to run under the control of the VMS system. Many of these are listed in the Programs and Utilities section of Module 1. Another term used to describe these is **layered products**.

Install optional products after installing the VMS system and running UETP. Use SYS$UPDATE:VMSLICENSE.COM to register the 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 system kit and requires no further installation. These **system integrated products** include:

- DECnet-VAX software
- VAXcluster software
- VAX Volume Shadowing
- RMS Journaling

For most products, you use SYS$UPDATE:VMSINSTAL to copy the product software to the system disk.

Each of these layered products is distributed on one or more volumes of the processor's console medium (RX50 floppy or TK50 cartridge) or on one or more volumes of magnetic tape. The VMSINSTAL command procedure, SYS$UPDATE:VMSINSTAL.COM, copies software files from these distribution volumes to the system disk.

VMSINSTAL is a command procedure that loads and executes the optional product's installation procedure. Therefore, for different products, you will not see the same questions or output as VMSINSTAL runs, nor will you enter the same responses for each product. You should always refer to the installation guide for the product so you can enter proper responses. Also, always execute VMSINSTAL on a hard-copy terminal (usually the console terminal) so you can refer to the output in the future or use it to solve problems during the installation.

The layered product software distribution kit may also include other mass storage media such as an RK07 disk volume or RL02 disk volume. VMSINSTAL also controls the installation of files from this volume.

Each optional product requires space on a disk volume. You must determine where the space is available (using the SHOW DEVICE/FULL command), and indicate where the procedure (VMSINSTAL) should store the product's files. If there is no room available on the volumes currently used, do not install the product. (As a general rule, try to leave at least one quarter of the system disk free for the best performance.)

To completely install a layered product, you often need to do more than restore software. You may need to modify limits in UAF records, create new directories and logical names, change SYSGEN parameters, and do other tasks. To avoid problems later, complete all of the instructions listed in the optional product's installation guide before you allow users to work with the product.

One optional product is the FORTRAN compiler. In Example 7-5, the VMSINSTAL procedure installs this product.

```
$ @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 7–5 Using the VMSINSTAL Command Procedure to Install VAX FORTRAN

```
* 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 VAX 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.

* Do you want to install the larger version of VAX 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
```

Example 7–5 (Cont.) Using the VMSINSTAL Command Procedure to Install VAX FORTRAN

```
        +-------------------------------------------------------+
        |        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
```

Example 7–5 (Cont.) Using the VMSINSTAL Command Procedure to Install VAX FORTRAN

For more information on the VMSINSTAL command procedure, read the section on VMSINSTAL in the *VMS Installation and Operation Guide* for your processor.

7.14 Optional Laboratory Exercise

Under the supervision of your course administrator, install one optional product on your system by following the instructions in the product's installation guide.

7.15 Solution to Optional Laboratory Exercise

If you have followed the instructions carefully, the self-test done by the product's installation procedure should be successful. If the procedure does not include a self-test, test the product yourself. If the tests are not successful, consult your course administrator.

7.16 Customizing and Backing Up the Console Volume

After you have installed a VMS system, you should customize the boot environment to match your system configuration. Setting up boot files is system-dependent, and there are more systems than this course alone can cover. The *VMS Installation and Operation Guide* contains precise instructions for setting up the boot files, and you should read the chapter, Startup and Shutdown Procedures.

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 and any other command procedures on console volume.

When backing up the console volume for systems with a removable console volume (VAX 8250, VAX 8350, VAX 8650, VAX-11/780, VAX-11/750, VAX-11/730) use SYS$UPDATE:CONSCOPY.COM to copy the console volume. Follow directions in your *VMS Installation and Operation Guide.*

- Use CONSCOPY.COM to copy the console files to a directory on the VAX system.
- 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 (VAX 8810, VAX 8820, VAX 8830, 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* if you need to reinstall the console files.

After you have installed a VMS system, customize the boot environment to match your system configuration. Redefine the default boot command procedure on your console volume if your system always boots using the console volume. See Selecting a Default Bootstrap Command Procedure in the *VMS Installation and Operation Guide.*

If needed, use the SYS$UPDATE:DXCOPY.COM procedure to modify the contents of the default boot file first. Use this procedure to modify any command procedure on the console volume. (Use the DXCOPY procedure to copy a boot file to disk. Then modify the file using a text editor. Finally, the DXCOPY procedure copies it back to the console volume.)

After you have customized the console volume, use the command procedure SYS$UPDATE:CONSCOPY.COM to back it up. This procedure invokes the EXCHANGE utility to copy the entire contents of the console volume to a file on the system disk. For systems dependent on the console volume to boot, create several copies of the console volume with the same procedure.

NOTE
When you upgrade a system, the upgrade procedure typically modifies the files on the console volume. You will probably not be able to boot the upgraded system using an earlier version of the console volume. Therefore, you should always back up your console volume on the system disk (using the CONSCOPY.COM procedure) whenever you upgrade the system or perform any update procedure that modifies the console volume. Then you will be able to generate a new copy of the console volume (in case the one you are using fails) that matches the current version of the VMS system on your system disk.

In this example, when you restore the disk, place a scratch diskette in CSA1. After restoring to the scratch diskette, leave that disk in place and treat the original diskette as the backup copy. As soon as possible, reboot the system to test the new disk. Example 7-6 was generated on a VAX 8350 system.

```
$ @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:
```

Example 7–6 Saving the Contents of the Console Volume

```
%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.VFO copied to CONSOLE_SAVE:FG81.VFO, 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.VFO       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 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 <RETURN> when ready: RETURN
%MOUNT-I-MOUNTED, VAX console  mounted on _CSA1:
```

Example 7–6 (Cont.) Saving the Contents of the Console Volume

The *VMS Installation and Operation Guide* for your processor has informative sections on:

- Bootstrap command procedures
- Customizing a default bootstrap command procedure
- Making copies of your console volume

7.17 Optional Laboratory Exercise

Under the supervision of your course administrator, use the SYS$UPDATE:CONSCOPY.COM command procedure to create a spare copy of the console volume.

7.18 Solution to Optional Laboratory Exercise

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.

7.19 Creating a New System Disk from Another System Disk

You can build a system disk at any time from a previously built system disk by invoking the command procedure SYS$UPDATE:VMSKITBLD.COM, shown in Example 7-7. You can run the VMS system from the disk you are copying because this procedure does not affect the operation of the VMS system, it simply copies system files to the output volume.

```
$ @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 = VAXVMSRL052]:
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$DUA0:, has been initialized.
%MOUNT-I-MOUNTED, VAXVMSRL052    mounted on _KNIFE$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
```

Example 7–7 Building a System Disk with VMSKITBLD.COM

For more information on how to use the VMSKITBLD procedure, read the section on Building and Copying a VMS Operating System Disk in the *VMS Installation and Operation Guide* for your processor.

7.20 Standalone BACKUP

The VMS system uses the system disk constantly while the system is running. If you try to copy the system disk using on-line BACKUP, the copy will not be accurate, because on-line BACKUP runs from the system volume. To copy any volume accurately, you should not use the volume during the backup procedure.

To create an accurate copy of the system volume, you can do either of the following:

- Shut the system down and reboot it using a system volume in a different drive. Then mount the primary system volume as a data volume and use on-line BACKUP (running on the alternate system volume) to copy it to disk or tape.
- Shut the system down and reboot it using the standalone BACKUP kit on your console volume or system volume. Since you are booting the kit instead of the normal VMS system, you can use standalone BACKUP to copy the system volume to disk or tape.

Module 6 discusses how to shut down your system and reboot it specifying an alternate system volume. Module 4 discusses how to use on-line BACKUP to copy a disk to another disk or to tape.

If your distribution kit for installing a major release of the VMS system does not contain a standalone BACKUP kit on console media, you will have to create one. If the standalone BACKUP kit you are presently using becomes corrupted, you will have to create another one. This module discusses how to create and use a standalone BACKUP kit.

7.20.1 Creating a Standalone BACKUP Kit

Invoke the SYS$UPDATE:STABACKIT.COM command procedure to create a stand alone BACKUP kit on console media. You will need several scratch volumes of the console media supported by your processor to complete this procedure. In Example 7-8, the procedure creates a standalone BACKUP kit on the system disk. Read the notes following the example to learn more about creating a BACKUP kit on console media.

Creating a standalone kit on console media requires (minimum):

- Five RX01 diskettes
- Four RX50 diskettes
- One TK50 tape cartridge

- The first time you build a kit, build all the volumes in the kit.
- If you choose the ANALYZE/MEDIA option to check the integrity of your scratch volumes, it will check each volume (using about 15 minutes or longer per volume).
- After you have built the kit, remove the last volume from the console drive and place your console volume in the drive. The procedure will then mount your console volume as a protected volume. If you do not want your console volume to remain mounted, shut the system down and reboot it. (The boot procedure does not normally mount the console volume.)

To create a standalone BACKUP kit on the system volume, you should also use the SYS$UPDATE:STABACKIT.COM command procedure. This procedure copies the kit to the [SYSEXE] directory on the root directory you specify. The system software usually resides in the SYS0 root directory, so you can choose any other available root directory from SYS1 to SYSF. However, the standard choice is the SYSE root directory. See the *VMS Installation and Operation Guide* for more information.

7.20.2 Using a Standalone BACKUP Kit

Usually, you use a standalone BACKUP kit to back up your system volume. Whether you boot the kit from the SYSE root directory on the system volume or from the console volume, the software in the kit executes in system memory. The kit does not use the normal system software (in the SYS0 root directory) or scratch space on the system volume outside of the SYSE root directory. Therefore, you will be able to create an accurate copy of the system software. (Actually, whenever you boot from an alternate root directory, you are, in a sense, booting a different system volume. The boot procedure for an alternate does not use any of the software in the normal root directory.)

```
$ @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.

$
```

Example 7–8 Creating a Standalone BACKUP Kit on Console Media

To boot the kit from the console volume, load the console volume in the console drive and enter a boot command. (On the VAX-11/750, you either load the console volume or the first volume in the standalone BACKUP kit, as shown in Table 7-4). The command you choose depends on the kind of processor you are using. In each case, you are asking the system to boot from the console volume instead of from a system volume. When the boot procedure reaches the point where it would normally use software on the system volume, it sends a message to the console terminal to request that you load a console volume containing the boot software (the first volume in the standalone BACKUP kit).

The system reads the software from the first volume in the kit, and loads the software into system memory. The procedure then requests the next volume. This continues until the system has processed all the volumes in the kit. Then it displays the DCL prompt ($) at the console terminal. At this point, you can enter standalone BACKUP commands. The *VMS Backup Utility Manual* indicates which BACKUP qualifiers are also valid with standalone BACKUP.

Table 7–4 Booting the Console Volume

Processor Type	Command	Comments
VAX 8800/8700 8600/8500	>>>BOOT dddnnn /R5 : data	Default boot command procedure uses the file DEFBOO.COM if no device name is given.
VAX 8200/8300	B/25:800 CSA1	To start the BOOT58 program, make sure the console RX50 is in console diskette drive CSA1 before entering this command.
VAX 6200 Series	B/XMI:a/BI:b DUu	No console media.
VAX-11/780	>>>BOOT CS1	Load the console volume in the console drive before entering this command.
VAX-11/750(CCL)	>>>B DDAO	Load the first volume of the standalone BACKUP kit in the console drive before entering this command. This processor contains microcode that performs the first part of the boot procedure.
VAX-11/750 (BOOT58)	BOOT CSA1	To receive the BOOT58> prompt, you must set the BOOT DEVICE switch to position A, load the console volume in the console drive, and press the INITIALIZE button.
VAX-11/730	>>>B CS1	Although the VAX-11/730 normally boots using the console volume in the CSA2 device, use the CSA1 device in this case because you want the boot procedure to use volumes that you will load into the CSA1 drive. Before entering this command, be sure to load a console volume into the CSA1 console device.

The *VMS Backup Utility Manual* indicates which BACKUP qualifiers are also valid with standalone BACKUP.

As of VMS system Version 5:

- If the system has at least 2 Mb of memory, standalone BACKUP does not need to page; so you can remove the volume you booted it from.
- You don't have to reboot standalone BACKUP between operations.
- Standalone BACKUP, like on-line BACKUP, performs tape label checking.

7.20.3 Using a Standalone BACKUP Kit

To boot standalone BACKUP from the system disk, see your *VMS Installation and Operation Guide*. Some systems already have a console command procedure that boots standalone BACKUP. On other systems, you need to create a new command procedure to boot the system from root [SYSE.]

Example 7-9 illustrates backing up the system volume to tape using a standalone BACKUP Kit.

```
>>>BOOT CS1
        CPU HALTED
 INIT SEQ DONE
 LOAD DONE, 00005000 BYTES LOADED
  Please remove the volume "780 console" from the console device.

  Insert the first standalone system volume and enter "YES" when ready: Y

  Resuming load operation on volume "SYSTEM_1:", please stand by . . .

  Please remove the volume "SYSTEM_1" from the console device.

  Insert the next standalone system volume and enter "YES" when ready: Y

  Resuming load operation on volume "SYSTEM_2", please stand by . . .

     VAX/VMS Version 4.7  20-MAY-1987 09:14
  Insert the standalone application volume and enter "YES" when ready: Y

  Resuming load operation on volume "BACKUP", please stand by . . .

  %BACKUP-I-IDENT, Standalone BACKUP V4.0; the date is 10-JUN-1988 19:02:00.82
  $
  $ BACKUP/VERIFY/IMAGE DMA0: MTA0:SAVE.BCK
  %BACKUP-I-SOFTWERRS, 26 recoverable media errors occurred writing
  MTA0:[000000]SAVE.BCK;
  %BACKUP-I-STARTVERIFY, starting verification pass
  %BACKUP-I-SOFTRERRS, 1 recoverable media error occurred reading
  MTA0:[000000]SAVE.BCK;
  %BACKUP-I-RESUME, resuming operation on volume 2
  %BACKUP-I-READYWRITE, mount volume 2 on _SALONE$MTA0: for writing
  Enter "YES" when ready: Y
  %BACKUP-I-SOFTWERRS, 19 recoverable media errors occurred writing
  MTA):[000000]SAVE.BCK;
  %BACKUP-I-STARTVERIFY, starting verification pass
  %BACKUP-I-SOFTRERRS, 2 recoverable media errors occurred reading
  MTA0:[000000]SAVE.BCK;
  %BACKUP-I-IDENT, Standalone BACKUP V4.0; the date is 10-JUN-1988 19:26:47.96
  $
```

Example 7–9 Backing Up the System Volume to Tape Using a Standalone BACKUP Kit

The procedure for using standalone BACKUP consists of the following steps:

- 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 your *VMS Installation and Operation Guide.*
- If you are booting from a removable console volume, standalone BACKUP directs the loading operation. Load appropriate volumes into the console drive as requested.
- After standalone BACKUP is loaded, the DCL prompt appears on the console terminal. At the DCL prompt, enter BACKUP commands.

It takes time to create a special boot command procedure to use a kit on the system volume. This method is much faster to use for backups however, because loading software into memory from a system volume (SYSE) is much faster than loading software into memory from any type of console volume.

The *VMS Installation and Operation Guide* has informative sections on:

- Building Standalone BACKUP on Console Media
- Using Standalone BACKUP on Disk
- Creating a boot procedure

7.21 Optional Laboratory Exercise

Use standalone BACKUP to copy the contents of one disk to another. (If your system has only one disk, perform a disk-to-tape BACKUP.)

This laboratory exercise requires your exclusive use of the computer for 30 to 60 minutes. Inform the course administrator that you want to perform this lab. If you are forced to delay until later, continue with the rest of the module, but return to this lab as soon as possible.

- Obtain a standalone BACKUP kit on console media from your course administrator. If no kit is available, create one. Use Example 7-6 and your *VMS Installation and Operation Guide* as references.

  (If your course administrator prefers, create a standalone BACKUP kit on the system disk instead. Use the discussion on booting from an alternate root directory in this module and your *VMS Installation and Operation Guide* as references. 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 lab.)

- Obtain a scratch disk (or set of scratch tapes) and a sample data disk (or sample system disk) from the course administrator. You will be copying the entire data (or system) disk to the scratch disk (or set of tapes).

- Shut down the computer using the SYS$SYSTEM:SHUTDOWN.COM procedure. (Module 6 discusses this command procedure.)

- Once the system is fully shut down, boot the standalone BACKUP program. Use Table 7-1, Example 7-7, and your *VMS Installation and Operation Guide* as references.

- 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 course administrator check both volumes and your BACKUP command before you press the RETURN key to enter the command.

- Once the BACKUP operation is finished, unload the two scratch and sample packs used for the lab and reload (if necessary) the normal system and data packs.
- Press the CTRL/P key sequence at the console terminal to prepare to boot the system. Halt the computer if necessary.
- Replace the BACKUP kit console media with the console media containing the boot software.
- Boot the VMS operating system from power on. (If you copied the system disk to an output disk, you might want to load the output disk and try to boot from it. Module 6 discusses how to boot the system from the default device or from a device you specify.)

7.22 Solution to Optional Laboratory Exercise

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 succeeded.

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). Module 4 discusses how to restore tapes to a scratch disk.

The *VMS Installation and Operation Guide* has informative sections on:

- Building Standalone BACKUP on Console Media
- Using Standalone BACKUP on Disk
- Creating a boot procedure

MAINTAINING SYSTEM INTEGRITY

8.1 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, get access to devices, and run long programs without fear that their work will be corrupted.

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 or hardware error caused it by analyzing recent error logs and running 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 arises. 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.

8.2 Objectives

To collect useful troubleshooting information, the system manager must:

- 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

8.3 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. *VMS Audit Analysis Utility Manual*
8. *Digital Site Preparation Guide*
9. *Self-Maintenance Handbook*
10. *Site Management Guide*

8.4 Hardware Maintenance

The system manager must maintain the hardware connected to the system. Hardware maintenance involves:

- Care of devices (CPU, disks, terminals, etc.)
- Care of media (tapes, disks, line printer paper, etc.)

8.4.1 Handling and Storing Media

Each device and each type of media must be handled properly to maintain system integrity. Most devices and media are sensitive to rough treatment, dust, sudden impact, electrical noise, and other abuse. Therefore, only trained personnel should handle media or load media to decrease the chance of breaking it. Also, keep unused media in closed cabinets (preferably metal) to decrease loss of information caused by dirt or electrical noise.

If information on storage tapes or disks is valuable, consider off-site storage in a fireproof vault as further protection. Digital Equipment Corporation has several of these vaults available, referred to as Record Management Services Centers.

8.4.2 Cleaning Media

Maintaining devices and media also includes cleaning. A **customer care** document is available for each peripheral shipped. This document details the cleaning and maintenance procedures the manager or operator should perform for each device. Typical tasks include:

- Cleaning the heads, rollers, and vacuum area of a tape drive
- Cleaning the outside cover of a disk drive (so dust and dirt will not fall into it)
- Changing printer ribbons
- Wiping the face of a terminal screen

A Digital Field Service representative should do the more complicated and delicate tasks, such as:

- Aligning a tape drive
- Cleaning the inside of a disk drive
- Fixing print heads

A manager or operator should **never** do any task not listed in the customer care document for a device.

Cleaning media (disks or tapes) is not included in the manager's tasks (unless the manager is trained and has specialized equipment). This job is normally referred to a specialist. The manager should schedule appointments with the specialist and ensure that the specialist has access to the media at that time.

Cleaning does not disturb the data on media, but the specialist also checks the media for safety. If the media is unsafe to load in a drive (because it might cause the drive to fail), the specialist will place a "reject" tag on it. Rejected media cannot be recovered. Therefore, the manager must be sure to back up all media before sending it to be cleaned.

8.4.3 Maintaining the Environment

Hardware maintenance involves maintaining the environment. Equipment fails more often in an improperly set up, and poorly maintained environment. The *Digital Site Preparation Guide* contains detailed discussions on how to set up and maintain a computer room environment, such as:

- Temperature
- Humidity
- Power
- Flooring
- Dust
- Electrical noise
- Lighting
- Fire and safety precautions

Understanding the effect the environment has on a computer will help prevent, diagnose, and correct problems more easily.

8.4.4 Preventive Maintenance

The system manager should schedule regular appointments with the Digital Field Service representative. The Field Service representative cleans devices and runs diagnostics as a preventive measure (referred to as PM - preventive maintenance). If these visits are not scheduled regularly, the system could develop undetected problems, which might result in a system or disk failure.

It is especially important to keep all drives aligned properly. Disk and tape drives normally become misaligned over a period of time. You may not notice that they are misaligned because they can still read media they wrote. However, if they become badly misaligned, they will not be able to read media produced on other, properly aligned drives. If you then align them, they cannot read media they wrote previously. Also, if you realign all drives on your system on the same day, it is possible that **none** of them will be able to read previously written volumes. Therefore, schedule alignments often, rotate the schedule so all are not aligned on the same day, and create your backup volumes on the most recently aligned drive.

The Field Service representative logs all cleaning done, diagnostics run, and parts replaced in the Site Management Log (shipped with every system). The manager should keep the log near the system so any Field Service or Software Service representative can review the history at any time. The history often helps a Digital representative locate the source of a problem quicker. Space is also available in the log for the system manager to record any problems that have been found or cleaning that has been done.

Some customers maintain their own hardware and find the *Self-Maintenance Handbook* helpful. This handbook covers service planning, site preparation and installation, remedial and preventive maintenance, and product upgrades. It also enables users to develop a self-maintenance program, including:

- How to determine whether self-maintenance is the most economical approach
- How to plan manpower needs
- How to determine labor charges
- How to schedule preventive maintenance

8.5 Software Maintenance

The system manager must maintain the software on the system, including:

- Maintaining the system files
- Maintaining user programs and data files

The system files include the operating system and layered product files as well as the manager's files. To maintain the former, install updates and upgrades correctly, referring to the appropriate installation guide. Module 7 discusses software installation in more detail. To maintain the system files such as SYSUAF.DAT, QUOTA.SYS and ACCOUNTNG.DAT, run the appropriate utilities on a regular basis to observe their contents. Keep all UAF records up to date, and observe the contents of the other data files to ensure that the system is being used effectively.

Back up system and user files on a regular basis. On many systems, the system software is on a separate disk from the user software. You may choose to establish a different backup schedule for each, but be sure every disk volume you use is on the schedule. Follow full backups with several incremental backups before doing another full backup. Read the chapter on BACKUP in the *Guide to Maintaining a VMS System* for more suggestions on backup schedules.

Users typically maintain the contents of their own files. However, some users execute long, complicated programs that make many modifications to large data files. These users should design their applications to recover from program errors and system failures. However, errors sometimes occur from which programs cannot successfully recover.

8.5.1 System Security

Different installations usually require different levels of security.

At one extreme, users can be allowed physical access to the computer as well as all privileges to do whatever they choose. At the other extreme, the computer and peripherals could be in a locked room accessible only to operators, and users could have few or no privileges. Between these two extremes, the manager may use the tools available to grant specific levels of access and privilege to specific users. The manager has control over the level of security implemented.

8.5.2 Physical Security

You should consider the physical security of the computer first because a user with access to the console terminal can bypass software controls. If dial-up lines are available, users can attempt to break in to the system from outside, presenting a security risk.

Media storage is also a security issue. Cabinets with locks are available for storing sensitive media as well as less expensive cabinets.

To allow access to the room but not to the media, you must obtain the more expensive cabinets. If users have no need to gain access to the computer or the media, the room should remain locked to prevent unnecessary traffic (traffic brings dust and dirt into the room) and to reduce the cost of maintaining security.

8.5.3 Software Security

When the computer itself is physically secure, the manager must maintain some level of software security. Use common sense in employing the commands and tools discussed in previous modules.

- Keep the SYSUAF.DAT and NETPROXY.DAT files up to date by removing old users.
- Use the flags and hourly restrictions provided through the AUTHORIZE utility to clearly define when and how each user can get access to the system.
- Insist on nontrivial passwords for all users (especially for sensitive accounts) and encourage users to change passwords frequently.
- Do not publicize dial-up numbers.
- Do not permit WORLD access to SYSUAF.LIS. (Otherwise, users can see what accounts have privileges, and try those user names with various passwords.)
- Encourage the use of file protection codes and ACLs.
- Request secondary passwords in login files of sensitive, captive accounts.
- Give users only the privileges they need and can use responsibly.
- Create accounting reports periodically to check queue usage, peripheral usage, etc.
- Label disks and tape in a systematic manner. (Operators and users should also do this.)

If your system needs to be more secure, choose any or all of the security tools provided by the VMS system, including:

- Erase-on-delete and erase-on-allocate characteristics for files and volumes
- Login security
- Break-in detection
- Security auditing

8.5.3.1 Erase-on-Delete and Erase-on-Allocate

The terms **erase-on-delete** (EOD) and **erase-on-allocate** (EOA) refer to the activity of the file system when a file is deleted, created, or extended. When you delete a file that is set for erase-on-delete, the file system overwrites the blocks that it used with zeros (or a pattern of your choice). This is done synchronously with the delete so the blocks are not available for allocation until they have been overwritten.

When you create or extend a file on a volume that is set for erase-on-allocate, the file system (by default) overwrites blocks with zeros (or the pattern of your choice) before allocating them to you.

These erasures prevent other users from allocating previously used disk blocks and reading good data. Users may implement EOD on a file-by-file basis, and managers may implement it on a volume-by-volume basis (see Table 8–1). EOA (or high-water marking) is set automatically for each volume when you initialize it. To improve performance on less security-sensitive or fragmented volumes, disable high-water marking (EOA) with the command:

```
$ SET VOLUME/NOHIGHWATER_MARKING
```

You can also initialize a volume for no high-water marking with the command:

```
$ INITIALIZE/NOHIGHWATER_MARKING volume-name
```

Table 8–1 Setting Erase-on-Delete for a File or Volume

Command/Qualifier	Function
`$ 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.

For more information on erase-on-delete and erase-on-allocate, read the two chapters on Disk Scavenging in the *Guide to VMS System Security*. Also read the command description and the applicable qualifier descriptions for the commands SET FILE, SET VOLUME, and INITIALIZE in the *VMS DCL Dictionary*.

8.5.3.2 Login Security

Users are normally assigned a user name and a password in the authorization file, SYSUAF.DAT. They must enter both (unless the password is null) before using the system.

You can define a secondary password for some accounts with the Authorize utility, as described in Table 8–2. If a UAF record contains a secondary password, the system prompts anyone who logs in with the user name of that UAF record for the secondary password, as shown in Table 8–4.

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 -`
`_UAF> /PASSWORD=("",SECOND)`
	To add a secondary password to a UAF record, use the /PASSWORD qualifier with the MODIFY command. It does not affect the current value of the primary password if you specify a null first password string.

`$ SET PASSWORD`	Modifies your own primary password.
`$ SET PASSWORD/SECONDARY`	Modifies your own secondary password.
`$ SET PASSWORD/GENERATE`	Requests or requires (using the qualifier /FLAG=GENPWD in AUTHORIZE) that the VMS system 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.

For terminals in remote or sensitive locations, you can require a **system password**. Define the system password by using the SET PASSWORD/SYSTEM command. Then, use the SET TERMINAL command to define certain terminals as "system password required" terminals (see Tables 8-3 and 8-4).

Table 8–3 Defining a System Password for a Terminal

Steps	Function

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 -`
`_$ /SYSPASSWORD /PERMANENT -`
`_$ TXA2`
	Set the specified terminal to require the system password to be used. To log in to this terminal a user must press the RETURN key and enter the system password (no prompt is given). If successful, the user continues the normal login procedure when the "Username:" prompt appears.

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, while another knows the secondary. Therefore, both must be present whenever this account is used.

Terminal requiring a system password	`SYSPASSWORD`
	`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.

You can use the Authorize utility to modify other fields in UAF records that affect passwords and login activity. You can set:

- An expiration date on a UAF record. After this date, no user can log in using that record.
- A minimum password length. Users modifying their password with the SET PASSWORD command must choose a password with this minimum length.
- An expiration date for the password. For several days before the password expires, the system displays a message each time the user logs in, telling him or her to change the password. Finally, the system displays a last chance message. If the user does not change the password during that login period, the system disables logins for the account.
- The password to be "preexpired" when you create the account. When a user logs in to the system with the user name of a UAF record that has a preexpired password, the system displays the last chance message. The user must change the password during that first login period, or the system disables logins for that account.

Devices that are not shareable, such as terminals, have an owner UIC and a protection code that determine what processes can allocate the device. Table 8–5 shows how to use the DCL command SET PROTECTION/DEVICE to control device ownership and protection.

> NOTE
> At many sites, the UIC owner of all terminals is the system manager, and only users in the System and Owner categories are allowed to allocate terminals. This level of protection prevents users from allocating a terminal with the ALLOCATE command, but it does not prevent them from logging in on any terminal.

You can also apply an access control list to terminals and to tape drives with the SET DEVICE/ACL command to refine the limits on the user's access.

Table 8–5 Establishing Ownership and Protection of Terminals and Other Nonshareable Devices

	Format of $ SET PROTECTION/DEVICE	
Operation	**Command (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 can 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.

8.5.3.3 Additional References

- For more information about applying access control lists to devices, read the section on protecting devices in the chapter, File Protection Features, in the *Guide to VMS System Security*. Also read the discussion on device protection in the *VMS DCL Dictionary*.
- For more information about applying UIC protection codes to devices codes, read the section on devices in the chapter, File Protection Features, in the *Guide to VMS System Security*. Also read the description for the SET PROTECTION/DEVICE command in the *VMS DCL Dictionary*.

You can protect terminals from password-collecting programs by setting the secure server characteristic on them (shown below):

```
$ SET TERMINAL/PERMANENT/SECURE_SERVER/DISCONNECT TTC1:
```

The secure server ensures that the VMS login program is the only program able to receive your user name and password. Once you have set this characteristic for a terminal, users must press the BREAK key, not the RETURN key, to receive the Username: prompt.

NOTE
The SECURE_SERVER characteristic has no affect on terminals with the AUTOBAUD characteristic set.

Break-In Detection at Login

Unauthorized users may attempt to gain access to an account on your system by trying to guess the password of a valid user. Unless you restrict these users by implementing break-in detection, they may guess the correct password and gain entry to your system. If you have valuable data or programs, this breach of security can be a serious problem.

When you implement break-in detection, the system records information about login failures in **suspect lists**. When users fail to log in successfully, their user names are recorded in the **user name suspect list**; the name of the terminal they are using is recorded in the **terminal name suspect list**; and the name of the node they are attempting to log in from is placed on the **node name suspect list**.

If attempts continue to fail within a specified time period and exceed the break-in limit set by the system manager, the user name, terminal name and node name are placed on respective **intruder lists**. Intruders are subject to evasive action by the system for a variable period of time. Evasive action includes responding to login attempts with false messages or with no messages. Even if an intruder enters a correct user name and password, the system will respond with an error message.

If a suspect enters a correct user name and password before he or she becomes an intruder, the system allows him or her to log in.

The manager defines the level of break-in detection using two SYSGEN parameters. Another parameter defines the amount of time intruders are evaded, as described in Table 8–6.

Break-in information lists are kept in memory. If many suspects and intruders are listed, there is less room for normal system activity, and system performance will suffer. Therefore, if performance degrades suddenly on a system with break-in detection established, the manager should suspect that a break-in attempt may be occurring. To find out more information, the manager should check the security log.

Table 8–6 SYSGEN Parameters for Break-In Detection

Parameter	Comments
LGI_BRK_DISUSER	Once an intruder has been detected, the VMS system 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, the VMS system sets this parameter to 1 so that terminals and user names are tracked together. If you use terminal servers, set this parameter to 0 (track only user names), since a LAT port (on the VAX computer 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.

```
❶ RETURN

❷            This is node TIDE
  Username: JONES
❸ Password:
  User authorization failure
❹ Username: JONES
  Password:
  User authorization failure
❺ Username: JONES
  Password:
               Welcome to node TIDE running V5.0
❻ Last interactive login on Saturday, 11-JUN-1988  08:59
❼          2 failures since last successful login
  $
```

Example 8–1 Break-in Suspect Logs in Successfully

Notes on Example 8–1:

❶ User presses the RETURN key.

❷ System displays the announcement message and "Username:" prompt.

❸ User enters an incorrect password and fails to log in.

❹ User presses RETURN again. This time the system displays no announcement message before displaying the "Username:" prompt. Two SYSGEN parameters govern whether the announcement message appears or not: LGI_RETRY_TMO and LGI_RETRY_LIM. The value of the first parameter determines how long the system waits between retries (default is 20 seconds). The value of the second parameter determines how many retries are allowed. These parameters affect dial-up users primarily because the system breaks their connection after these limits are reached, forcing the user to dial in again. The on-line user simply presses RETURN and receives the announcement message as well as the "Username:" prompt.

❺ User presses RETURN, logs in successfully and receives the welcome message and two other messages.

❻ The first message tells the user when someone last logged in successfully using that user name and password. If this message shows a login more recent than the last time this user actually logged in, the user should report it as someone may have guessed his or her password.

❼ The second message tells the user how many login attempts were made before the login was successful. If there is a difference between the number of failures reported and the number of times he or she actually caused a failure, the user should report it as someone may be trying to break in to the system.

```
  RETURN
               THIS IS NODE TIDE

  Username:  WEBSTER
❶ Password:
  User authorization failure
  Username:  WEBSTER
  Password:
  User authorization failure
  Username:  WEBSTER
  Password:
  User authorization failure

  RETURN
❷            This is node TIDE

  Username:  WEBSTER
  Password:
❸ User authorization failure
  Username:  WEBSTER
  Password:
❹ User authorization failure
  Username:  WEBSTER
  Password:
  User authorization failure
```

Example 8–2 Break-in Suspect Becomes an Intruder

Notes on Example 8–2:

❶ The user enters an incorrect password on three login attempts and receives a user authorization failure message each time.

❷ The user presses the RETURN key and receives the announcement message again because the LGI_RETRY_LIM parameter is set to 3 on this system. A dial-up user loses the carrier after the third unsuccessful attempt.

❸ The user enters an incorrect password and receives an error message. If the password were correct, the user could still log in at this fourth attempt because the value of the LGI_BRK_LIM parameter on this system is 5.

❹ The user enters the correct password, but since this is the sixth attempt, the system begins evasive action. (Evasive action includes responding to correct input with error messages for a variable period of time.) This user name cannot be used for a while to log in. If a user reports being unable to enter his or her account even after entering the correct password, an intruder might be trying to break in to the system. Change the password immediately and read the security audit reports on the console terminal.

8.5.3.4 UIC and ACL Protection

VMS System Protection Using UICs

A group ususally consists of users who work in the same department or on the same project. Group numbers are assigned by the system manager in the group number part of their UICs. (The Managing Files module discusses UICs in more detail.) When you log in, the system creates a process for you and assigns the UIC defined in your UAF record to your process. Thereafter, you have access to files or to other users based on the value of your UIC.

Therefore, before assigning UICs to users, analyze the extent to which the user processes on your system need to:

- Share access to files, volumes, and devices on your system
- Communicate interactively with each other
- Affect or control each other

Table 8–7 suggests how to define groups based on user needs when you set up their UICs. You can increase the ability of processes to interact and share information by assigning them UICs in the same group, or you can decrease this ability by assigning them UICs in different groups. Table 8–8 details the effects of UIC group assignment on process interaction.

Table 8–7 Dividing Users into Groups

User Situation	Course of Action	Result
Users should 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 can 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 do not need to 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) 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.	1) 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. 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.

8.5.3.4 UIC and ACL Protection

VMS System Protection Using UICs

A group ususally consists of users who work in the same department or on the same project. Group numbers are assigned by the system manager in the group number part of their UICs. (The Managing Files module discusses UICs in more detail.) When you log in, the system creates a process for you and assigns the UIC defined in your UAF record to your process. Thereafter, you have access to files or to other users based on the value of your UIC.

Therefore, before assigning UICs to users, analyze the extent to which the user processes on your system need to:

- Share access to files, volumes, and devices on your system
- Communicate interactively with each other
- Affect or control each other

Table 8–7 suggests how to define groups based on user needs when you set up their UICs. You can increase the ability of processes to interact and share information by assigning them UICs in the same group, or you can decrease this ability by assigning them UICs in different groups. Table 8–8 details the effects of UIC group assignment on process interaction.

Table 8–7 Dividing Users into Groups

User Situation	Course of Action	Result
Users should 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 can 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 do not need to 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.	1) 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.
	2) 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 properly setting the protection codes of the files, volumes, and devices.
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 System Protection Using ACLs

Although you should divide system users into appropriate UIC groups, occasionally a user in one group may need regular access to a certain file belonging to another group. You could give the SYSPRV privilege to the user desiring access, or you could change the protection on the file to allow World users more access. However, these solutions are too broad for normal systems. Most managers are reluctant to give the powerful SYSPRV privilege to users, and the file owner may not want to allow access to all users.

While VMS system protection using UICs is usually sufficient for most files on your system, the VMS operating system provides an additional layer of protection to solve this problem and others like it. Because this extra protection is based upon identifiers, users must possess the required identifier to gain access to the protected file, even if VMS protection admits them.

To enable this type of protection, record the access information for a file in an Access Control List (ACL) for that file. Define the identifiers held by a user in the rights database. When a user logs in, the VMS system creates an access rights list (using the information in the rights database) consisting of the identifiers he or she holds. When a user attempts to gain access to a file, the VMS system compares the contents of the file's ACL with the contents of the user's access rights list to determine the type of access to grant.

For example, if the ACL for a file contains the identifier name CHEMISTRY, the system examines your access rights list to see if you hold it. If you do, the system examines the ACL on the file to see what type of access is allowed for holders of that identifier. The access can be any combination of Read, Write, Execute, and Delete, or it could be None.

If the type of access is None, the system does not allow you to gain access to the file. (However, if you have a system UIC, or are the owner of the file, the system allows you the access allowed for your user category in the VMS system protection code on the file.) If the type of access is Read, Write, Execute or Delete, you are allowed that type of access only. If you do not hold the identifier CHEMISTRY or any others listed in the file's ACL, the system determines your access rights according to your UIC and the VMS system protection code on the file, as shown in Table 8–9.

You can specify the access rights for users holding an identifier differently for each file. For example, the ACL for FILE.DAT allows users holding the identifier CHEMISTRY to read FILE.DAT, while the ACL for DATA.DAT allows users holding the same identifier to read, write, and delete DATA.DAT.

Table 8–9 ACL- and UIC-Based Protection

File Name/ 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.

Table 8–10 Terminology Used to Discuss VMS System 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	A 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	The 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 the VMS system uses to perform all protection checks.
Access Control List	A 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 can 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	An 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.

System-Defined and User-Defined Identifiers

The system creates several default identifiers for you at system startup and adds them to the rights database. Their names are: LOCAL, DIALUP, REMOTE, INTERACTIVE, NETWORK, and BATCH. These identifiers allow you to control user access to a file according to how the user is logged in. For example, if a file's ACL contains the following ACE, the user holding the MATH identifier has the specified access only when he or she is logged in at a local terminal:

```
(IDENTIFIER=MATH+LOCAL,OPTIONS=NONE,ACCESS=READ+WRITE+EXECUTE)
```

The system also creates one identifier for each UAF record that contains a unique UIC. It stores these identifiers in the rights database. The identifier names are the same as the user names of the UAF records. Once an identifier exists for a UIC value, you cannot create another with the same UIC value. Therefore, when you create a new UAF record that has the same UIC value as an existing record, the Authorize utility associates the identifier for that UIC with the new record, as shown in Example 8–3.

```
  $ SET DEFAULT SYS$SYSTEM
  $ SET PROCESS/PRIVILEGE=SYSPRV
  $ RUN AUTHORIZE
  UAF>
❶ UAF>SHOW/ID/VALUE=UIC:[11,204]
  unable to complete show command
  %SYSTEM-F-NOSUCHID, unknown rights identifier
  UAF>
❷ UAF>ADD TOM/UIC=[11,204]/ACCOUNT=VMS/DEVICE=WORK1/DIRECTORY=[TOM]
  user record successfully added
❸ identifier TOM value:  [000011,000204] added to RIGHTSLIST.DAT
  UAF>
❹ 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>
❺ UAF>SHOW/IDENTIFIER/VALUE=UIC:[11,204]
     Name                          Value             Attributes
     TOM                           [000011,000204]   NORESOURCE
  UAF>
❻ UAF>SHOW/IDENTIFIER/USER=TOM
     Name                          Value             Attributes
     TOM                           [000011,000204]   NORESOURCE
❼ 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
  $
```

Example 8–3 Authorize Utility Automatically Creates Identifiers

Notes on Example 8–3:

❶ No identifier exists with a UIC value of [11,204].

❷ This command adds a UAF record with the user name TOM and a UIC value of [11,204].

❸ The Authorize utility creates the identifier TOM with a value of [000011,000204] and adds it to the rights database SYS$SYSTEM:RIGHTSLIST.DAT.

❹ These commands add two UAF records with user names DICK and HARRY. These have a UIC value of [11,204] also. Notice that the Authorize utility does not create a new identifier for them because an identifier name for the UIC value [11,204] already exists.

❺ The identifier name for the value [11,204] is TOM.

❻ The UIC identifier belonging to the UAF record for TOM is TOM.

❼ The UIC identifier belonging to the UAF record for HARRY is TOM and the UIC identifier belonging to the UAF record for DICK is TOM. Users logging in to the DICK or HARRY accounts can gain access to any object that users of the TOM account can gain access to.

Finally, the system creates one identifier for each UIC group represented in the UAF and stores that identifier in the rights database. The name of a group identifier is the same as the name in the **account field** of the UAF records in that group. The value of a group identifier is a UIC value in the form [group-number,177777]. For example, the UIC value for the group identifier for group 11 is [000011,177777].

The system requires that each identifier has a unique value, so the ACL protection scheme will work properly. However, you need not be concerned about what the values actually are. When you create a general identifier (such as MATH or CHEMISTRY), the Authorize utility normally gives it a unique hexadecimal value. You can specify a different value, but since you only use the identifier **name** in an ACL, it does not really matter what the value is (as long as it is unique).

In Example 8–4, the Authorize utility displays some of the identifier names and values stored in the rights database for a sample system. In Example 8–5, the Authorize utility is used to create new identifiers and allow various users to become holders of them. Read the notes following each example and refer to Table 8–11 to better understand the commands used in the examples.

Table 8–11 Managing the Rights Database Using the AUTHORIZE Command

Command	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 can 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> REMOVE/IDENTIFIER PAYROLL TOM`
	This is the converse of GRANT/IDENTIFIER.

Use the DCL command SET RIGHTS_LIST to modify process and system rights lists. For example, to add the identifier ADMIN to your process rights list, use the following command:

```
$ SET RIGHTS_LIST/ENABLE ADMIN
```

When using the SET RIGHTS_LIST command, you must specify either the /ENABLE or the /DISABLE qualifier to enable or disable an identifier. Table 8–12 lists some of the qualifiers for this command.

Table 8–12 Using the SET RIGHTS_LIST Command

Qualifier	Function
`/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) list is to be modified. Requires CMKRNL privilege.

```
  $ SET DEFAULT SYS$SYSTEM
  $ SET PROCESS/PRIVILEGE=SYSPRV
  $ RUN AUTHORIZE
❶ UAF>SHOW/IDENTIFIER *
    Name                             Value              Attributes
    BATCH                            %X80000001         NORESOURCE
    BIERLY                           [000011,000340]    NORESOURCE
    CHAPUT                           [000031,000030]    NORESOURCE
    CHERPAS                          [000071,000040]    NORESOURCE
    DATACOMM                         [000131,177777]    NORESOURCE
    DECNET                           [000240,000240]    NORESOURCE
    DIALUP                           %X80000002         NORESOURCE
    FIELD                            [000001,000010]    NORESOURCE
    FIELDTEST2                       %X80080011         RESOURCE
    FRIEDMAN                         [000101,000030]    NORESOURCE
    GROUP11                          %X80080021         RESOURCE
    HARBO                            [000031,000007]    NORESOURCE
    HYLNKA                           [000011,000120]    NORESOURCE
    HOUSSELL                         [000011,000130]    NORESOURCE
    ID                               [000071,177777]    NORESOURCE
    INTERACTIVE                      %X80000003         NORESOURCE
    INVENTORY                        [000011,000343]    NORESOURCE
    J65                              [000370,177777]    NORESOURCE
    JAMIESON                         [000011,000330]    NORESOURCE
    KANATA                           [000160,000005]    NORESOURCE
    KENAH                            [000021,000160]    NORESOURCE
    LOCAL                            %X80000004         NORESOURCE
    MANAGEMENT                       %X80080020         RESOURCE
    MARSH                            [000011,000220]    NORESOURCE
    MATH                             %X8001000A         NORESOURCE
    NETMGR                           [000013,000002]    NORESOURCE
    NETPRINT                         [000020,000002]    NORESOURCE
    NETWORK                          %X80000005         NORESOURCE
    OPERATOR                         [000011,000001]    NORESOURCE
    PAYROLL                          %X80010006         NORESOURCE
    PROJECT                          %X8001000B         NORESOURCE
    PROJMGMT                         %X80010007         NORESOURCE
    REMOTE                           %X80010006         NORESOURCE
    SASSMAN                          [000131,000350]    NORESOURCE
    SEVERIN                          [000021,000250]    NORESOURCE
    SMITH                            [000011,000200]    NORESOURCE
    SUPERVISOR                       %X80080022         NORESOURCE
    SYSTEM                           [000001,000004]    NORESOURCE
    SYSTEST                          [000001,000007]    NORESOURCE
    TOM                              [000011,000204]    NORESOURCE
    VARGISH                          [000021,000320]    NORESOURCE
    VMS                              [000011,177777]    NORESOURCE
```

Example 8–4 Displaying Identifiers and Values in the Rights Database

```
❷ UAF>SHOW/RIGHTS/USER=[*,*]
    Name                            Value            Attributes
  Identifiers held by BEYER :
    FIELDTEST2                      %X80080011       NORESOURCE
  Identifiers held by MARSH :
    FIELDTEST2                      %X80080011       NORESOURCE
    MANAGEMENT                      %X80080020       NORESOURCE
  Identifiers held by BIERLY :
    PROJMGMT                        %X80010007       NORESOURCE
    FIELDTEST2                      %X80080011       NORESOURCE
❸ UAF>SHOW/IDENTIFIER/FULL FIELDTEST2
    Name                            Value            Attributes
    FIELDTEST2                      %X80080011       RESOURCE
      Holder                          Attributes
      MARSH                           NORESOURCE
      BIERLY                          NORESOURCE
      BEYER                           NORESOURCE
  UAF>
```

Example 8–4 (Con't) Displaying Identifiers and Values in the Rights Database

Notes on Example 8–4:

❶ Displays the names, values, and attributes of the currently defined identifiers. Notice that some values are UIC values while others are numerical values. The system creates one identifier for each unique UIC and one group identifier for each UIC group when you initialize the system. The Authorize utility creates one identifier for each UAF record you subsequently add (if it has a unique UIC).

❷ Displays the identifiers held by particular users. In this case, only three users (BEYER,MARSH,BIERLY) hold identifiers. Notice that the listed identifiers also appear in the overall list just displayed.

❸ Displays, for each identifier specified, the names of the users who hold that identifier. Notice that this display corresponds to the previous display.

```
  $ SET DEFAULT SYS$SYSTEM
  $ SET PROCESS/PRIVILEGE=SYSPRV
  $ RUN AUTHORIZE
❶ UAF>ADD/IDENTIFIER MATH
  identifier MATH value:  %X8001000C added to RIGHTSLIST.DAT
❷ UAF>GRANT/IDENTIFIER MATH BIERLY
  identifier MATH granted to BIERLY
  UAF>
  UAF>GRANT/IDENTIFIER MATH TOM
  identifier MATH granted to TOM
  UAF>
❸ UAF>SHOW/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
```

Example 8–5 Managing the Rights Database

```
  Authorized Privileges:
    TMPMBX NETMBX
  Default Privileges:
    TMPMBX NETMBX
    Name                          Value              Attributes
  Identifiers held by TOM :
❹   PROJECT                       %X8001000B         NORESOURCE
    MATH                          %X8001000C         NORESOURCE
  UAF>
```

Example 8–5 (Con't) Managing the Rights Database

Notes on Example 8–5:

❶ Adds a new identifier named MATH that you can now include in any ACE.

❷ Grants the new identifier to BIERLY and TOM. These users are now holders of the identifier MATH.

❸ Displays the identifiers belonging to BIERLY and TOM. Notice that users are not required to have the same list of identifiers. You can grant an identifier to any user, allowing you greater freedom in defining groups.

❹ The Authorize utility also displays identifiers held by a user at the end of the user's UAF record when you use the SHOW command.

Table 8–10 summarizes the terminology used in this discussion of VMS system access control.

Restricting User Access to Files

An earlier section of this module introduced the use of access control lists (ACLs) to protect files against access by certain users or groups of users. This section discusses how to create and use ACLs in more detail.

To define an ACL for a file, use the ACL editor. Example 8–6 shows how to use this editor.

```
❶ $ EDIT/ACL CALCULUS.DAT
❷   (screen clears)
       ❸               ❹              ❺
  (IDENTIFIER=MATH,OPTIONS=NONE, ACCESS=NONE)
  (IDENTIFIER=PROJECT,OPTIONS=NONE,ACCESS=READ+WRITE)
                 ❻                      ❼
    (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:
❽ Access Cntrl List:  (IDENTIFIER=MATH,ACCESS=NONE)
                      (IDENTIFIER=PROJECT,ACCESS=READ+WRITE)

  Total of 1 file, 1/1 block.
  $
```

Example 8–6 Defining an ACL for a File

Notes on Example 8–6:

❶ Invoke the ACL editor to add an ACL to the file CALCULUS.DAT.

❷ The ACL editor is a full-screen editor. Therefore, after you enter the EDIT/ACL command and press the RETURN key, the screen clears.

❸ If you are using the editor in the default prompt mode, you will see a prompt for the identifier name at the top left corner of your terminal screen.

Each time you see this prompt, you can enter information to create an Access Control Entry (ACE) in the ACL for this file. Each ACE is composed of three parts: an identifier name, a list of options, and the access rights granted to users holding the identifier.

❹ Although you can use the function keys at this point to complete the first ACE, you can also enter the required text. In this example, the user enters the text, beginning with the identifier name MATH, and concluding with the closing parenthesis. Note that the user also enters the other keywords for the ACE, "OPTIONS=" and "ACCESS=."

❺ To add another ACE to this ACL, the user presses the RETURN key after completing the first ACE. The editor displays the IDENTIFIER= prompt again.

❻ For this ACE, the user enters the identifier name PROJECT. To allow users holding the identifier PROJECT to read and write the file CALCULUS.DAT, include these types of access separated by plus (+) signs after the keyword ACCESS. Identifier names must already exist in the rights database or the ACL editor will generate an error. The next section discusses how to add identifiers to the rights database.

❼ The user completes this ACE with a closing parenthesis and completes the editing session by entering the CTRL/Z key sequence. The editor then assigns this ACL to the file, CALCULUS.DAT.

❽ The new ACL is listed in the file header of CALCULUS.DAT. Users holding the MATH identifier are not allowed access to this file even if they also hold the PROJECT identifier, because the system reads ACLs from top to bottom, granting access according to the first match it finds (see Table 8–9 for exceptions). Users holding the PROJECT identifier can read and write CALCULUS.DAT.

8.5.3.5 Additional References

- To learn more about the ACL editor, read the *VMS Access Control List Editor Manual.*
- For more information on creating and using ACLs and identifiers, and for more information on the default identifiers the system creates, read the section on Access Control Lists in the chapter, File Protection Features, in the *Guide to VMS System Security.*

8.6 Laboratory Exercises

Log in to two accounts on two terminals to do this excercise. You must have write access to the user authorization file to do this exercise. You may also need the SYSPRV privilege.

1. Log in to your first 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 READER.
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 READER in the ACL of your file. In the ACE, specify that holders of the READER 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 READER.
6. Try to type the file from the second account.
7. Run the Authorize utility. Grant the READER identifier to your second account. Show the UAF record for your second account and note the identifiers that have been assigned to it.
8. Log out and then log back in to the second account. Try typing the file again. 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.

8.7 Solutions to Laboratory Exercises

1.
```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF>SHOW /IDENTIFIER *

BATCH
LOCAL
INTERACTIVE
      .
      .
      .
```

2.
```
UAF>ADD /IDENTIFIER READER
```

3.
```
UAF>EXIT
$ SET DEFAULT SYS$LOGIN
```

4. 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:

```
READER,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. Your first account has access to the file because it is the owner, and System and Owner UIC-based protection overrides ACL-based protection.

6. Your second account will get an error message when trying to type the file, since your second 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 READER ACCOUNT2
UAF>SHOW ACCOUNT2
```

   The utility lists the identifiers held at the bottom of the display.

   (Substitute your second account user name for ACCOUNT2 in this example.)

8. ACCOUNT2 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 ACCOUNT2 to have write access to the file.

9.
```
UAF> SHOW /RIGHTS /USER=username
```

10.
```
UAF> LIST /IDENTIFIER *
```

8.7.1 Security Auditing

Security auditing is the act of recording security-relevant events as they occur on the system. Audit analysis is the periodic review of the security audit records. By auditing a discrete set of system events and performing regular audit analysis on this information, you can more easily recognize attempts to compromise the security of your system.

As of VMS V5.2, a detached AUDIT_SERVER process is created at system bootstrap. The AUDIT_SERVER works in conjunction with OPCOM to record events that the system manager wants to audit.

$SET AUDIT is the command that establishes the class(es) of events for which auditing is enabled or disabled.

$SHOW AUDIT displays the current recording status of the AUDIT_SERVER.

The reporting function is provided with the new $ANALYZE/AUDIT command. This is more powerful and convenient than the previous extraction method provided by the command procedure SECAUDIT.COM.

The system manager can select particular activities to track. These activities are grouped into event classes, as shown in Table 8–13.

Table 8–13 Auditing Event Classes

Type of Event	Comments
ALL	All possible events are audited (at some cost to performance).
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, NETPROXY.DAT, RIGHTSLIST.DAT) are audited.
Break-in attempts Detached Dialup Local Network Remote	You can specify all or any 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 used one of the listed privileges to access a file.
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.
Audit	You can record uses of the SET AUDIT command.
Install	You can record INSTALL operations
Mount/Dismount	You can audit each time a mount or dismount request is made.

The audit server process performs the following functions:

- Maintains a private database containing the initial security auditing characteristics. (This information is updated whenever SET AUDIT commands are issued.)
- Creates a binary, cluster-wide (if in a cluster), security audit log file.
- Enables security auditing for a default set of security events.
- Monitors system resources, for example, available disk space, to prevent the loss of security event messages.

Security events on the system result in two forms of output: alarms and audits. Alarms are one-time notifications of security events that are sent to all terminals enabled as security operators. Audits are the recorded history of security events that can be retrieved and analyzed at a future date.

> NOTE
> All security events are propagated as both alarm messages at security operator terminals and as system audits written to the security audit log file. Digital intends to remove this restriction in a future version of the VMS operating system and allow you to control whether classes of security events generate alarms, audits, or both.

All security audit messages are recorded in the security audit log file. Use the Audit Analysis utility to review the security audit messages written to the log file.

Figure 8–1 illustrates the relationship among the VMS system security auditing subsystem components and shows the flow of security events as they are formatted into alarm messages and entries in the system security audit log file.

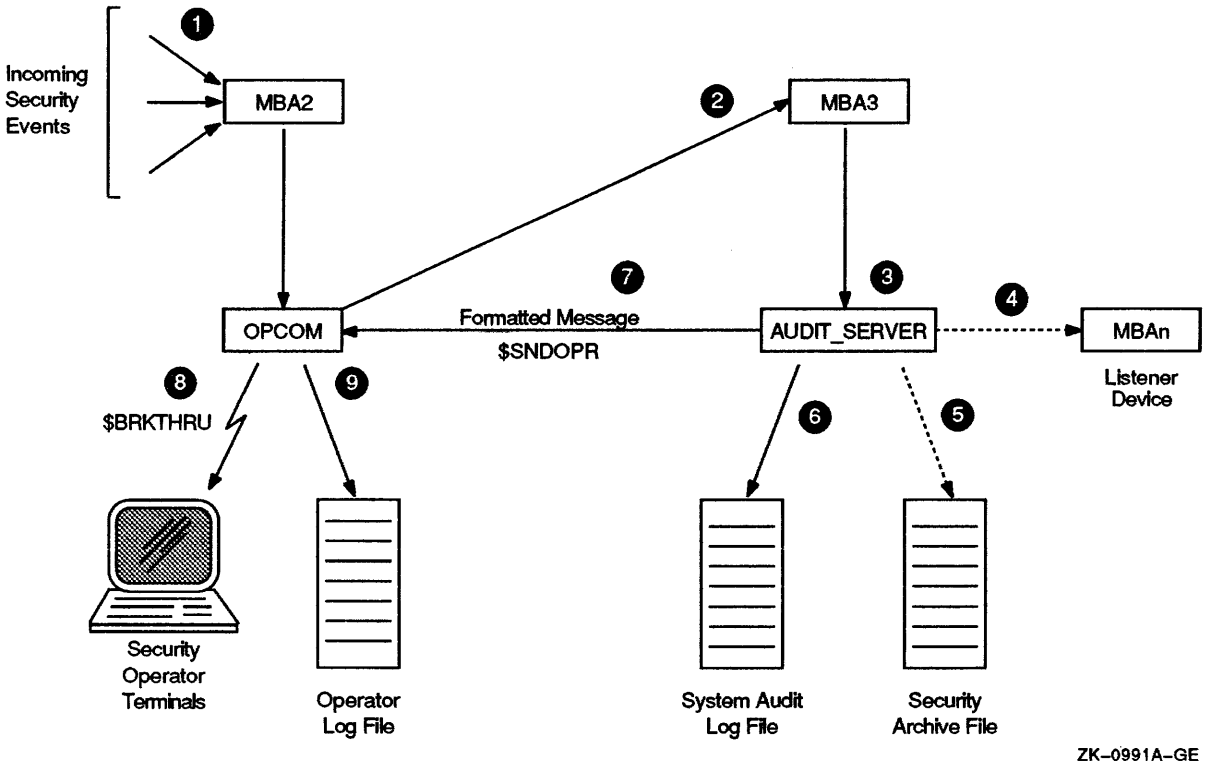


Figure 8–1 Auditing VMS System Security

The callout numbers in Figure 8–1 label the path that security events take through the VMS operating system.

Notes on Figure 8–1:

❶ A security event is generated (based on the event classes enabled on the system) and written to the operator communication manager (OPCOM) mailbox (MBA2) as a binary message.

❷ OPCOM reads and reformats the message and writes the message to the audit server process (AUDIT_SERVER) mailbox (MBA3).

❸ AUDIT_SERVER reads the message.

❹ AUDIT_SERVER writes a copy of the binary message to the listener device, if enabled.

❺ AUDIT_SERVER writes a copy of the binary message to the security archive file, if enabled. The security archive file can be located on a remote system.

❻ AUDIT_SERVER writes a copy of the binary message to the system security audit log file. This is called a **system audit**.

❼ AUDIT_SERVER formats the binary message into ASCII text and sends the message to OPCOM (using the $SNDOPR system service).

❽ Using the $BRKTHRU system service, OPCOM displays the message at all terminals enabled as security operators. This is called a **system alarm**.

❾ In addition to sending alarm messages to OPCOM, AUDIT_SERVER also sends error and informational messages to OPCOM. OPCOM then copies these messages to the operator log file (OPERATOR.LOG).

Thus, when an audited activity occurs,

- A security event is generated.
- OPCOM is informed.
- OPCOM sends a message to the AUDIT_SERVER process.
- The AUDIT_SERVER records the message, most importantly in its log file.
- The AUDIT_SERVER reformats the message and sends it back to OPCOM.
- OPCOM sends an alarm message to the operator terminals enabled to receive security messages.
- The AUDIT_SERVER writes security auditing error and informational messages to OPCOM. OPCOM places these messages in the operator log file (OPERATOR.LOG).

To enable or disable detection of the classes listed in Table 8–13, use the SET AUDIT command shown in Table 8–14. This requires SECURITY privilege.

To list the classes and types of audit currently enabled, use the SHOW AUDIT command shown in Table 8–14.

```
%%%%%%%%%%  OPCOM 11-APR-1989 09:32::53.39  %%%%%%%%%%%
Message from user AUDIT$SERVER on TIDE
Security alarm (SECURITY) and security audit (SECURITY) on TIDE,
system id: 167772163   / System UAF record modification
Event time:              11-APR-1989 09:32:53.38
PID:                     20800513
Username:                JONES
Image name:              $11$DUA53:[SYS0.SYSCOMMON.][SYSEXE]AUTHORIZE.EXE
Object name:             SYS$COMMON:[SYSEXE]SYSUAF.DAT;1
Object type:             file
User record modified:    SMITH
Fields modified:         PRIVILEGES
```

Example 8–7 Security Alarm Message on Console Terminal

Table 8–14 Defining and Listing Audit Classes

Command	Comments

`$ SET AUDIT/ALARM -`
`_$ /ENABLE=(class[=keyword])`
 	
`$ SET AUDIT/ALARM -`
`_$ /ENABLE=AUTHORIZATION`
 	
`$ SET AUDIT/ALARM -`
`_$ /ENABLE=(BREAKIN=DIALUP)`
	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=(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.

8.7.2 Audit Analysis

The data collected by the AUDIT_SERVER can be analyzed with the ANALYZE/AUDIT command. The *VMS Audit Analysis Utility Manual* provides complete descriptions for all selection qualifiers. Unless otherwise directed, the analysis examines the default security audit log file, SYS$MANAGER:SECURITY_AUDIT.AUDIT$JOURNAL.

- The following command extracts all security audit records in a brief, one-line format (default), and writes the output to SECURITY.LIS:

  ```
  $ANALYZE/AUDIT/OUTPUT=SECURITY.LIS
  ```

- This command extracts all security audit records generated by the user MACARTHUR since January 1, 1989, and displays them at the terminal:

  ```
  $ANALYZE/AUDIT/SELECT=USERNAME=MACARTHUR/SINCE=1-JAN-1989
  ```

- The next command selects records generated by break-in attempts and file access that succeeds by using SYSPRV or BYPASS privilege.

  ```
  $ANALYZE/AUDIT -
  _$ /EVENT_TYPE=(BREAKIN,FILE_ACCESS=(SYSPRV,BYPASS))
  ```

- At any time during an audit analysis listing, you can interrupt the report being displayed and enter interactive command mode by typing CTRL/C. At the COMMAND> prompt, you can EXIT, CONTINUE, or enter commands to generate a new report.
- The Audit Analysis utility provides a HELP facility that contains information about all interactive mode commands.

For more information on these and other software security issues, read the appropriate chapters in the *Guide to VMS System Security*.

8.8 Laboratory Exercises

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.

8.9 Solutions to Laboratory Exercises

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. 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 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 cannot 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 solution required.
8. No solution required.

8.10 System Problems

VMS systems are built to be reliable, maintainable, and dependable. Many features were designed and built into each system to help prevent frequent system failures, and, for certain types of failures, to continue system operation. These same features also make VMS systems easier to repair when there is a failure.

However, problems can still occur on a VMS system. Problems fall into two categories:

1. Those caused by software, such as:
   - Operator error (incorrect input)
   - Programming error (infinite loops, wrong logic)
2. Those caused by hardware, such as:
   - Head crash (a type of disk drive failure)
   - Memory failure
   - Misalignment of disk or tape drives

Problems can result in:

- Corruption of data
  - From incorrect input or program error
  - From intermittent failures of hardware, such as a bad data line
- Loss of data
  - If the heads crash on a disk, all data on that disk becomes unreadable. Changes made since the last backup are lost.
  - If the system fails, recent changes stored in memory may not be written to disk. These changes could be corrupted or unrecoverable.
- Loss of system resources
  - Head crashes make the data on the volume inaccessible. The drive is also unavailable for loading other volumes.
  - A system that has failed cannot do work until it has been rebooted.
- Loss of computation
  - If a system fails during the run of a program, it must be run again from the beginning (unless checkpointing is enabled).

8.10.1 Software Problems

If the software causes a problem, the system manager must take action to:

- Retrain the users in proper software usage.
- Correct errors in programs causing the problem.

If the users are operating the software properly (for VMS system software, check documentation) and the same problem occurs each time they perfom a specific sequence of actions, the program may have a problem. For reproducible problems with VMS system software, document the problem and submit a Software Performance Report (SPR) to Digital Software Engineering.

Unless a problem is reproducible and well-documented in the SPR, it is difficult for Software Engineering to solve it. The *Guide to VMS System Security* contains an SPR form and details the information to submit with an SPR. The following rules are extracted from that guide.

- Include listings that reveal the state of the entire system. A problem may result from the interaction of many system components. A listing that shows the incorrect output is not sufficient.
- Explain how to reproduce the problem by using as simple a method as possible. Eliminate all irrelevant steps.
- Provide as much information as possible in machine-readable form. The media will be returned to you.
- Describe aspects of the system environment not apparent from listings.
- Include any clues and analyses of your own that may assist the software engineer who studies your problem.

NOTE
Customers who have purchased telephone support from Software Product Services, or whose software warranty is still in force, can telephone their problems to a Digital Customer Support Center (CSC). Employees at such a center will tell the customer if the problem has already been reported and if there is a way around it. If the customer is the first to report such a problem, the CSC will ask the customer to submit an SPR.

8.10.2 Hardware Problems

Hardware-related problems, such as a head crash, are obvious. Others, such as memory errors, are intermittent. You can detect and correct obvious problems quickly if you check the system periodically. For example, if the ERROR light on the front panel of the CPU, the OFF-LINE light on a disk drive, or the ALARM light on a printer light up, you should check the device immediately for a problem. You can detect more severe problems, such as a head crash on a disk, by the sight and smell of smoke, or by unusual noises. Become familiar with the normal appearance of your machine and peripherals to make it easier to detect problems.

The manager uses several tools to collect information when less obvious or intermittent problems occur. Some tools can also work in a preventive manner to detect problems that have not yet occurred but may occur soon. These tools include:

- The error logging facility
- The system failure dump facility
- Hardware diagnostics
- The operator's log file
- The UETP

8.10.2.1 The Error Logging Facility

The VMS system monitors all activity on disk and tape drives, and uses a system process, ERRFMT, to record certain activities in the system **error log file**, SYS$ERRORLOG:ERRLOG.SYS. The activities it records include:

- Device errors
- Memory errors
- Volume mounts and dismounts
- System startups

By default, the ERRFMT process logs errors for all disk and tape devices. You can use the DCL command SET DEVICE/ERROR_LOGGING to specify the devices on which to enable and disable error logging, as shown in Table 8–15. You can display the current status of any device to see if error logging is enabled with the SHOW DEVICE/FULL command.

Table 8–15 Enabling and Disabling Error Logging

Operation	Command Format (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:`

To examine the contents of the error log file, use the Errorlog Report Formatter (ERF).

The ERF can generate several types of reports using qualifiers with the command ANALYZE/ERROR, including:

- A full report of selected entries (/FULL. This is the default.
- A brief report of selected entries /BRIEF.
- A summary report of selected entries /SUMMARY.

The information included in the reports depends on what type of report you request, and what entries you select. Select entries using qualifiers for time intervals, device names, device classes, and entry types (see Table 8–16).

NOTE
If you include the /OUTPUT=file-name qualifier with the ANALYZE/ERROR command, it writes the output to the specified file. You can use a text editor, such as EDT, to examine the file when it is complete.

The following command generates a report on device errors and volume changes. Example 8–8 contains the final portion of this report.

```
$ ANALYZE/ERROR_LOG/INCLUDE=(DEVICE_ERRORS,VOLUME_CHANGES) -
_$ /SINCE=09-JUN-1988/OUTPUT=DEV_VOL.DAT
```

Table 8–16 Selecting Entries for an Error Log Report

Qualifier	Comment
/BEFORE /SINCE /ENTRY	Specify 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	Include or exclude 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.).

```
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 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 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
```

Example 8–8 Portion of Device Error and Volume Changes Report Generated by ANALYZE/ERROR

```
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
```

Example 8–8 (Con't) Portion of Device Error and Volume Changes Report Generated by ANALYZE/ERROR

Examine the log on a regular basis to determine whether errors are increasing for any devices. The system can recover most errors. The ERRFMT process records recoverable errors as **soft errors**. If the system cannot recover the error, the ERRFMT process records it as a **hard error**. A sudden increase in the number of hard errors is a warning that a more serious problem may soon occur. Note that the reports are primarily intended to assist Digital Field Service personnel, so you are not expected to understand their details, but you can keep track of whether hard errors increase or not.

As you look at the log, keep in mind that not all error log entries actually represent device or volume failure. For example, a user's attempt to write to a write-locked device is recorded as a device error in the error log file.

Keep all logs in a hard-copy form or on a disk until the next PM is scheduled, so the Field Service representative can look at the history since the last PM. It is helpful to keep some logs longer, but once a problem has been solved, the old logs are not as useful.

8.10.2.2 The System Failure Dump Facility

System shutdown occurs for two reasons:

1. An operator or manager requested the shutdown
2. The system detects an unrecoverable error

Regardless of the reason for the shutdown, the system saves a copy of the contents of memory in a system file named SYS$SYSTEM:SYSDUMP.DMP.

You can examine the contents of this file with the System Dump Analyzer (SDA) utility, invoked by the ANALYZE/CRASH command described in Table 8–17.

Table 8–17 Analyzing a System Dump

Steps in Running SDA Utility	Comments

`$ ANALYZE/CRASH_DUMP -`
`_$ SYS$SYSTEM:SYSDUMP`
	Requires:
	— Read access to dump file
	— Read access to system symbol table
	— Value of system parameter VIRTUALPAGECNT at least 4000 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 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`

Managers should create and collect these reports for each system failure, as information in them can be useful to a system programmer in determining the cause of the failure. The dump file is overwritten each time a failure occurs, so the manager may not be able to create reports on each failure unless the old dump file is saved as part of the startup procedure. Reports are also created as part of the startup procedure, as shown in Example 8–9.

```
$ !
$ ! Copy Dump file and create listing if system just failed
$ !  This procedure to be invoked from SYSTARTUP_V5.COM
$ !
$ 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]SAVEDUMP.LIS
```

Example 8–9 Copying the Dump File at Startup and Creating Reports

```
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 8–10 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 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 8–11 Second 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
```

Example 8–12 Third Page of Report Produced by SDA Command SHOW CRASH

Use the SDA command SHOW SUMMARY/IMAGE to:

- Display a list of users working on the system when it failed
- List images that were running at the time of failure

```
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
```

Example 8–13 Fourth Page of Report Produced by SDA Command SHOW SUMMARY/IMAGE

For more information on analyzing a system dump, read the *VMS System Dump Analyzer Utility Manual*, which contains an example of each type of report that can be generated.

8.10.2.3 Hardware Diagnostics

Digital Field Service representatives use **hardware diagnostics** to isolate and diagnose hardware problems.

A Field Service Engineer can run diagnostics on-site, or from a Digital Diagnosis Center (DDC) at a remote location.

Field Service can execute diagnostics when the VMS system is running (on-line diagnostics) or when it is not running (standalone diagnostics).

On-Line Diagnostics

If the VMS system is running, the Field Service engineer executes diagnostics only on parts of the system that can be allocated without affecting system execution, such as tape drives, printers, and nonsystem disk drives. A Field Service engineer typically loads special media in a device before executing diagnostics on it. Be sure to remove sensitive volumes from drives before the engineer begins to work.

> NOTE
> On-line diagnostics are not supplied as part of every system. The Field Service engineer brings them and restores them to the SYS$MAINTENANCE directory. The FIELD account should be available to execute these diagnostics. Also, do not delete the diagnostic programs unless you need the disk space. Otherwise, Field Service must restore them each time. If you know how to run the diagnostics (ask your Field Service engineer for tips) you might be able to provide some help by running some programs before the engineer arrives (depending upon your site, your assigned engineer, and the scope of the problem.)

Standalone Diagnostics

If Field Service must diagnose components that the VMS system needs to run, the engineer must run the diagnostics in standalone mode. Shut down the system and remove the system disk and any other sensitive volumes. Then provide the engineer with access to the console terminal and console device.

NOTE
The VAX 11/730 Customer Runnable Diagnostics (CRD) is run in standalone mode. Follow the instructions supplied with the CRD and contact your local Field Service office if errors exist in the printed output.

An engineer who isolates a problem using software diagnostics may run more specific tests on an individual device. If you can isolate the device from the running system, then you do not have to shut the system down. Finally, the engineer will fix the problem.

Remote Diagnosis

Field Service can run on-line or standalone diagnostics from a remote location (a DDC) if the manager cooperates and the system has the optional remote diagnosis port. Also, the customer maintenance agreement must entitle the system to receive remote diagnostics. The key switch on the front panel of the CPU controls remote access. If the key is not set to REMOTE, the DDC cannot gain access to your system. If you have the remote diagnostic option, the DDC can determine the problem and send an engineer with a replacement part or a good idea of what specific tests to run. Every customer supported by the DDC receives a detailed set of instructions on how to set up and run a remote diagnostic session. Read and follow these instructions each time you place a call to the DDC for a system.

8.10.2.4 User and Operator Comments

Verbal, written, or mailed comments from users or operators can be very useful in tracking the cause of a problem. Encourage all users of the system to report any unusual occurrences. Be sure to follow up comments with training sessions or further investigations using the tools previously discussed. User comments are usually your first notification that a problem exists.

Log all comments in an operator log book along with other information such as:

- Backups done
- Cleaning done
- Shift change information
- Warnings and notes for other operators
- Work left from the previous shift that must be completed
- Unusual events (power out, unauthorized personnel access, etc.)
- Specific hardware or software problems

Some sites log this information in the appropriate section of the *Site Management Guide.*

8.10.2.5 Operator Log File

The operator log file (SYS$MANAGER:OPERATOR.LOG) contains:

- All REQUEST and REPLY communication between users and operators
- All communication between the operating system and operator (such as mount and dismount messages)
- Messages indicating when terminals are enabled or disabled as operator terminals
- The time of day, entered every 60 minutes in the file
- Off-line messages when a device has been taken off-line (for example, the line printer runs out of paper and goes off-line)

To close the log and open a new one, enter the REPLY/LOG command. After the system creates a new log, you can examine the old one.

The operator log is a text file that you can display by using the PRINT or TYPE commands. A comparison between the errors logged in ERRORLOG.OLD and the messages logged in OPERATOR.LOG can be helpful in investigating some types of problems. For example, an attempted mount of a write-protected drive is logged as an error in ERRORLOG.OLD. If you look at the corresponding section of the operator log, you will see an error message output to the user from the system and conclude that there is one less real error for the device than OPERATOR.LOG records.

8.10.2.6 UETP

Run the User Environment Test Package (UETP) after system installation to demonstrate that VMS hardware and software are working correctly. You can also run it any time after that.

The UETP is a collection of automated reusable test modules that exercise all the peripherals and major system software components in a simulated customer environment. The UETP is not a replacement for diagnostics or system exercisers. However, since it tests how the hardware and software work together, it can force and find integration and interface errors that diagnostics and system exercisers cannot.

The UETP does not identify the cause of a problem. It simply displays an error message. Module 7 discusses the UETP in more detail.

> NOTE
> The UETP must be run from the SYSTEST account, one of four default accounts set up on the VMS distribution kit. Only the password can be changed in this account. If you change any privileges or quotas, system errors may result during the UETP run.

8.10.2.7 Additional References

- For more information on controlling error logging, read the command description for SET DEVICE/ERROR_LOGGING in the *VMS DCL Dictionary*.
- For more information on analyzing the error log, consult the *VMS Error Log Utility Manual*.

8.11 Laboratory Exercises

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 course administrator has made one available, take the time to look at it during this exercise. Take note of the types of information requested, so if a problem occurs, you will 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? Why?

7. List the test files used by the UETP. Use a system logical name in the command.

8.12 Solutions to Laboratory Exercises

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`

8.13 Optional Laboratory Exercises

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.

8.14 Solutions to Optional Laboratory Exercises

1. See Example 8–9.
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

9.1 Introduction

The VMS operating system provides many tools for monitoring system behavior. With these tools, you can examine the activity of VMS processes and system resource use. If user demands on system resources are too great, system performance will be unsatisfactory.

The primary factor limiting VMS system performance is hardware resources, especially physical memory and disk storage. Hardware resources that are inadequate for the workload generally perform poorly, regardless of any other efforts to improve performance.

When the VMS operating system is installed and started for the first time, the system determines the amount of physical memory and disk storage available. It 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 system performance.

9.2 Objectives

To know how your system is performing, you must:

- Monitor the system
- Modify system parameters to reflect a change in system resources

9.3 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*
8. *Guide to VMS Performance Management*

9.4 Monitoring System Activity

The VMS operating system provides many methods for examining system activity. Table 9-1 lists some commands and utilities you can use for this purpose. (Not all of these commands are discussed in this module, some are covered elsewhere in the course.)

Table 9–1 System, Process, and Device Monitoring

Information Displayed	Command or Utility
General System Information	
Overview of system processes	`$ 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`
Overview of system memory resources	`$ SHOW MEMORY`
Demands on system resources	`$ MONITOR`
Error counts for the CPU, memory, and physical devices	`$ SHOW ERROR`
Cluster activity and performance	`$ SHOW CLUSTER`
Specific Information	
Interactive users, terminal codes, 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`

Table 9–1 (Cont.) System, Process, and Device Monitoring

Information Displayed	**Command or Utility**
Information about devices and volumes	`$ SHOW DEVICE device`
Consumption of resources by processes	`$ ACCOUNTING`

9.4.1 Monitoring Active Processes

The SHOW SYSTEM command lists all processes on the system. You can use the process name or the process identification (PID) number shown in the display in other DCL commands to specify a particular process. Note that to refer to a process by process name, your **own** process and **that** process must be in the **same** UIC group. For a closer examination of individual processes, you can use the command SHOW PROCESS/CONTINUOUS/ID covered in this module, or SHOW PROCESS/ALL/ID.

Example 9-1 shows the output from a SHOW SYSTEM command that was entered on a node in a VAXcluster system. The columns are labeled "State" (process state) and "Pri" (process current priority). The numbers in Example 9-1 correspond to the numbers in Table 9-2.

Processes in the COM (computable) state are ready to use the processor. Processes in one of the other states are usually waiting for information or a system resource. Since processes in these states are all waiting for something, they are called **wait states.**

The VMS system adjusts the state of a process whenever the process's readiness to use the processor changes. The state may change from the computable state (COM) to a wait state (such as LEF), or from a wait state to the computable state. The VMS system also changes the state if it moves the process to disk to make room in memory. This move is called **outswapping.** An outswapped process may have the same state as before, but the VMS system adds the letter O to the end of the state (for example, LEF becomes LEFO).

If the VMS system outswaps many processes, or holds them in certain wait states (see Table 9-2) for more than several minutes, you may have an overloaded system. Performance is typically degraded when the system is overloaded. To solve this problem, consider limiting the amount of memory each process uses by adjusting working set limits in the user UAF records. Also, consider limiting the number of batch and interactive processes on the system. To limit batch processes, use the /JOB_LIMIT qualifier to the INITIALIZE/QUEUE command. To limit the number of interactive processes, use the SET LOGINS/INTERACTIVE command.

For more suggestions on how to modify your system when it is overloaded, refer to the *Guide to VMS Performance Management*.

```
  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
❷ 21200081 SWAPPER          HIB     16         0   0 00:01:03.47          0      0
           [GROUP1,SYSTEM]
❷ 21200086 ERRFMT           HIB      9      3221   0 00:00:38.08         70    111
           [1,6]
❷ 21200089 OPCOM            LEF      8      1393   0 00:00:19.59       4844     81
           [GROUP1,SYSTEM]
❷ 2120008A JOB_CONTROL      HIB      8     81129   0 00:24:52.39        200    342
           [GROUP1,SYSTEM]
❷ 2120008C NETACP           HIB     10     35622   0 00:38:08.93       5965   5640
           [GROUP1,SYSTEM]
❷ 2120008E REMACP           HIB      9       467   0 00:00:02.05         78     65
           [GROUP1,OPERATOR]
❷ 2120008F SYMBIONT_0001    HIB      6      7223   0 00:05:23.34      29327     69
           [GROUP1,SYSTEM]
❷ 21200090 SYMBIONT_0002    HIB      4        16   0 00:00:13.32        193     70
           [GROUP1,SYSTEM]
❷ 2120009B VAXsim_Monitor   HIB      8       642   0 00:00:11.49        361    255
           [1,6]
❷ 2120009E VPA_DC           HIB     15      2870   0 00:10:15.45        234    323
           [GROUP1,SYSTEM]
❶ 21200524 BATCH_1036       COM      2   7137740   1 00:01:43.25        381    346 B
           [GROUP1,OPERATOR]
❹ 212006AC MATTHEWS         LEF      7      7973   0 00:01:07.32       4387   1500
           [GROUP11,MATTHEWS]
❸ 21200731 MATTHEWS_1       CUR      4       177   0 00:00:02.28        433    269 S
           [GROUP11,MATTHEWS]
❹ 21200635 SERVER_0635      LEF      5       369   0 00:00:09.17       1732    267 N
           [DECNET]
❷ 212004D3 NOTES$0007_0*    HIB      6      1537   0 00:00:19.87       7787     67 N
           [GROUP111,NOTES$SERVER]
```

Example 9–1 Output from the SHOW SYSTEM/FULL Command

Table 9–2 System States

Name of State and Abbreviation	OK	Probably OK	Possible Problem	Definition of State
❶ Computable COM/COMO		X		Available to use the processor
[1]Common Event Flag CEF/CEFO		X		Waiting for something to happen in the image
[1]Collided Page COLP			X	Waiting for the system
❸ Current CUR	X			Executing on the processor
[1]Free Page Wait FPG			X	Waiting for physical memory
❹ Local Event Flag LEF/LEFO		X		Waiting for something to happen in the image
❷ Hibernate HIB/HIBO		X		Intentionally doing no work[2](inactive)
[1]Mutex/Misc RWxxx MWAIT			X	Waiting for a resource
[1]Suspended SUSP/SUSPO		X		Intentionally doing no work[3](inactive)
[1]Page Fault Wait PFW			X	Waiting for the system

[1]Example 9–1 does not include an example of this state.
[2]System processes normally use this state when they have no task to perform. They put themselves in this state. They are available when needed because they have control over their state. However, they are not competing with computable processes for the processor (until they need it).
[3]This state should be used for short periods only. One process puts another process in this state. A process in this state cannot change its state; another process must intervene to make a change.

NOTE
The occurrence of outswapping does not necessarily indicate a serious system problem. However, if it takes a long time for an outswapped process to become computable again, there **could** be a problem.

Processes that are computable must share the processor. The VMS system schedules computable processes to use the processor according to their priority. The highest priority process gets first access to the processor. This process is called the **current** process.

NOTE
Your own process is always the current (CUR) process when you look at the display from the SHOW SYSTEM command because it is using the CPU to collect the information for the display.

Assign priorities to interactive user processes, typically, with a base priority between 0 and 15 in user UAF records. The VMS system automatically adjusts the priority of this kind of process upward and downward between its base priority and 15 to ensure that all processes share access to the processor.

If you assign one user a higher base priority than another user, the first user can monopolize the processor. This can degrade the system response time for all other processes. Therefore, you should give all interactive users the same base priority (the recommended value is 4).

NOTE
The SHOW SYSTEM command lists the current priority of a process (with the adjustment) rather than the base priority.

Some system and user processes must have base priorities between 16 and 31 because they have a critical job to perform. These processes do not share the processor, and the VMS system does not adjust their priorities. When the VMS system gives the processor to a process with a priority above 15, the processor executes the image in that process until it is finished or until a higher priority process becomes computable. Since only one process can use the processor at a time, all other processes must wait for the job to be done. Therefore, do not assign a priority above 15 to a user process unless absolutely necessary.

For more information about how the VMS system adjusts the priorities of interactive processes, refer to the *Guide to VMS Performance Management*.

9.4.2 Monitoring System Processes

Independent processes perform some of the functions of the VMS operating system. The VMS system creates most of these processes at system initialization. Table 9-3 lists these **system processes**, the states in which they often appear, and their purposes.

Table 9–3 System Processes

Process Name	Usual States	Purpose
SWAPPER	HIB,COM	Transfers processes between physical memory and disk
ERRFMT	HIB,COM	Records device errors in a file on the system disk
JOB_CONTROL	HIB,COM	Manages print jobs, batch jobs, interactive processes, and process accounting
OPCOM	HIB,COM	Transmits and records user requests and operator replies
SYMBIONT_nnnn	HIB,COM	Prints files to on-line printers
xxxACP	HIB,COM	Manages data transmission to and from tapes, interconnected computers, and some disks

When any system process is absent or unable, it affects the behavior of the system. Table 9-4 lists the system processes and the effects of deleting such a process.

Table 9–4 Results of System Process Deletion

Process Name	Result of Process Deletion
SWAPPER	Cannot be deleted
ERRFMT	No device errors logged
JOB_CONTROL	User cannot log in and queues do not function
OPCOM	No user or operator communication
SYMBIONT_nnnn	Printers may not print
xxxACP	System slows down, possibly stops entirely

To restart the ERRFMT, OPCOM, or JOB_CONTROL process, log in as SYSTEM and run the command procedure STARTUP.COM, entering the name of the process-image (e.g., ERRFMT, OPCOM) as a parameter:

```
$ @SYS$SYSTEM:STARTUP process-image
```

9.4.3 Obtaining Information About a Device

To display information about the devices on your system, use the SHOW DEVICES command. Table 9–5 shows how to use this command to display various kinds of information.

Table 9–5 SHOW DEVICES Command

Information Displayed	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.[1]	`$ SHOW DEVICES/FILES -`
	 `_$ device-name`

[1]You need the SYSPRV and WORLD privileges to display complete information with this qualifier.

Examples 9-2 and 9-3 illustrate the SHOW DEVICE command using selected qualifiers.

```
$ 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 9–2 SHOW DEVICES/MOUNTED 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.
```

Example 9–3 SHOW DEVICES/FULL Output

9.4.4 Monitoring Memory Resources

System performance greatly depends on the amount of physical memory contained in the system. For the VMS system to manage process and system virtual memory, it must divide available physical memory among different functions. The SHOW MEMORY command displays information about the physical memory of the system and the disk files used in the management of virtual memory. Example 9-4 shows a sample of the SHOW MEMORY output. When the values of the numbered items in the "Free" column become too small, system problems occur. Table 9-6 lists some of these problems. To solve them may require that you add more physical memory, or that you "tune" the system by adjusting various SYSGEN parameters. A temporary measure is to decrease working set sizes for all processes. You might also want to decrease the number of processes allowed in memory at the same time, but this may increase the number of outswapped processes unless you also reduce the total number of processes allowed on the system at a time.

```
$ 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.
```

Example 9–4 SHOW MEMORY Output

Table 9–6 Effect of Memory Sizes on Performance

Item	Description	Problem
Free Physical Memory	The free memory is the size of the free page list (the number of pages available for VMS to give to processes that need memory space)	If less than a few hundred blocks, swapping will occur.
Free Process Entry Slots	The number of additional processes that VMS 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 VMS can inswap	If zero, then swapping may occur even if there are free pages available in memory. If too big, it can waste physical memory space (1-2 blocks per free slot).
Fixed-Size Pool Areas (Packets)	The nonpaged memory in system space mainly used for I/O	If any item is zero, the system tries to increase it. Enter the SHOW MEMORY/FULL command for more detailed information.
Free Paged Dynamic Memory and Free Nonpaged Dynamic Memory	The amount of dynamic memory left for process use	If system response deteriorates when too small.
Free Swap File Pages [1]	The number of pages VMS has on disk for swapping	If too small (or nonexistent), the VMS system uses page file instead (can significantly reduce performance).

[1]For more information on the page and swap files, enter the SHOW MEMORY/FULL/FILES command for each file. You need the WORLD and SYSPRV privileges to display complete information.

Table 9–6 (Cont.) Effect of Memory Sizes on Performance

Item	Description	Problem
Free Page File Pages [1]	The number of pages the VMS system has on disk for paging.	If too small, will cause process to wait in MWAIT state. The VMS system prints a message on the console terminal when the page file reaches 60% and 90% full.

[1]For more information on the page and swap files, enter the SHOW MEMORY/FULL/FILES command for each file. You need the WORLD and SYSPRV privileges to display complete information.

9.4.5 Monitoring Print and Batch Queues

To examine the queues on the system, enter the following commands:

```
$ SHOW QUEUE/DEVICES/ALL

$ SHOW QUEUE/BATCH/ALL
```

Module 3 details these commands further.

9.4.6 The Monitor Utility

The Monitor utility displays information about system resource usage. A limited amount of this information is available from the SHOW MEMORY and SHOW SYSTEM commands.

Run MONITOR regularly while the system is running well. This provides a baseline of data. If the system encounters a performance problem at a later time, you can compare current MONITOR data with the baseline data. *VMS Monitor Utility Manual* explains how to use the MONITOR command, but not how to interpret the output. The *Guide to VMS Performance Management* provides background information necessary to interpret MONITOR displays.

You may run the Monitor utility in one of two ways. If you enter the MONITOR command followed by a carriage return, you will see the MONITOR prompt, as shown in Example 9-5.

```
$ MONITOR
MONITOR>
```

Example 9–5 Invoking the Monitor Utility

You can also enter MONITOR as a command to display one or more classes of information. The utility shows each class of information for a few seconds and updates each display at regular intervals. To execute the utility using this method, type:

```
$ MONITOR class-name(s)[/qualifiers]
```

At the MONITOR prompt, you may enter MONITOR commands to:

- Display a class of information
- Set default classes
- List defaults
- Execute command procedures
- Exit the utility
- Obtain help

While displaying a class of information, you can enter the CTRL/C key sequence to return to the MONITOR> prompt and request another display. To exit the utility and return to the DCL prompt, enter the EXIT command at the prompt or enter the CTRL/Z key sequence while a display is running.

Table 9-7 lists the class names and gives a brief description of each. After monitoring the system with this method, enter the CTRL/Y or the CTRL/Z key sequence to exit the display and return to the DCL prompt. To return to the MONITOR> prompt, enter the CTRL/C key sequence instead.

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 software
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 the nonpaged dynamic memory	POOL
Statistics on all processes	PROCESSES
System communications services statistics	SCS
Number of processes in each of the scheduler states	STATES
Brief display of general status of system (includes information displayed in other classes)	SYSTEM

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 9-6 shows the MONITOR screen display of the PAGE class. Note the four columns of numbers. These columns stand for the current, average, minimum, and maximum values. When MONITOR is running, it updates the values in the display dynamically to reflect the current state of the system.

After several weeks or months of collecting monitoring information, you will see the average, minimum, and maximum values for an item stabilize. Consider this range for the item to be normal for your system. Values on one system are often different from values on another, but you should be able to determine your normal values by regular monitoring. Later, if the workload or system capability changes, you should note a corresponding change in the values of some items to establish a new normal range for them.

```
$ 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
```

Example 9–6 MONITOR Screen Display of the PAGE Class

Note that each value is updated every few seconds.

You will also be interested in the PROCESSES class. This class display looks similar to the SHOW SYSTEM display, but there are four class qualifiers that change the display to a bar chart, as shown in Table 9-8.

Table 9–8 MONITOR PROCESSES Class Qualifiers

Qualifier Description	Qualifier
Top buffered I/O users	/TOPBIO
Top CPU users	/TOPCPU
Top direct I/O users	/TOPDIO
Top page fault users	/TOPFAULT

For example, to see what processes are using the most CPU time, issue the DCL command:

```
$ MONITOR PROCESSES/TOPCPU
```

Example 9-7 gives you an idea what appears on the PROCESSES/TOPCPU class screen. The utility updates a real screen of this type every few seconds by default.

```
$ 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   *
                                       |         |         |         |          |
                                       |         |         |         |          |
                                       |         |         |         |          |
                                       |         |         |         |          |
                                       |         |         |         |          |
                                       + - - - - + - - - - + - - - - + - - - - -+
```

Example 9–7 MONITOR PROCESSES/TOPCPU Screen Display

Another screen that the Monitor utility can display is the SYSTEM screen. This screen combines several items from other screens to show you a more general view of the system status, as shown in Example 9-8.

```
Node: OTHER                 VAX/VMS Monitor Utility     30-APR-1988 17:18:41
Statistic: CURRENT             SYSTEM STATISTICS
                                                     Process States
          + CPU Busy (96)           -+         LEF:     5      LEFO:     0
          |************************  |         HIB:    19      HIBO:     0
CPU     0 +--------------------------+ 100     COM:     2      COMO:     0
          |**************            |         PFW:     0      Other:    1
          +--------------------------+         MWAIT:   0
          Cur Top: BATCH_1036 (56)                       Total: 27

          + Page Fault Rate (108)   -+         + Free List Size (76414)  -+
          ||*************************|         |**********************    | 89K
MEMORY  0 +--------------------------+ 100   0 +--------------------------+
          |**************************|         |*********                 | 2621
          +--------------------------+         + Modified List Size (917) +
          Cur Top: MARSH (106)

          + Direct I/O Rate (52)    -+         + Buffered I/O Rate (4)   -+
          |**********************    |         |                          |
I/O     0 +--------------------------+ 60    0 +--------------------------+ 150
          |*******************       |         |                          |
          +--------------------------+         +--------------------------+
          Cur Top: BATCH_1036 (47)             Cur Top: MARSH (4)
```

Example 9–8 MONITOR SYSTEM Screen Display

9.4.7 Information on a Specific Process

The SHOW PROCESS/CONTINUOUS command displays information about a particular process. The system updates this information every few seconds. To activate the utility, type:

```
$ SHOW PROCESS/CONTINUOUS [/ID=proc-id]  [proc-name]
```

Example 9-9 shows the output from this utility. To exit from the utility, type the letter E.

> **NOTE**
> You must have WORLD privilege to use this command to examine any process on the system whose UIC is not the same as yours.

```
$ SHOW PROCESS /CONTINUOUS /ID=7CC

                        Process MARSH ❶                  17:27:29

    State               LEF ❷              Working set             861
    Cur/base priority   9/4 ❸              Virtual pages          3895
    Current PC          7FFEE44C           CPU time        00:10:43.48 ❹
    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 ❺
```

Example 9–9 Output from SHOW PROCESS/CONTINUOUS

Notes on Example 9-9:

❶ Name of process

❷ Current state

❸ Current and base priority

❹ CPU time

❺ Name of image being run

9.4.8 Interactive Users

The DCL command SHOW USERS lists the interactive processes and their terminal codes. The form of this command is:

```
$ SHOW USERS
```

Example 9-10 shows the sample 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:

❶  KREUTZER      KREUTZER       ❷  210006A8     ❸  VTA178:       ❹  TTA3:

   PAGANINI      PAGANINI          2100033D        RTA1:

   PIATIGORSKY   PIATIGORSKY       21000544        VTA176:          LTA8:

   ROSE          ROSE              21000A6C        VTA26:           LTA12:
```

Example 9–10 Output from SHOW USERS Command

Notes on Example 9-10:

❶ KREUTZER User name

❷ 210006A8 PID for the process

❸ VTA178: Virtual terminal device code

❹ TTA3: Terminal device code

9.4.8.1 Additional References

- For more information on commands used to monitor the system as a whole, read the command descriptions for the SHOW SYSTEM, SHOW DEVICES, SHOW MEMORY, and SHOW QUEUE commands in the *VMS DCL Dictionary*
- For more information on the use of the Monitor utility, read the *VMS Monitor Utility Manual.*
- For more information on commands used to monitor processes on the system, read the command descriptions for SHOW PROCESS and SHOW USERS in the *VMS DCL Dictionary*.

9.5 Laboratory Exercises

1. Using a video terminal, run the Monitor utility to display the MONITOR> prompt. Enter a command to the utility at the prompt to display the PAGE class with an update interval of 10 seconds. Allow the utility to run for 5 to 10 minutes. Watch the display and notice how (if you are on an active system) the display changes.
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?

9.6 Solutions to Laboratory Exercises

1.
   ```
   $ MONITOR
   MONITOR> MONITOR PAGE
   ```
2. Enter the CTRL/C key sequence to return to the MONITOR> prompt.
   ```
   MONITOR> MONITOR SYSTEM
   ```
3. Enter the CTRL/C key sequence to return to the MONITOR> prompt.
   ```
   MONITOR> MONITOR IO
   ```

4.
   ```
   $ MONITOR
   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>  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.

9.6.1 Collecting Process Information with the Accounting Utility

The system accounting file SYS$MANAGER:ACCOUNTNG.DAT is used to collect process information with the Accounting utility. The contents of this file are used to:

- Record system activity
- Charge for system resources used
- Analyze relationship between system activity and performance
- Monitor system activity for security reasons

This file can be read by the VMS system Accounting utility to produce formatted reports. It can also be read by the user-written processing utility. Input/output can be controlled by the 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

The contents of the accounting record include:

- System resource usage
- Identity of resource user

Example 9-11 displays an accounting record.

```
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
```

Example 9–11 Accounting Record, Full Format

Recording is enabled by default (except image activity). The SET ACCOUNTING command controls which types of records are written to ACCOUNTNG.DAT by JOB_CONTROL:

- BATCH
- INTERACTIVE
- MESSAGE
- LOGIN_FAILURE
- PRINT
- PROCESS
- IMAGE

See Table 9-9 for information using the SET ACCOUNTING command.

Table 9–9 Recording Accounting Information

Operation	Command Format
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 file	`$ SET ACCOUNTING/NEW_FILE`

9.6.2 Using the Accounting Utility to Produce Reports

The Accounting utility reads the system accounting file and produces reports. Qualifiers to the DCL ACCOUNTING command control:

- Accounting records to analyze
- Details to disclose:
  - Full
  - Brief
  - Summary
- Order to display records

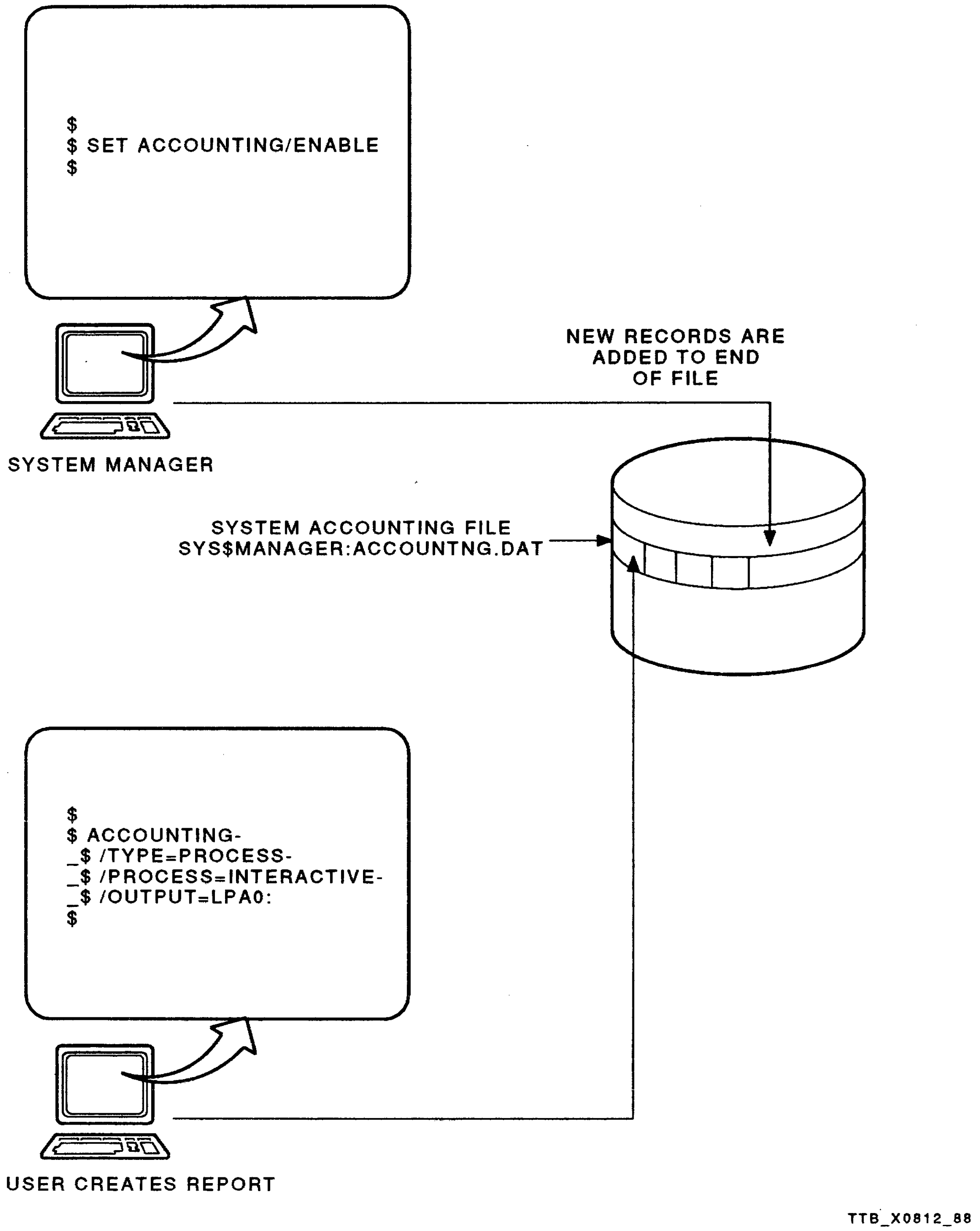
The full format accounting report (Example 9–11) is generated by:

```
$ ACCOUNTING /FULL /TYPE=PROCESS /PROCESS=INTERACTIVE
```

The brief format accounting report (Example 9–12) is generated by:

```
$ ACCOUNTING /SINCE=27-APR-1988:07:30 /BEFORE=27-APR-1988:08:00
```

See Figure 9-1 for an illustration of a system accounting file.


SYSTEM MANAGER
NEW RECORDS ARE ADDED TO END OF FILE
SYSTEM ACCOUNTING FILE SYS$MANAGER:ACCOUNTNG.DAT
USER CREATES REPORT
TTB_X0812_88

```
$
$ SET ACCOUNTING/ENABLE
$
```

```
$
$ ACCOUNTING-
_$ /TYPE=PROCESS-
_$ /PROCESS=INTERACTIVE-
_$ /OUTPUT=LPA0:
$
```

Figure 9–1 The System Accounting File

Example 9-12 displays a brief listing of accounting records.

```
     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
```

Example 9–12 Accounting Records, Brief Format

The summary format accounting report shown in Example 9–13 is generated by:

```
$ ACCOUNTING /SINCE=27_APR-1988:05:00 /BEFORE=27-APR-1988:10:00 -
_$ /TYPE=PROCESS /SUMMARY=(HOUR,USER) /REPORT=BUFFERED_IO
```

```
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
```

Example 9–13 Accounting Report, Summary Format

Table 9-10 lists selected qualifiers used to specify content of an accounting report.

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 lists selected qualifiers that affect output format of an accounting report.

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

Because the default accounting file can become quite large, you should rename default file periodically or create historical accounting files with partial accounting information.

Example 9-14 shows how you can specify one or more files with the ACCOUNTING command.

```
$ ACCOUNTING /TYPE=PRINT
$
$ ACCOUNTING /TYPE=PRINT SYS$MANAGER:ACCO_88_JUN_12.DAT
$
$ ACCOUNTING /TYPE=PRINT SYS$MANAGER:ACCO_88_JUN*.DAT
```

Example 9–14 Selecting Accounting Files

Use six basic steps to create an accounting report (as shown in Table 9-12).

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-1989:05:00 -
_$ /BEFORE=02-JUL-1989:10:00-
_$ /TYPE=PROCESS -
_$ /SUMMARY=(HOUR,USER) -
_$ /REPORT=BUFFERED_IO
```

The following is the contents of DCL command procedure BUFFSUM.COM:

```
/TYPE=PROCESS/SUMMARY=(HOUR,USER)/REPORT=BUFFERED_IO
```

Type the following to get the abbreviated command string incorporating the command procedure BUFFSUM.COM:

```
$ ACCOUNTING -
_$ /SINCE=02-JUL-1989:05:00 -
_$ /BEFORE=02-JUL-1989:10:00@BUFFSUM
```

9.7 Laboratory Exercises

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.

9.8 Solutions to Laboratory Exercises

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
```

9.9 Maintaining System Performance

The data structures and files the VMS system needs for its operation occupy physical memory and disk storage. System performance is best when the sizes of the VMS system data structures and files match the system resources.

As summarized in Module 5, the VMS system data structures are constructed each time the system is started up. The values of the **system parameters** determine how much physical memory these data structures use. To ensure that system parameter values reflect current hardware resources, the system automatically determines and records these hardware resources when the VMS system software is installed. After compiling a current list of the hardware, the system computes appropriate parameter values. These parameters become the default parameter values for the system (stored in SYS$SYSTEM:VAXVMSSYS.PAR). The system also creates a default list of images (stored in SYS$MANAGER:VMSIMAGES.DAT) to be installed at system startup based on the size of physical memory and other factors. If you want to install other images from the SYSTARTUP_V5.COM procedure, you may have to remove some of the system default images to make room for them in memory (as shown in Module 5).

Certain system files must reflect the system they are on. These are also customized when you install the system software to reflect the physical memory size and disk capacity of your system. These files are:

- Paging file
- Swapping file
- Dump file

Table 9-13 lists these files and their functions.

Table 9–13 System Files

File[1]	File Specification	Function
Paging file	SYS$SYSTEM:PAGEFILE.SYS	Used to create and manage virtual memory
Swapping file	SYS$SYSTEM:SWAPFILE.SYS[2]	Used to manage physical memory
Dump file	SYS$SYSTEM:SYSDUMP.DMP	Used to save a copy of physical memory when the system fails

[1]For more information on the paging and swaping files, enter the SHOW MEMORY/FILES/FULL command for each of them. (You need the WORLD and SYSPRV privileges to display complete information using this command.)
[2]If the storage capacity of the system disk is less than 25,000 blocks, the file specification is SYS$SYSTEM:SWAPFILE1/SYS.

When the physical resources of a system are increased to improve system performance, system parameter values and file sizes should also be recomputed. There are a number of ways to make the system recognize these new resources:

- Command procedure SYS$UPDATE:AUTOGEN.COM (this is the recommended method):

  Determines hardware resources, performs computations, changes system parameters, creates a list of image files to install, and calculates the sizes of the page, swap, and dump files. Autogen details are in the *Guide to Setting Up a VMS System*.

- Utility SYS$SYSTEM:SYSGEN.EXE:

  Changes system parameters, calculates the sizes of the page, swap, and dump files.

- Conversational startup utility SYSBOOT:

  Changes system parameters.

9.9.1 Reconfiguring the System with AUTOGEN

The AUTOGEN command procedure automatically runs at system installation to determine system hardware resources and record the appropriate system configuration established at startup. To run AUTOGEN:

```
@SYS$UPDATE:AUTOGEN [start-phase] [end-phase] [execution-mode]
```

You should run AUTOGEN again whenever the physical resources of the system change. For example, if you add or remove memory or disks, run AUTOGEN. It will modify system parameters, the size of system files, and the number of images installed to match the new configuration.

> NOTE
> The changes made by AUTOGEN do not take effect until the next time the system starts. For this reason, you can optionally have AUTOGEN shut down the system, or manually shut down the system and restart it automatically.

You can also use AUTOGEN when you want to specify a system parameter or file size. Because the value of one parameter can affect the preferred value for other parameters, AUTOGEN recalculates values and sizes whenever you manually change a single parameter value or system file size.

Table 9-14 lists AUTOGEN phases. See the *Guide to VMS Performance Management* for more information about AUTOGEN.

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
HELP	Provide information about AUTOGEN and its phases	None	None

NOTE
All data files are in the directory SYS$SYSTEM:.

9.9.2 Reconfiguring the System with SYSGEN

The AUTOGEN command procedure sets system parameter values and creates system files using the SYSGEN utility. In general, you should use AUTOGEN rather than directly using SYSGEN, because AUTOGEN can modify related parameters to improve system performance. However, certain system parameters have no effect on system data structures. One example is the parameter UAFALTERNATE, used in Module 6. It is acceptable to use SYSGEN to change such a parameter, because no other parameters are related. In addition, you can use SYSGEN to specify the system startup file, to back up system parameters in a parameter file, and to handle other functions not available from AUTOGEN.

To invoke the SYSGEN utility, issue the DCL command:

```
$ RUN SYS$SYSTEM:SYSGEN
```

The system parameters stored in SYS$SYSTEM:VAXVMSSYS.PAR are called the **current** system parameters. The parameters kept in memory while the system is running are called the **active** system parameters. Immediately after system startup, the current parameters and the active parameters are identical. You can use SYSGEN to modify the current parameters, the active parameters, or both.

If you modify the **current** parameters, the changes do not take effect until the next time the system starts. If you modify the **active** parameters, the changes take effect immediately (if possible). Parameters whose changes can take effect while the system is running are called dynamic parameters. Changing active parameters that are not dynamic has no effect. Changes to the active parameters are in effect only until the next system startup. For a change to be permanent, you must change the current system parameters.

9.9.3 Running AUTOGEN

9.9.3.1 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 wish to change any parameters, edit MODPARAMS.DAT and rerun AUTOGEN, as in step 1.
4. `@SYS$UPDATE:AUTOGEN SETPARAMS REBOOT`

9.9.3.2 Changing System File Sizes

1. `@SYS$UPDATE:AUTOGEN SAVPARAMS TESTFILES`
2. Examine file sizes.
3. If you wish to change file sizes, edit MODPARAMS.DAT and rerun AUTOGEN, as in step 1.
4. `@SYS$UPDATE:AUTOGEN GENPARAMS REBOOT`

```
!
! 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)
```

Example 9–15 A MODPARAMS.DAT File

Table 9-15 lists the principal SYSGEN commands and their functions. For more information on these and other SYSGEN commands, see the reading in the *VMS System Generation Utility Manual*.

Table 9–15 Using the SYSGEN Utility

Function	Command Format

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

Modifying a system parameter in the SYSGEN buffer	`SYSGEN> SET parameter value`
	`SYSGEN> SET UAFALTERNATE 1`

Read a set of system parameters in the SYSGEN buffer	`SYSGEN> USE source`
	`SYSGEN> USE ACTIVE`
	`SYSGEN> USE CURRENT`
	`SYSGEN> USE DEFAULT`
	`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 -`
	`_SYSGEN> /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 -`
	`_SYSGEN> /SWAPFILE`

Deactivate a paging or swapping file	`SYSGEN> DEINSTALL system-file /type`
	`SYSGEN> DEINSTALL DISKALT:[SYSEXE]NEWPAGEFILE.SYS -`
	`_SYSGEN> /PAGEFILE`

Use an alternate DCL startup file	`SYSGEN> SET/STARTUP SYS$SYSTEM:startup-file`
	`SYSGEN> SET/STARTUP SYS$SYSTEM:ALTSTART.COM`

Display HELP information	`SYSGEN> HELP [command[/qualifier]]`
Exit from SYSGEN	`SYSGEN> EXIT`

Figure 9-2 illustrates the way in which SYSGEN affects the VMS system parameters.

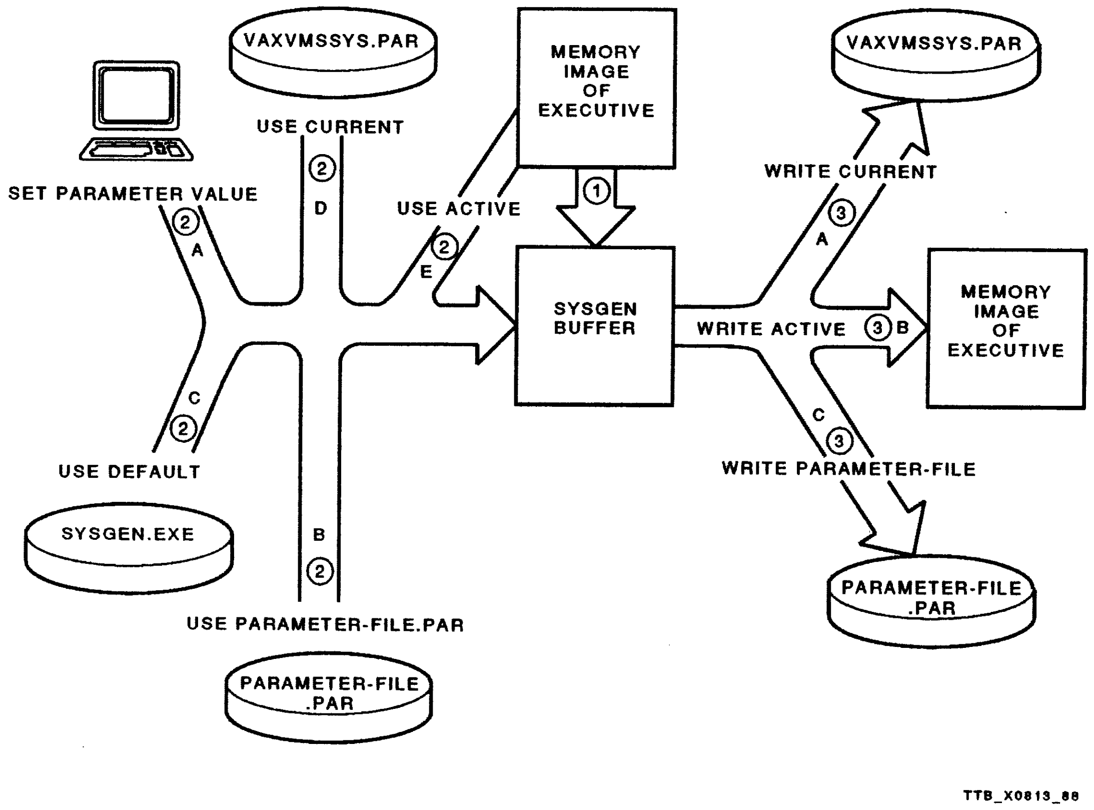


Figure 9–2 Using the SYSGEN Utility to Modify System Parameters

The numbers and letters displayed in the figure are keyed to the following notes.

Notes on Figure 9-2:

1. When you start up SYSGEN, it copies the active system parameters from the executive image in memory to the SYSGEN buffer.
2. SYSGEN can modify the parameters in the SYSGEN buffer in several ways:
   a. It can modify individual parameters. SET commands specify the parameters and their new values.
   b. It can replace all parameter values with values stored in a .PAR file. The WRITE command of SYSGEN creates such files, which contain values for every system parameter.
   c. It can replace all parameter values with **default** values stored in the SYSGEN image.
   d. It can replace all parameter values with the **current** values stored in SYS$SYSTEM:VAXVMSSYS.PAR.
   e. It can replace all parameter values with the **active** values stored in the executive image in memory.
3. After the parameter values in the SYSGEN buffer have been suitably modified, the WRITE command can record them in several places:
   a. It can replace all **current** values with those in the SYSGEN buffer.
   b. It can replace all **active** values with those in the SYSGEN buffer.
   c. It can record the values in the SYSGEN buffer in a parameter file, for later use by a USE command.

9.9.3.3 Additional References

- For more information on using the SYSGEN utility to modify system parameter values, see the *Guide to VMS System Management and Daily Operations*.
- For more information on using the SYSGEN utility to create system files, see the *Guide to VMS System Management and Daily Operations*, which also explains how to create these files with the command procedure SYS$UPDATE:SWAPFILES.COM.

9.9.4 Performance Tuning

System parameters control system functions such as scheduling, memory management, and input/output. These parameter values and the sizes of certain files affect system performance (for example, memory management makes heavy use of system files). The *VMS System Performance Management* course covers this relationship of system parameter values to system performance, while the *VMS Internals* course covers system functions in detail.

However, you are not expected to improve system performance by selectively modifying system parameter values or file sizes. You should use AUTOGEN to enable the VMS system to match the system configuration to hardware resources when those resources are modified. Many factors influence system performance. If you use AUTOGEN, it is unlikely that you will be able to tune system parameter values and file sizes much more effectively. See *VMS System Performance Management* for an in-depth discussion of factors that affect system performance.

9.9.4.1 Additional References

- For more information on using AUTOGEN to modify system parameters and files, read the section on System Parameters and Files in the *Guide to VMS Performance Management.*
- For more information on SYSGEN commands, read the *VMS System Generation Utility Manual.*
- For more information on modifying system parameters using SYSGEN, read the section on System Parameters in Module 11 of the *Guide to VMS System Management and Daily Operations.*
- For more information on creating system files using the SYSGEN utility or the SWAPFILES command procedure, read the section on System Files in Module 11 of the *Guide to VMS System Management and Daily Operations.*
- For more information on factors affecting system performance, read the *Guide to VMS Performance Management.*

9.10 Laboratory Exercises

In these exercises, you use a command procedure called HARMLESS_AUTOGEN.COM. This file can be found in a directory pointed to by the logical name COURSE$V5SYSMGT.

HARMLESS_AUTOGEN mimics the actions of SYS$UPDATE:AUTOGEN.COM, except that it does not create or modify system files or shut down the system. Instead, it creates files in your default login directory (SYS$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.
   - Run HARMLESS_AUTOGEN so that it starts at the earliest possible phase and ends with the phase that reports on system file sizes.

2. 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.

3. 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.)

4. 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.

5. Still using this AUTOGEN.PAR file, generate a listing file that shows the values of the MAJOR parameters. Call the file MAJOR.LIS. Print the file.

9.11 Solutions to Laboratory Exercises

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 required.

4. `$ @dir:HARMLESS_AUTOGEN GENPARAMS REBOOT`

5.
```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> USE AUTOGEN.PAR
SYSGEN> SHOW SCSNODE
SYSGEN> SHOW SCSSYSTEMID
SYSGEN> SHOW LRPSIZE
      .
      .
      .
 SYSGEN> SET/OUTPUT=MAJOR.LIS
SYSGEN> SHOW/MAJOR
SYSGEN> EXIT
$ PRINT MAJOR.LIS
```