Managing DECwindows Motif for OpenVMS Systems

Order Number: AA–Q1E9A–TE

January 1994

Revision/Update Information:	This is a new manual.
Software Version:	DECwindows Motif Version 1.2 for OpenVMS AXP DECwindows Motif Version 1.2 for OpenVMS VAX
Operating System Version:	OpenVMS AXP Version 1.5 VMS Version 5.5–2

Digital Equipment Corporation
Maynard, Massachusetts

January 1994

Compaq Computer Corporation makes no representations that the use of its products in the manner described in this publication will not infringe on existing or future patent rights, nor do the descriptions contained in this publication imply the granting of licenses to make, use, or sell equipment or software in accordance with the description.

Possession, use, or copying of the software described in this publication is authorized only pursuant to a valid written license from Compaq or an authorized sublicensor.

© Compaq Computer Corporation 1994. All Rights Reserved.

The postpaid Readers Comments forms at the end of this document request your critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation: Alpha AXP, AXP, Bookreader, DECnet, DECwindows, Digital, OpenVMS, ReGIS, ULTRIX, VAX, VAX DOCUMENT, VAXcluster, VMS, VMScluster, VT100, VT320, and the DIGITAL logo.

The following are third-party trademarks:

Display PostScript and PostScript are registered trademarks of Adobe Systems Incorporated.

Motif is a trademark of the Open Software Foundation, Inc.

TEKTRONIX is a registered trademark of Tektronix, Inc.

Open Software Foundation is a trademark of the Open Software Foundation, Inc.

UNIX is a registered trademark licensed exclusively by X/Open Co. Ltd.

X Window System is a trademark of the Massachusetts Institute of Technology.

All other trademarks and registered trademarks are the property of their respective holders.

ZK6300

This document is available on CD-ROM.

This document was prepared using DECdocument, Version 3.3-1b.

Contents

Preface vii

1 System Overview of DECwindows Motif

1.1 Client/Server Model 1–1
1.1.1 The Display Server 1–3
1.1.2 The Client 1–3
1.1.3 The Transport Layer 1–3
1.2 DECwindows Motif Components 1–4
1.2.1 DECwindows Motif Layered Product Components 1–4
1.2.2 Operating System Bundled Components 1–5

2 Starting DECwindows Motif

2.1 System Startup Considerations 2–1
2.2 Understanding the Startup Procedure 2–2
2.2.1 Using the DECW$STARTUP.COM Procedure 2–3
2.2.2 The Startup Process 2–4

3 Configuring the Display Server

3.1 Customizing the DECwindows Server Startup Environment 3–1
3.1.1 Workstation-Specific Customizations 3–2
3.2 Server Customization Parameters 3–4
3.2.1 Server Process 3–7
3.2.2 Device Setup 3–10
3.2.3 Transport/Connection Setup 3–14
3.2.4 Font Setup 3–16
3.2.4.1 Font Path 3–16
3.2.5 Font Caching 3–17
3.2.5.1 VAX Dynamic Font Caching (VAX Only) 3–18
3.2.6 Scaling Bitmap Fonts (AXP Only) 3–20
3.2.7 Keyboard Setup 3–21

3.2.8 Mouse Setup 3–24
3.2.9 Color Database 3–25
3.2.10 Screen Saver Setup 3–25
3.2.11 Backing Store and Save Under Setup 3–27
3.2.12 Error Reporting 3–30
3.3 Customization Examples 3–34
3.3.1 Sample Workstation Setup 3–34
3.3.2 Setting Up a Dual-Head System on a VAX System (VAX Only) 3–34
3.3.2.1 Sample Setup for a Dual-Head System (VAX Only) 3–34
3.3.2.2 Sample Setup for a Dual-Head System With Different Monitors (VAX Only) 3–35
3.3.3 Setting Up a Dual-Head System on an AXP System (AXP Only) 3–36
3.3.4 Sample Setup for Transport Connections 3–36
3.3.5 Changing the Default Keyboard Layout 3–37
3.4 Specifying New Fonts 3–38
3.4.1 Using Third-Party Fonts 3–38
3.4.2 Using Nonstandard Fonts 3–40
3.4.3 Font Server Support (AXP Only) 3–40
3.5 DECwindows Transport 3–41
3.5.1 Using Local Transport 3–41
3.5.2 Using DECnet Transport 3–41
3.6 Using Alternate Transport 3–41
3.6.1 Using DEC TCP/IP 3–42
3.6.2 Using Other Transports with DECwindows 3–43
3.6.2.1 Writing a Transport Interface 3–43
3.6.2.2 Installing the Transport Interface Image 3–43
3.7 Determining Current Server Parameters 3–44
3.7.1 Displaying the Server Logical Name Table 3–44
3.7.2 Using the XDPYINFO Utility 3–46
3.7.3 Using the XSET Utility 3–48

4 Using DECwindows

4.1 Setting the Display 4–1
4.1.1 Examples for Setting the Display 4–2
4.1.2 Using the SHOW DISPLAY Command 4–4
4.2 The Login Sequence 4–5
4.3 Customizing the Session Manager Environment 4–7
4.3.1 Creating a Custom Bookreader Directory 4–10
4.3.2 Changing Your Logo 4–11

4.3.3 Creating Dedicated Accounts 4–12
4.3.3.1 Modifying the Session Manager Command Procedure . . . 4–12
4.3.3.2 Modifying the Session Manager Executable File 4–15
4.3.3.3 Modifying the Session Manager Profile File 4–17
4.4 Modifying Session Manager Behavior 4–19
4.5 Modifying System Resource Files 4–20
4.6 Customizing Print Formats 4–22
4.6.1 Defining Print Formats 4–22
4.6.2 Logical Names and Print Formats 4–23

A Tuning the DECwindows System

A.1 Establishing UAF Parameters for DECwindows Applications . . . A–1
A.2 Establishing System Parameters for DECwindows Applications and Display Server A–2

B DECwindows Motif Keymap Names

C Troubleshooting Guidelines

C.1 General Troubleshooting Techniques C–1
C.2 Common Problems C–6
C.2.1 Start Session Dialog Box C–7
C.2.2 Licensing C–8
C.2.3 FileView C–8
C.2.4 Window Manager C–9
C.2.5 Font Access C–9
C.2.6 Keyboard, Monitor, and Pointer C–12
C.2.7 Network C–14
C.2.8 Applications C–15
C.2.9 Session Manager C–19
C.2.10 DECterm C–22
C.2.11 X Terminals and Third-Party Workstations C–27
C.2.12 X Server C–31

Index

Examples

3–1	Workstation-Specific Customizations	3–3
3–2	Sample Workstation Setup	3–34
3–3	Sample Setup for a Dual-Head System (VAX Only)	3–35
3–4	Sample Setup for a Dual-Head System With Different Monitors (VAX Only)	3–36
3–5	Sample Setup for Transport Connections	3–36
3–6	Sample Setup for Dutch Keyboards and 100-dpi Monitors	3–38
3–7	Displaying the Server Logical Name Table	3–44
4–1	Using a Private Session Manager Command File	4–13
4–2	Modifying the Session Manager Executable File	4–16

Figures

1–1	DECwindows Motif Architecture	1–2
2–1	DECwindows Startup Command Procedure Flow	2–3

Tables

2–1	System Logicals	2–1
2–2	Startup Parameter Values	2–4
3–1	DECwindows Customization Parameters	3–5
3–2	Default Process Quotas for VAX and AXP Systems	3–7
4–1	Client Areas That Can Be Modified	4–8
4–2	Session Manager Logicals	4–19
4–3	Logical Names for the Print Dialog Box	4–23
A–1	Recommended Settings for UAF Limits	A–1
A–2	Recommended Settings for System Parameters on VAX Systems	A–3
A–3	Recommended Settings for System Parameters on AXP Systems	A–5
B–1	DECwindows Motif Keymap	B–1

Preface

Intended Audience

The intended audience for this manual is experienced OpenVMS AXP and VAX system managers who need to manage and customize DECwindows Motif for VMScluster or standalone systems.

Document Structure

This manual consists of the following chapters and appendixes:

- Chapter 1—An overview of the DECwindows Motif architecture and a description of DECwindows components.
- Chapter 2—A description of the startup process and how to customize the startup command procedure.
- Chapter 3—How to configure and customize the display server, including the use of symbols for server process, device setup, font setup, backing store, extensions, and error reporting. This chapter also includes customization examples for dual-head setup, for specifying new fonts, and alternative methods of transport.
- Chapter 4—A description of the DECwindows Motif login sequence and how to customize the Session Manager environment, including creating your own logo, creating dedicated accounts, and customizing print formats.
- Appendix A—Recommended settings for UAF limits and system parameters for both OpenVMS AXP and OpenVMS VAX systems.
- Appendix B—Lists the DECwindows Motif keymap names. This appendix is arranged by the language for which each keyboard is designed.

- Appendix C—General troubleshooting techniques and a listing of common problems and suggested solutions.

Conventions

In this manual, every use of OpenVMS AXP means the OpenVMS AXP operating system, every use of OpenVMS VAX means the OpenVMS VAX operating system, and every use of OpenVMS means both the OpenVMS AXP operating system and the OpenVMS VAX operating system.

In this manual, every use of DECwindows and DECwindows Motif refers to DECwindows Motif for OpenVMS software.

The following conventions are also used in this manual:

Ctrl/*x*	A sequence such as Ctrl/*x* indicates that you must hold down the key labeled Ctrl while you press another key or a pointing device button.
PF1 *x*	A sequence such as PF1 *x* indicates that you must first press and release the key labeled PF1, then press and release another key or a pointing device button.
. . .	A vertical ellipsis indicates the omission of items from a code example or command format; the items are omitted because they are not important to the topic being discussed.
boldface text	Boldface text represents the introduction of a new term or the name of an argument, an attribute, or a reason. Boldface text is also used to show user input in online versions of the manual.
italic text	Italic text emphasizes important information, indicates variables, and indicates complete titles of manuals. Italic text also represents information that can vary in system messages (for example, Internal error *number*), command lines (for example, /PRODUCER=*name*), and command parameters in text.
UPPERCASE TEXT	Uppercase text indicates a command, the name of a routine, the name of a file, or the abbreviation for a system privilege.

-	A hyphen in code examples indicates that additional arguments to the request are provided on the line that follows.
numbers	All numbers in text are assumed to be decimal, unless otherwise noted. Nondecimal radixes—binary, octal, or hexadecimal—are explicitly indicated.
mouse	The term *mouse* refers to any pointing device, such as a mouse, a puck, or a stylus.
MB1, MB2, MB3	MB1 indicates the left mouse button, MB2 indicates the middle mouse button, and MB3 indicates the right mouse button. (The buttons can be redefined by the user.)

Note

In this document, discussions that refer to VMScluster environments apply to both VAXcluster systems that include only OpenVMS VAX nodes and VMScluster systems that include at least one OpenVMS AXP node unless indicated otherwise.

1

System Overview of DECwindows Motif

This chapter provides an overview of the DECwindows Motif for OpenVMS VAX and AXP systems and includes the following topics:

Topic	Section
Client/Server Model	Section 1.1
Display Server	Section 1.1.1
Client	Section 1.1.2
The Transport Layer	Section 1.1.3
Components of DECwindows Motif	Section 1.2

1.1 Client/Server Model

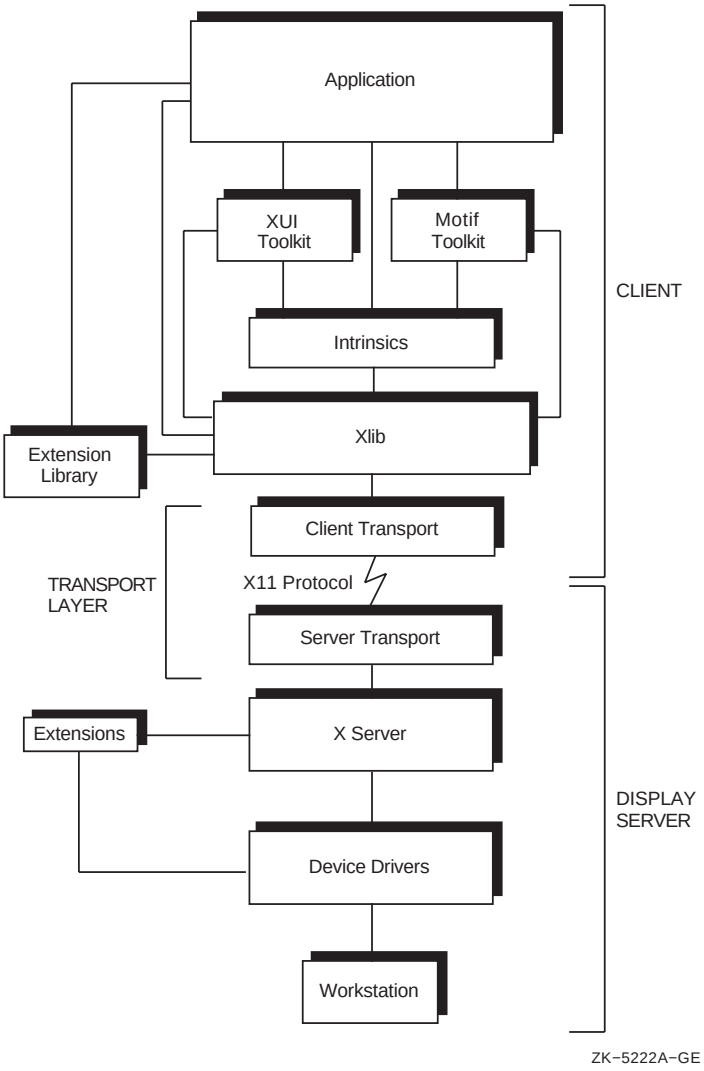
DECwindows Motif for OpenVMS software operates on a client/server model. A server is a single-shared process that performs operations at the request of many client processes. In the DECwindows Motif environment, the graphics applications (such as DECterm and DECwindows Mail) are the clients that interact with the display server. The display server manages the physical graphics display and input devices on behalf of the client applications.

In most client/server relationships, you think of the client system as a computer that sits on the desktop and the server system as being in the machine room. In DECwindows, as in all X window environments, the server sits on the desktop and displays graphics on the screen, while the client might be in the machine room running applications like DECterm and DECwindows Mail. To differentiate between the DECwindows server and other types of servers, the DECwindows server is referred to as the display server.

System Overview of DECwindows Motif
1.1 Client/Server Model

Figure 1–1 illustrates the DECwindows Motif client/display server architecture.

Figure 1–1 DECwindows Motif Architecture



1.1.1 The Display Server

The **display server** is the component of the architecture that allows application interfaces to interact with all the supported systems in the same way. Residing on the displaying system, the display server receives X protocol requests from clients through transport layers and performs the functions required to fulfill the request for a specific device.

Essentially, the server converts data that represents the request into commands that can be executed by the appropriate graphics device. When the user of an application enters data with the mouse or keyboard, the display server receives input from the device drivers and passes protocol packets back through the transport layers to Xlib and toolkit routines.

The display server supports asynchronous input from the user to the application and asynchronous output from the application to a display.

1.1.2 The Client

The **client** is a process that makes the X protocol request and is usually a user's application, such as DECterm or DECwindows Mail. The client controls what is displayed on the server display system and generates the user interface with which the user interacts.

1.1.3 The Transport Layer

In the DECwindows architecture, as in most client/server models, the client and server do not have to reside on the same machine. The client and server are connected by a transport layer that hides the details of the connection. The client and server must each contain their own transport component.

The **transport layer** is responsible for transferring data between the client and server systems and does not alter this data in any way.

DECwindows Motif for OpenVMS supports the following transport mechanisms:

- Local transport (shared memory)
- DECnet
- TCP/IP

1.2 DECwindows Motif Components

This section lists the components that makeup the DECwindows Motif layered product and the components that are bundled with the operating system.

1.2.1 DECwindows Motif Layered Product Components

The following components make up the DECwindows Motif layered product:

- Applications
  - Bookreader
  - Calculator
  - Calendar
  - Cardfiler
  - Clock
  - CDA Viewer
  - DECsound
  - DECterm
  - FileView
  - Mail
  - Notepad
  - Paint
  - PrintScreen
- Session management utilities
- Utilities
- Examples
- Libraries
  - Motif toolkit
  - Intrinsics toolkit
  - Xlib
  - Extensions library

1.2.2 Operating System Bundled Components

The following DECwindows components are bundled with the operating system:

- X server—Consists of a collection of the following shareable images:
  - Main entry-point stub
  - Device-independent server image
  - Device-dependent server image
  - Dynamically loadable extensions

  The main image and device-independent server image are directly linked together. During initialization, depending on which graphics devices are available and have been selected, the device-dependent server images are dynamically activated. The loadable extensions are dynamically activated either during initialization for OpenVMS AXP or when they are first used.
- Transport—Consists of a single common image and a collection of transport-specific images.

  The common transport image is built into the display server. Based on transport initialization parameters, one or more of the transport-specific images is dynamically activated.
- Device drivers—The DECwindows display server uses two types of drivers: input drivers and graphics/video drivers.
- Data files—The display server references the following data files:
  - Font files
  - Keymap files that define how keyboard keys are interpreted
  - Color database that defines how color names and RGB values are associated

2

Starting DECwindows Motif

This chapter describes the DECwindows startup process, which begins when the first DECwindows startup command file executes and continues until the DECwindows Start Session dialog box appears. This chapter contains the following sections:

Topic	Section
System Startup Considerations	Section 2.1
Understanding the Startup Procedure	Section 2.2
Using DECW$STARTUP.COM	Section 2.2.1
Startup Process	Section 2.2.2

2.1 System Startup Considerations

Before starting DECwindows, you may want to make changes or add logicals to the system startup file, SYSTARTUP_VMS.COM, which is located in the SYS$MANAGER directory. (On systems running VMS Version 5.5–2, the startup file is SYSTARTUP_ V5.COM.)

Table 2–1 lists each logical name and its meaning.

Table 2–1 System Logicals

Logical Name	Meaning
DECW$IGNORE_DECWINDOWS	Do not start up DECwindows.
DECW$IGNORE_WORKSTATION	Do not perform workstation-specific startup.

(continued on next page)

Table 2–1 (Cont.) System Logicals

Logical Name	Meaning
DECW$IGNORE_SUBPROCESS	Do not spawn as a subprocess.
DECW$IGNORE_AUTOGEN	Do not verify SYSGEN parameters.
DECW$IGNORE_DECNET	Do not wait for DECnet to start.
DECW$PRIMARY_DEVICE	Use a graphics device on a multihead system.

DECwindows Motif requires that certain system parameters be set to specific or minimum values (For information about these values, see Appendix A). During DECwindows startup, the system checks whether these parameter values are set appropriately. If the values are not set properly, the system prompts you to run AUTOGEN to modify the values.

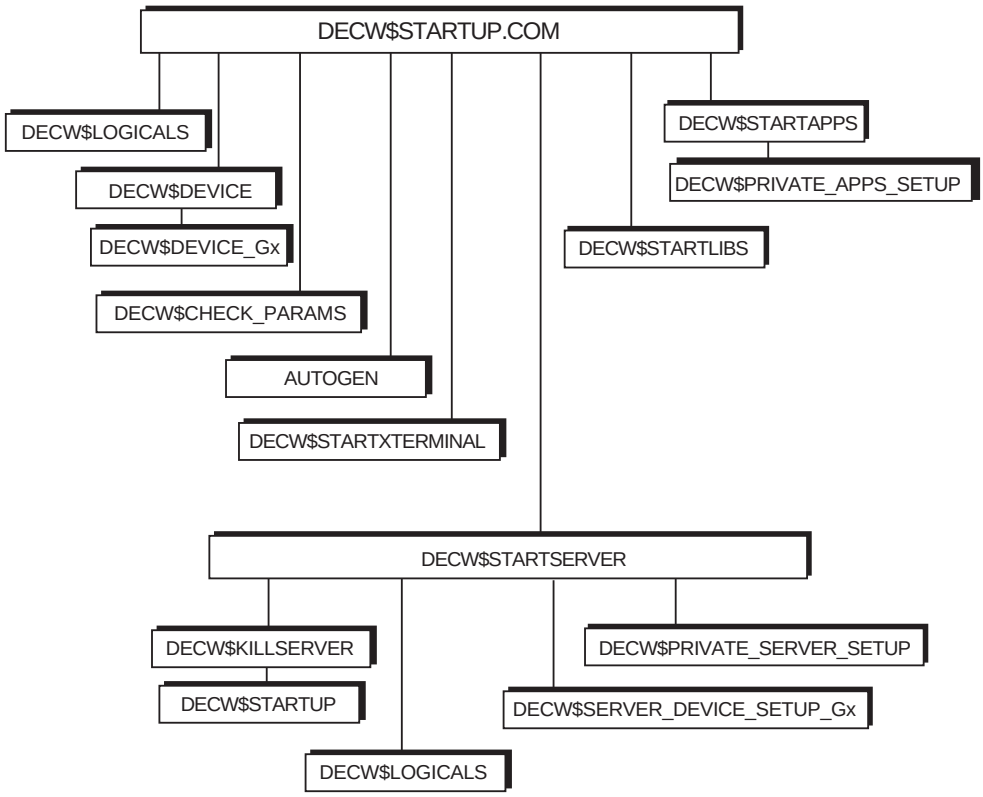
Caution

Parameter values set in SYS$SYSTEM:MODPARAMS.DAT override all other settings. Setting values in this file may prevent DECwindows Motif from starting.

2.2 Understanding the Startup Procedure

This section describes the DECwindows Motif startup process. Figure 2–1 illustrates the startup sequence.

Figure 2–1 DECwindows Startup Command Procedure Flow



2.2.1 Using the DECW$STARTUP.COM Procedure

The DECW$STARTUP.COM procedure is the only DECwindows command procedure with which a system manager invokes directly. This command procedure is used in the following ways:

- Called to startup DECwindows Motif during system startup.
- May be called by a privileged user to restart the display server if required.
- May be called by a privileged user to start the DECwindows server, libraries, or applications. "Starting" the libraries, for example, consists of defining logicals and installing images.

The DECW$STARTUP.COM command procedure takes one parameter. Table 2–2 lists the possible parameter values and the procedure they invoke.

Table 2–2 Startup Parameter Values

P1 Value	Invokes This Procedure	For More Information
" " (null)	Invokes full startup process	See step 1 in Section 2.2.2.
RESTART	Invokes DECW$STARTSERVER.COM	See step 5 in Section 2.2.2.
XTERMINAL	Invokes DECW$STARTXTERMINAL.COM if it exists	See step 4 in Section 2.2.2.
SERVER	Invokes DECW$STARTSERVER.COM	See step 5 in Section 2.2.2.
LIBS	Invokes DECW$STARTLIBS.COM	See step 6 in Section 2.2.2.
APPS	Invokes DECW$STARTAPPS.COM	See step 7 in Section 2.2.2.

2.2.2 The Startup Process

This section describes the events that occur when the DECwindows Motif startup procedure is invoked until the DECwindows Start Session dialog box appears. The startup process is as follows:

1. **DECW$LOGICALS.COM is invoked to create the DECW$LOGICAL_NAMES table.** DECwindows application startup and configuration parameters are defined in this step.
2. **DECW$DEVICE.COM is invoked to load or configure the DECwindows drivers.** This procedure also defines the symbols used in subsequent phases of the DECwindows startup sequence. Most importantly, DECW$DEVICE.COM sets the symbol DECW$DEVICE, which contains a list of the graphics devices that the display server attempts to use.
3. **DECW$CHECK_PARAMS.COM is invoked.** This procedure verifies that all system parameters required during DECwindows startup are set appropriately.

   If one or more parameters are not set correctly, DECW$CHECK_PARAMS.COM displays a list of parameters that need to be modified. The DECW$STARTUP.COM procedure asks whether the user wants to run AUTOGEN. If you invoked DECW$STARTUP.COM with the RESTART parameter, the question displays on the system console.

If you answer no to this question, DECW$STARTUP.COM displays a message stating that DECwindows cannot start until the system parameters are modified, and exits. If the user answers Yes, DECW$STARTUP.COM runs AUTOGEN from the GETDATA phase to REBOOT. This modifies system parameters and then reboots system.

To bypass the system parameter check, define the logical name DECW$IGNORE_AUTOGEN before running DECW$STARTUP.COM (see Table 2–1).

4. **DECW$STARTXTERMINAL.COM is invoked, if it exists.** If the logical DECW$INSTALL_XTERMINAL has been defined in the system startup procedure, the following functions are performed:
   a. Provides client support using the LAT transport
   b. Adds XTDRIVER as class driver for Xlib to communicate to the LT driver
   c. Installs DECW$TRANSPORT_LAT.EXE
   d. Provides font-file sharing through the DECW$FD process
   e. Runs the font daemon process as a detached process
   f. Installs DECnet access gateway server image with SYSNAM privileges
5. **DECW$STARTSERVER.COM is invoked to perform the following:**
   a. Handle the RESTART option to the DECW$STARTUP procedure. If RESTART is selected, a command file named DECW$KILLSERVER*n*.COM is created and executed as a detached process with the process name "Server *n* Restart". This procedure stops the current server process and executes DECW$STARTUP.COM with P1 set to null.
   b. Create the server-specific logical name table DECW$SERVER*n*_TABLE.
   c. Invoke the server customization file DECW$PRIVATE_SERVER_SETUP.COM. See Section 3.1 for information about how to customize the display server.

d. Populate the server-specific logical name table based on symbols defined in DECW$DEVICE.COM and DECW$PRIVATE_SERVER_SETUP.COM.

e. Purge previous versions of the server error log file DECW$SERVER_*n*_ERROR.LOG.

f. Check for the logical DECW$SERVER_DUMP that indicates whether the server saves a process dump file when a server crash occurs. Setting this logical disables the server's condition handler.

g. Check for user-specified server process parameters. See Section 3.2.1 for a list of server process parameters that you can modify.

h. Determine the server executable name. Normally this is SYS$SYSTEM:DECW$SERVER_MAIN.EXE. An alternative server image executes if:

   – The logical DECW$SERVER_MAIN is defined to point to an alternative server executable file.

   – A file exists with the name SYS$SYSTEM:DECW$SERVER_MAIN_*xx*.EXE, where *xx* are the first two letters of the primary graphics device defined by the symbol DECW$DEVICE

   Note that an alternate server image is executed only for bug fixes and special hardware releases.

i. Run the display server image as a detached process using the device name and process parameter information collected.

6. **DECW$STARTLIBS.COM is invoked to perform the following:**

   a. Defines DECwindows logicals

   b. Installs the Xlib, toolkit, and CDA shared libraries

   c. Connects the WSDRIVER for the DCL command SET DISPLAY

7. **DECW$STARTAPPS.COM is invoked to set up the user application developer environment.** This procedure also creates a WSA*n*: device in executive mode and creates a Start Session dialog box. If you have created the customization command file DECW$PRIVATE_APPS_SETUP.COM, it is invoked here. Use this file to customize the login sequence. See Section 4.3 for information about how to customize application startup.

3
Configuring the Display Server

This chapter describes how to configure the display server and includes the following topics:

Topic	Section
Customizing the DECwindows Server Startup Environment	Section 3.1
Workstation-Specific Customizations	Section 3.1.1
Server Customization Symbols and Logicals	Section 3.2
Customization Examples	Section 3.3
Specifying New Fonts	Section 3.4
DECwindows Transport	Section 3.5
Using Local Transport	Section 3.5.1
Using DECnet Transport	Section 3.5.2
Using Alternate Transports	Section 3.6
Using DEC TCP/IP	Section 3.6.1
Using Other Transports with DECwindows	Section 3.6.2
Writing a Transport Interface	Section 3.6.2.1
Installing the Transport Interface Image	Section 3.6.2.2
Determining Current System Parameters	Section 3.7

3.1 Customizing the DECwindows Server Startup Environment

The DECW$PRIVATE_SERVER_SETUP.TEMPLATE file is located in the SYS$MANAGER directory. This template file ships with the display server, which is bundled with the operating system. This file contains information you need to customize your DECwindows Motif environment.

If DECW$PRIVATE_SERVER_SETUP.COM is placed in SYS$COMMON:[SYSMGR], symbols defined within the file apply to every workstation system that shares that system disk, which is usually every member of a cluster.

To customize the DECwindows environment for all workstations in your VMScluster, perform the following steps:

1. Copy the template file into SYS$COMMON:[SYSMGR]DECW$PRIVATE_SERVER_SETUP.COM.
2. Edit the command file and search for the section titled *Cluster Common or Standalone Workstation Setup*. If no such section exists, search for NODE_LIST = " ".
3. Add any customizations in this section.
4. Restart the DECwindows server with the following command:

   ```
   $ @SYS$STARTUP:DECW$STARTUP RESTART
   ```

   You need to restart DECwindows on each workstation on which you want the customizations to take effect.

   Caution

   Restarting the DECwindows server disconnects all current client processes.

3.1.1 Workstation-Specific Customizations

In some cases, one or more workstations in a cluster may have special server requirements. Use one of the following two methods to customize specific workstations in your cluster.

Method 1

Create a private copy of the DECW$PRIVATE_SERVER_SETUP.COM file and place it in SYS$SPECIFIC:[SYSMGR] on each workstation that you want customized.

Method 2

Use the node list facility in the DECW$PRIVATE_SERVER_SETUP.COM file.

To use method 2, perform the following steps:

1. Edit the DECW$PRIVATE_SERVER_SETUP.COM and search for NODE_LIST = " ".
2. Add the nodes in your cluster between the quotation marks. For example:

   ```
   $ NODE_LIST = "NODE1/NODE2/NODE3"
   ```

3. At the end of the command procedure, add a label and your workstation-specific definitions.

   Example 3–1 shows how to customize NODE1, NODE2, and NODE3.

Example 3–1 Workstation-Specific Customizations

```
$!
$! server symbol definitions for NODE1
$!
$DO_NODE1:
$ DECW$SERVER_TRANSPORTS == "DECNET,LOCAL,TCPIP"
$ EXIT
$!
$! server symbol definitions for NODE2
$!
$DO_NODE2:
$ DECW$SERVER_DENSITY == "75"
$ EXIT
$!
$! server symbol definitions for NODE3
$!
$DO_NODE3:
$ DECW$DEFAULT_KEYBOARD_MAP == "US_LK201AA"
$ EXIT
```

3.2 Server Customization Parameters

This section describes the parameters that you can use to customize the DECwindows Motif environment.

These customizations modify the default behavior of the DECwindows display server. When the display server is started, it looks in the logical name table (DECW$SERVER*n*_TABLE) for logical names that can override the default characteristics.

The logical name table is created and populated during the DECwindows startup process. For any parameter defined in the DECW$PRIVATE_SERVER_SETUP.COM file, the startup procedure creates a logical name in the display-server logical name table with the same name as the parameter.

However, some parameters are used only during the startup procedure and do not have a matching logical name in the display server logical name table. These parameters either identify system parameters to be used in starting the display server process or are used in combination with other parameters to obtain the value of a logical name to be added to the display server logical name table. If the display server does not find a logical name in its private logical name table, it looks in the system logical name table. Therefore, you can use the DCL command DEFINE/SYSTEM in the DECW$PRIVATE_SERVER_ SETUP.COM file to directly define logicals for those functions that are supported but do not have an associated parameter in the version of OpenVMS that you are using.

In Section 3.2.1 through Section 3.2.12, each parameter description includes an example of how to use it. Note those cases where you need to define a logical instead of the symbol.

Table 3–1 lists the parameters alphabetically along with the type, default value, and range for each parameter.

Table 3–1 DECwindows Customization Parameters

Parameter	Type	Default Value	Range
†DECW$BITONAL	Boolean	False	
DECW$CLIENT_ERROR_THRESHOLD	Integer	1	
DECW$COLOR	Boolean	True	
‡DECW$CURSOR_SIZE	Integer	32	16,32
DECW$DEFAULT_KEYBOARD_MAP	String	Device dependent	
DECW$DEVICE	String list		
†DECW$FM_INIT_NUM_CACHE_ROWS	Integer	60	
†DECW$FM_MAX_ROW_CACHE_SIZE	Integer	300	
†DECW$FM_MIN_ROW_CACHE_SIZE	Integer	20	
†DECW$FM_NO_CACHE_AUTOTUNING	Boolean	False	
†DECW$FM_ROW_CACHE_EXTEND	Integer	5	
†DECW$FM_TIME_TO_SHRINK_CACHE	Integer	600	
‡DECW$FONT_SERVERS	String list		
DECW$KEYMAP	String		
DECW$MONITOR_DENSITY	Integer list		
†DECW$MULTI_HEAD	Boolean	False	
†DECW$PRIMARY_DEVICE	String		
DECW$RGBPATH	String	SYS$MANAGER: DECW$RGB.DAT	
DECW$SERVER_BUG_COMPATIBILITY	Boolean	True	
DECW$SERVER_CONNECT_LOG	Boolean	False	
DECW$SEVER_DEFAULT_VISUAL_CLASS	Integer	Device dependent	0...5
DECW$SERVER_DEFAULT_BACKING_STORE	Integer	0	0...2
DECW$SERVER_DENSITY	Integer	Device dependent	75,100
DECW$SERVER_DISABLE_BACKING_STORE	Boolean	False	
DECW$SERVER_DISABLE_SAVE_UNDER	Boolean	False	
DECW$SERVER_ENQUEUE_LIMIT	Integer	See Table 3–2	
DECW$SERVER_ERROR_LOG_TO_KEEP	Integer	2	

†VAX specific
‡AXP specific

(continued on next page)

Table 3–1 (Cont.) DECwindows Customization Parameters

Parameter	Type	Default Value	Range
‡DECW$SERVER_FONT_CACHE_SIZE	Integer	0	
‡DECW$SERVER_FONT_CACHE_UNIT	Integer	128	
DECW$SERVER_BELL_BASE_VOLUME	Integer	50	0...100
DECW$SERVER_DUMP	Boolean	False	
DECW$SERVER_ENABLE_KB_AUTOREPEAT	Boolean	True	
DECW$SERVER_ERROR_THRESHOLD	Integer	10	
DECW$SERVER_KEYCLICK_VOLUME	Integer	0	0...100
†DECW$SERVER_MAIN_KB_UPDOWN	Boolean	False	
DECW$SERVER_MOUSE_ACCELERATION	Integer	2	0..2
DECW$SERVER_MOUSE_THRESHOLD	Integer	4	
DECW$SERVER_PAGE_FILE	Integer	See Table 3–2	
DECW$SERVER_RETRY_WRITE_INTERVAL	Integer	500	
DECW$SERVER_RETRY_WRITE_TIMEOUT	Integer	30000	
‡DECW$SERVER_SCALE_BITMAP_FONTS	Boolean	False	
DECW$SERVER_SCREEN_SAVER_INTERVAL	Integer	600	
DECW$SERVER_SCREEN_SAVER_PREFER_BLANKING	Boolean	True	
DECW$SERVER_SCREEN_SAVER_TIMEOUT	Integer	600	
DECW$SERVER_TRANSPORTS	String list	Local, DECnet	
DECW$SERVER_WSDEF	Integer	See Table 3–2	
DECW$SERVER_WSEXTENT	Integer	See Table 3–2	
DECW$SERVER_WSQUOTA	Integer	See Table 3–2	
DECW$XPORT_SYNC_TIMEOUT	Integer	30000	

†VAX specific
‡AXP specific

The following sections contain definitions and examples for all symbols and logicals listed in Table 3–1. The symbols and logicals are grouped into the following areas:

- Server process
- Device setup
- Transport
- Font

- Keyboard
- Mouse
- Color database
- Screen saver
- Backing store
- Error reporting

3.2.1 Server Process

As part of the DECwindows startup process, the DECwindows display server is invoked as a detached process. Normally, the default process quotas assigned to the server process are sufficient. Table 3–2 lists the default settings for both VAX and AXP systems.

Table 3–2 Default Process Quotas for VAX and AXP Systems

Logical	VAX	AXP
DECW$SERVER_WSDEF	500	3000
DECW$SERVER_WSQUOTA	1300	6000
DECW$SERVER_WSEXTENT	WSMAX[1]	WSMAX[1]
DECW$SERVER_PAGE_FILE	25000	50000
DECW$SERVER_FILE_LIMIT	200	200
DECW$SERVER_ENQUEUE_LIMIT	512	512

[1]Equal to the WSMAX system parameter

However, in some instances, certain parameters may need to be increased. By defining symbols in DECW$PRIVATE_SERVER_SETUP.COM, server process quotas can be adjusted.

There are two primary cases where you need to increase these parameter values:

- **High memory usage**—This results from running applications that use many large pixmaps. In addition, extensive use of the Display PostScript or PHIGS extensions (PEX) requires more memory.

To adjust for a high memory-usage environment, increase the value assigned to DECW$SERVER_PAGE_FILE. This controls the PGFLQUOTA of the server process. Note that this parameter is limited by the VIRTUALPAGECNT system parameter and by the size of the system page file. In addition, you may consider increasing the values for DECW$SERVER_WSDEF and DECW$SERVER_WSQUOTA.

- **Concurrent use of many fonts**— All font files that are referenced and for which font caching is enabled remain open until all of the client applications that reference those fonts are terminated.

  This occurs on VAX systems and on AXP systems with 16-bit font caching enabled.

  To adjust for this situation, increase the value of DECW$SERVER_FILE_LIMIT. This value must be greater than the total number of fonts used by any set of concurrently active client applications.

DECW$SERVER_WSDEF

This parameter defines the process limits (in pages) to be applied to the DECwindows server process. DECW$SERVER_WSDEF values that are larger than the value of DECW$SERVER_WSQUOTA revert to the value of DECW$SERVER_WSQUOTA. For information about establishing working set sizes, see the *Guide to OpenVMS Performance Management*.

The following logical definition defines a working set size of 512 pages for the DECwindows server:

Example

```
$ DEFINE DECW$SERVER_WSDEF 512
```

DECW$SERVER_WSQUOTA

This parameter defines the maximum amount of physical memory (working set) pages (pagelets on DECwindows Motif for OpenVMS running on an AXP system) that can be allocated to the DECwindows server. For information about establishing working set sizes, see the *Guide to OpenVMS Performance Management*.

The following logical definition establishes the maximum number of pages allocated to the DECwindows server as 2048 pages:

Example

```
$ DEFINE DECW$SERVER_WSQUOTA 2048
```

DECW$SERVER_WSEXTENT

This parameter defines the absolute limit on physical memory pages that the DECwindows server is to be allocated if the server requires more pages than the DECW$SERVER_WSDEF value allots. The total number of pages allocated to the DECwindows server may exceed the value in DECW$SERVER_WSQUOTA (up to the value of DECW$SERVER_WSEXTENT if the additional pages are available). For information about establishing working set sizes, see the *Guide to OpenVMS Performance Management*.

The following symbol definition allocates 256 pages for the DECwindows server on an as-needed basis, not to exceed the value in DECW$SERVER_WSQUOTA:

Example

```
$ DEFINE DECW$SERVER_WSEXTENT 256
```

DECW$SERVER_PAGE_FILE

This parameter defines the maximum amount of virtual memory (in pages) that the DECwindows server can use.

The following logical definition increases the size of the page file to 30000 blocks:

Example

```
$ DEFINE DECW$SERVER_PAGE_FILE 30000
```

DECW$SERVER_FILE_LIMIT

This parameter defines the maximum number of files the server can open at one time. The default is 200 files.

The following logical definition increases the maximum number of files the server can open to 275:

Example

```
$ DEFINE DECW$SERVER_FILE_LIMIT 275
```

DECW$SERVER_ENQUEUE_LIMIT

This parameter defines the maximum number of outstanding locks that are used in sharing resources, particularly files, between processes. The default is 512 locks.

The following logical definition doubles the default enqueue limit to 1024 locks.

Example

```
$ DEFINE DECW$SERVER_ENQUEUE_LIMIT 1024
```

3.2.2 Device Setup

During startup, the DECwindows startup procedures attempt to identify and activate device-specific server components to manage all graphics devices of which the system is aware. You can use the symbols and logicals in this section to influence how many and which specific devices are used by the display server.

Additionally, there is some device information that the server cannot obtain from a device, or that you may want to override about the display device. For example, you may need to provide information about a special type of monitor with a nonstandard size or limited color capabilities.

On AXP systems, DECW$SERVER_PRIVATE_SETUP.COM contains a section for modifying device configuration parameters. The symbols DECW$PRIMARY_DEVICE and DECW$MULTI_HEAD should be defined in this section.

DECW$MULTI_HEAD (AXP Only)

On AXP systems, this parameter configures the system for multihead support. The DECW$MULTI_HEAD symbol is already set in the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.TEMPLATE file.

To activate this parameter, copy the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.TEMPLATE to DECW$PRIVATE_SERVER_SETUP.COM.

DECW$PRIMARY_DEVICE (AXP Only)

On AXP systems, the server uses this parameter to check for a device name by activating a specific DECW$DEVICE_*xx*.COM procedure, where *xx* is the string supplied for the symbol. This parameter is ignored if DECW$MULTI_HEAD is set to True. The following symbol definition assigns GXA0 as the primary device:

Example

```
$ DECW$PRIMARY_DEVICE == "GXA0"
```

DECW$DEVICE

This parameter allows you to specify explicitly the order and identify those graphics devices that are used in a multihead configuration.

Note

On VAX systems, this parameter is the only method to set up multiple monitors for multihead support.

The following symbol definition shows how to specify a list of graphics devices to be controlled by one server, one mouse, and one keyboard:

Example

```
$ DECW$DEVICE == "GAA0,GAB0"
```

DECW$DEFAULT_VISUAL_CLASS

This parameter overrides the default visual class for each head on a mutihead system. The visual classes, which are numeric and match the definitions in DECW$INCLUDE:X.h, are as follows:

0 = StaticGray
1 = GrayScale
2 = StaticColor
3 = PseudoColor
4 = TrueColor
5 = DirectColor

The default for a specific device type is dependent on the hardware and is typically PseudoColor for an 8-plane color board, and TrueColor for a 24-plane option. If you have a monochrome display, you can change the default visual class to GrayScale, which causes the system to convert colors to levels of gray. Output for GrayScale is on the green lead. You can assign multiple values to this parameter for each head in a multihead system.

The following symbol definition specifies PseudoColor on head 0, TrueColor on head 1, StaticGray on head 2:

Example

```
$ DECW$SEVER_DEFAULT_VISUAL_CLASS == "3,4,0"
```

Note

This parameter is not supported on VMS Version 5.5–2.

DECW$COLOR

This parameter specifies the use of color or grayscale for systems capable of displaying color. Monochrome hardware support uses the default value (False) and ignores any specification of this parameter.

If your system and monitor can display either color or grayscale, set this value to True in DECW$PRIVATE_SERVER_SETUP.COM as follows:

Example

```
$ DECW$COLOR == "T"
```

Note

This parameter is not supported on VMS Version 5.5–2.

DECW$BITONAL (VAX Only)

On VAX systems if you have GPX hardware, you can set the bitonal parameter to True in DECW$PRIVATE_SERVER_SETUP.COM. Other hardware configurations automatically set the value to False and ignore any specification of this parameter. The following example sets the bitonal symbol to True for GPX hardware:

Example

```
$ DECW$BITONAL == "T"
```

DECW$MONITOR_DENSITY

The monitor density defines the dots per inch (dpi) value of the monitor so that DECwindows applications can determine the actual width of the screen by using X Library routines.

The default value for the monitor density is the server density. Because few monitors are actually 75 or 100 dpi (the values used for DECW$SERVER_DENSITY with regard to font size), these values cannot be used to calculate accurately the actual width and length of items on the screen. By setting DECW$MONITOR_DENSITY to the actual value, you can obtain correct values for the width and height of the screen using X Library routines.

Use the following method to calculate the actual monitor density:

1. Determine the pixel width of the screen.

   Generally, the number of pixels is either 1024 or 1280, depending on the graphics adapter on your system. You can use the XDPYINFO utility to obtain the pixel width and height of the current display (see Section 3.7.2).

2. Measure across the visible portion of the screen (in inches).

3. Divide the pixel value by the screen value.

   If you have a VRT19 monitor and SPX graphics, make this calculation:

   ```
   1280 pixels / 13.5 inches  = 94.81 dpi
   ```

By rounding the dpi value to the nearest integer, assign 95 to DECW$MONITOR_DENSITY, as shown in the following example:

Example

```
$ DECW$MONITOR_DENSITY == "95"
```

Note

Setting different values for the monitor density and the server density can cause display problems with "what-you-see-is-what-you-get" (WYSIWYG) applications because you cannot scale the 75- and 100-dpi fonts to match the actual monitor density.

DECW$MONITOR_DENSITY can be set on a per-monitor basis. The following example shows how to set the monitor densities for a dual-head workstation, where screen 0 is set to 95 dpi and screen 1 is set to 75 dpi:

Example

```
$ DECW$MONITOR_DENSITY == "95,75"
```

3.2.3 Transport/Connection Setup

The DECwindows transport is the link between DECwindows client applications and the display server. Use the symbols in this section to control the types of available links and the timing characteristics of the link.

DECW$SERVER_TRANSPORTS

You can specify which transports your server monitors for incoming connections. Valid values for DECW$SERVER_TRANSPORTS are "DECNET", "LOCAL", and "TCPIP". If you have another transport, you can also specify its name. For information about alternative methods of transport, see Section 3.6.

The following example shows how to specify DECnet, local, and TCP/IP as the transports you will use:

Example

```
$ DECW$SERVER_TRANSPORTS == "DECNET,LOCAL,TCPIP"
```

DECW$XPORT_SYNC_TIMEOUT

This parameter defines the transport timeout value (in milliseconds). The default is 30000 milliseconds (30 seconds). If the client does not take action to empty its buffers before the timeout, the server disconnects the client.

The following logical definition extends the timeout value to 60000 milliseconds (1 minute):

Example

```
$ DEFINE/SYSTEM DECW$XPORT_SYNC_TIMEOUT 60000
```

DECW$SERVER_RETRY_WRITE_TIMEOUT

This parameter defines the number of milliseconds before which the DECwindows server stops attempting transport write requests. The default is 30000 milliseconds (30 seconds).

The following logical definition extends the retry total to 60000 milliseconds (1 minute):

Example

```
$ DEFINE/SYSTEM DECW$SERVER_RETRY_WRITE_TIMEOUT 60000
```

DECW$SERVER_RETRY_WRITE_INTERVAL

This parameter defines the number of milliseconds between failed transport write-request attempts. The default is 500 milliseconds (one-half second).

The following logical definition causes the DECwindows server to attempt transport write requests at an interval of 250 milliseconds.

Example

```
$ DEFINE/SYSTEM DECW$SERVER_RETRY_WRITE_INTERVAL 250
```

3.2.4 Font Setup

Displaying text is one of the major tasks of the display server. To display text, the display server is given a font name and a set of characters to be drawn. Usually, the font name corresponds to a particular font file resident on the server system. When trying to match a font name with a font file, the display server searches the **font path**, which is an ordered list of directories. Each valid font directory contains a font directory file (DECW$FONT_DIRECTORY*.DAT) that lists font file names and corresponding font names. The display server searches each directory file, in order, until it finds a matching font name.

In addition, DECwindows Motif supports font alias files (DECW$FONT_ALIAS*.DAT), which can reside in any valid font directory. These files map one font name to a different font name. After the display server checks the font directory file in a given font directory without finding a match, it will then search the font alias file, if it exists. If it finds a match in the alias file, the translated font name is substituted and the search is restarted from the beginning of the font path.

3.2.4.1 Font Path

As stated in Section 3.2.4, the font path is an ordered list of directories in which font files that are available to the display server reside. The font path is constructed from a number of setup parameters.

The type of available graphics devices may also influence the contents of the font path. The logical name DECW$FONT in the server logical name table contains the font path. The font path is a subset of the following directories:

```
SYS$COMMON:[SYSFONT.DECW.CURSOR32]
SYS$COMMON:[SYSFONT.DECW.CURSOR16]
SYS$COMMON:[SYSFONT.DECW.100DPI]
SYS$COMMON:[SYSFONT.DECW.75DPI]
SYS$COMMON:[SYSFONT.DECW.COMMON]
```

On AXP systems, the following directories are also included in the font path:

```
SYS$COMMON:[SYSFONT.DECW.SPEEDO]
SYS$COMMON:[SYSFONT.DECW.TYPE1]
```

Pointers to font servers can also be added to the end of the font path. For more iformation, see Section 3.4.3.

A USER variant of each directory exists in which user-supplied fonts can be added for example, SYS$COMMON:[SYSFONT.DECW.USER_100DPI]. For more information, see Section 3.4.

DECW$SERVER_DENSITY

The server density value is used to determine the font size to use, either 75 or 100 dots per inch (dpi). If you have a 100-dpi monitor, you can have a screen density of 75 or 100 dpi. Fonts intended for 75-dpi monitors that are displayed on 100-dpi monitors may appear small and difficult to read.

The following symbol definition sets the screen density to use 100-dpi fonts.

Example

```
$ DECW$SERVER_DENSITY == "100"
```

DECW$CURSOR_SIZE (AXP Only)

This parameter defines which cursor font directories are included in the font path. It can be set to 16 to include only the 16x16 cursors, or 32 to include both the 32x32 and 16x16 cursor. The larger cursor size resulting from a value of 32 is usually appropriate for a 100-dpi display.

Example

```
$ DECW$CURSOR_SIZE == "16"
```

3.2.5 Font Caching

A font file contains a compressed binary representation of all the **glyphs**, or characters, within that font. For example, the following VAX font file contains all the information for the glyphs in the Helvetica 12-point font:

```
DECW$SYSCOMMON:[SYSFONT.DECW.100DPI]HELVETICA12_100DPI.DECW$FONT
```

3.2.5.1 VAX Dynamic Font Caching (VAX Only)

The DECwindows Motif VAX display server provides a font manager with **dynamic font caching**. When a client requests a glyph, the DECwindows server determines whether the requested glyph is in one of the rows in the font cache. If a glyph is found in the font cache, the row containing the glyph is placed in the first row of the font cache, and all other rows are shifted down. This process speeds the response to repeated requests for glyphs.

If a requested glyph is not found in the font cache, the row in which the glyph is stored is retrieved from a font file. The new glyph is placed in the first row of the font cache. Depending on the font cache parameters, another row of glyphs may be removed from the font cache.

For example, if you specify a maximum number of the font cache rows to be 25 and a new row is added to the front of the cache, the server removes the last row from the font cache.

For a detailed description about font mechanics, see *VMS DECwindows Guide to Xlib (Release 4) Programming: MIT C Binding*.

Use the symbols in this section to modify font caching on VAX systems.

DECW$FM_INIT_NUM_CACHE_ROWS (VAX Only)

On VAX systems, this parameter defines the initial size of the cache, in rows. This value must fall within the minimum and maximum bounds that you define. If this value is outside of this range, the cache size takes on the value of the absolute minimum size of the cache. The default is 20 rows.

The following logical definition establishes the initial size of the font cache as 200 rows:

Example

```
$ DEFINE/SYSTEM DECW$FM_INIT_NUM_CACHE_ROWS 200
```

DECW$FM_MAX_ROW_CACHE_SIZE (VAX Only)

On VAX systems, this parameter defines the maximum number of rows that can be allocated to the font cache. The default is 300 rows. For large memory systems of 32MB or more, you may want to increase this value. For low memory systems of 12MB or less, you may want to decrease the default value.

The following logical definition increases the maximum number of rows that can be allocated to the font cache:

Example

```
$ DEFINE/SYSTEM DECW$FM_MAX_ROW_CACHE_SIZE 500
```

DECW$FM_MIN_ROW_CACHE_SIZE (VAX Only)

On VAX systems, this parameter defines the minimum number of rows that can be allocated to the font cache. The default minimum number of rows in the font cache is 20, which is generally acceptable for Latin font sets. You can set the value to a higher number using the DECW$FM_MIN_ROW_CACHE_ SIZE symbol; you cannot set it lower than 20 rows. For Asian DECwindows equipped machines with 16MB or more of memory, you may want to increase this value to 100 rows, for example, so that the cache always maintains a list of the 100 most frequently accessed rows.

The following logical definition extends the minimum number of rows to 100:

Example

```
$ DEFINE/SYSTEM DECW$FM_MIN_ROW_CACHE_SIZE 100
```

DECW$FM_NO_CACHE_AUTOTUNING (VAX Only)

On VAX systems, this parameter determines whether the font cache is dynamic or static. A dynamic font cache is one in which the system can modify parameters based on font usage. A static font cache is one in which parameters remain fixed. The default value, false, enables autotuning so that all font cache symbols are active. If you set the symbol value to true, you disable autotuning, but you can still set the initial number of rows to be allocated to the font cache using the DECW$FM_INIT_NUM_CACHE_ROWS symbol.

The following logical definition enables autotuning:

Example

```
$ DEFINE/SYSTEM DECW$FM_NO_CACHE_AUTOTUNING F
```

DECW$FM_ROW_CACHE_EXTEND (VAX Only)

On VAX systems, this parameter defines the number of rows to be added to the font cache when the cache is full and another row must be read in from a font file. The default is 5 rows.

The following logical definition causes the DECwindows server to allocate 10 rows to the font cache when needed, up to the maximum number of rows allowed:

Example

```
$ DEFINE/SYSTEM DECW$FM_ROW_CACHE_EXTEND 10
```

DECW$FM_TIME_TO_SHRINK_CACHE (VAX Only)

On VAX systems, this parameter defines the number of seconds that a row is allowed to remain in the font cache without having been accessed, after which the row is removed unless the size of the cache has reached the minimum number of rows allowed. The default is 600 seconds (10 minutes).

The following logical definition enables an unaccessed font to remain in the font cache for up to 1200 seconds (20 minutes). An unaccessed font can be removed in 20 minutes if the maximum number of rows is reached.

Example

```
$ DEFINE/SYSTEM DECW$FM_TIME_TO_SHRINK_CACHE 1200
```

3.2.6 Scaling Bitmap Fonts (AXP Only)

In addition to supporting several formats of scalable outline fonts, the AXP display server also can scale any of its bitmap fonts to any size.

Note, however, that the quality of these scaled bitmap fonts is significantly less than the original bitmap fonts or even comparable scaled outline fonts. By default, this capability is disabled. In some cases, this reduction in quality may be an acceptable tradeoff, given the greater flexibility in font sizes.

DECW$SERVER_SCALE_BITMAP_FONTS (AXP Only)

On AXP systems, this parameter enables the scaling of bitmap fonts to render arbitrary point-size characters. The default value (False) makes the AXP server act like the VAX server.

The following logical definition enables scalable fonts on an AXP system:

Example

```
$ DEFINE/SYSTEM DECW$SERVER_SCALE_BITMAP_FONTS T
```

3.2.7 Keyboard Setup

Some default characteristics of the keyboard attached to the server system can be modified. These include keyboard operations, such as keyclick and bell volume and autorepeat, as well as how the keyboard keys are mapped to keyboard independent symbols used by client applications.

All of the keyboard settings in this section are overriden by the Session Manager. Setting these symbols takes effect before the user logs in or if you are not using DECwindows Session Manager.

DECW$DEFAULT_KEYBOARD_MAP

You can specify the language for which your keyboard is designed.

Valid values for DECW$DEFAULT_KEYBOARD_MAP are the file names (without file type) in the SYS$COMMON:[SYS$KEYMAP.DECW.SYSTEM] and SYS$COMMON:[SYS$KEYMAP.DECW.USER] directories.

The following symbol definition changes the keyboard layout to a Dutch typewriter layout:

Example

```
$ DECW$DEFAULT_KEYBOARD_MAP == "DUTCH_LK201LH_TW"
```

For a list of keymap names, see Table B–1.

DECW$SERVER_BELL_BASE_VOLUME

This parameter determines the volume of the bell sound in a keyboard. Values are from 0 to 100, with 100 being the loudest. The default volume level is 50.

The following logical definition sets the volume to one quarter of full volume:

Example

```
$ DEFINE/SYSTEM DECW$SERVER_BELL_BASE_VOLUME 25
```

Note

This parameter is not supported on VMS Version 5.5–2.

DECW$SERVER_KEYCLICK_VOLUME

This parameter determines the volume of the click sound in a keyboard. Values are from 0 to 100, with 100 being the loudest. The default is 0.

The following symbol definition sets the volume to one quarter of full volume.

Example

```
$ DEFINE/SYSTEM DECW$SERVER_KEYCLICK_VOLUME 25
```

Note

This parameter is not supported on VMS Version 5.5–2.

DECW$SERVER_ENABLE_KB_AUTOREPEAT

Keyboard autorepeat is an option that causes a character to repeat itself automatically while that character key is pressed. You can enable this option by specifying True for this parameter.

The following logical definition enables keyboard autorepeating of typed characters:

Example

```
$ DEFINE/SYSTEM DECW$SERVER_ENABLE_KB_AUTOREPEAT T
```

Note

This parameter is not supported on VMS Version 5.5–2.

DECW$SERVER_MAIN_KB_UPDOWN (VAX Only)

On VAX systems, this parameter defines whether the server receives reports of the down and up movements made by pressing the main keyboard keys. When set to False, the keyboard driver reports a key release immediately after any main keyboard key is pressed. When set to True, key releases are not reported until a key is physically released.

Caution

Setting this value to True may interfere with the proper operation of your display as a console.

The following logical definition causes main keyboard key presses to be reported to the server:

Example

```
$ DEFINE/SYSTEM DECW$SERVER_MAIN_KB_UPDOWN T
```

3.2.8 Mouse Setup

Use the symbols in this section to modify mouse pointer characteristics. All the mouse pointer settings in this section are overriden by Session Manager. Setting these symbols takes effect before the user logs in or if you are not using DECwindows Session Manager.

DECW$SERVER_MOUSE_ACCELERATION

This parameter defines the relationship of mouse movement to pointer movement. The possible values are as follows:

2 (fast)
1 (medium)
0 (slow)

The following logical definition causes the pointer to move at the fast speed in relation to your mouse movements.

Example

```
$ DEFINE/SYSTEM DECW$SERVER_MOUSE_ACCELERATION 2
```

Note

This parameter is not supported on VMS Version 5.5–2.

DECW$SERVER_MOUSE_THRESHOLD

This parameter defines the minimum motion of the mouse (in pixels) at which time the DECwindows server is notified of the motion. The default is 4 pixels.

The following logical definition causes the mouse to be very sensitive to movement, resulting in the report of movement to the DECwindows server.

Example

```
$ DEFINE/SYSTEM DECW$SERVER_MOUSE_THRESHOLD 1
```

Note

This parameter is not supported on VMS Version 5.5–2.

3.2.9 Color Database

The display server uses a color database file to translate color names passed to the server from client applications into RGB values. Use the symbols in this section to modify color database characteristics.

DECW$RGBPATH

This parameter defines the RGB database file-name to be used with the server. The default name is SYS$MANAGER:DECW$RGB.DAT:

The following symbol definition changes the file name value to DECW$RGBPATH.DAT.

Example

```
$ DECW$RGBPATH == "SYS$MANAGER:DECW$NEWRGB.DAT"
```

Note

This parameter is not supported on VMS Version 5.5–2.

3.2.10 Screen Saver Setup

The Screen Saver feature causes the monitor to go blank after a specified length of time (10 minutes by default) during which no user input occurs. Note that the actual value is specified in seconds.

Optionally, the server displays a black or white X of a random size and location on the screen when the screen saver timeout expires. A second parameter specifies the interval after which the Screen Saver pattern is repainted. This Screen Saver is not recommended for normal use because the screen background remains visible.

DECW$SERVER_SCREEN_SAVER_PREFER_BLANKING

This parameter determines the method by which Screen Saver is performed. When the value is True (the default), the DECwindows server causes the video device driver to turn off the video signal when the Screen Saver timeout expires. When the value is False, the DECwindows server blanks the screen when the timeout expires.

The following logical definition causes the screen to be cleared when the Screen Saver timeout expires.

Example

```
$ DEFINE/SYSTEM DECW$SERVER_SCREEN_SAVER_PREFER_BLANKING F
```

Note

This parameter is not supported on VMS Version 5.5–2.

DECW$SERVER_SCREEN_SAVER_TIMEOUT

This parameter defines the initial time (in seconds) before the screen saver is activated, after which the screen saver interval takes effect. The default is 600 seconds (10 minutes).

The following logical definition causes the initial screen saver time to be 1200 seconds(20 minutes):

Example

```
$ DEFINE/SYSTEM DECW$SERVER_SCREEN_SAVER_TIMEOUT 1200
```

Note

This parameter is not supported on VMS Version 5.5–2.

DECW$SERVER_SCREEN_SAVER_INTERVAL

This parameter defines the number of minutes the server waits before repainting the screen background. The Screen Saver option rearranges the screen pixels after 10 minutes to avoid burning the screen phosphor. Some servers use nonblanking screen savers, such as swimming fish or logos. The default is 600 seconds.

The following logical definition causes the background screen to be repainted in 420-second (7-minute) intervals:

Example

```
$ DEFINE/SYSTEM DECW$SERVER_SCREEN_SAVER_INTERVAL 420
```

Note

This parameter is not supported on VMS Version 5.5–2.

3.2.11 Backing Store and Save Under Setup

In general, it is the responsibility of the client application to restore occluded regions of the screen once they are exposed. However, the display server has several techniques available to perform this operation on behalf of the client. These are the backing store and save under options.

Backing store saves portions of windows in server memory just before they are obscured. Later, when the portions are exposed, the server can repaint them without involving the client. This can drastically reduce the time required to repaint, particularly for windows containing complex graphics or for environments where the client/server link is slow.

Save under is a similar mechanism where just prior to painting a window, the server saves the portion of the screen that is about to be obscured.

Both of these mechanisms are available by default and can be set on a window-by-window basis. Heavy use of these mechanisms does increase the memory requirements of the server. You may want to disable these features when debugging client applications to ensure that the application handles all expose events properly.

DECW$SERVER_DEFAULT_BACKING_STORE

You can enable this option in three types of window states. The following table lists possible values for the symbol and their meanings:

Value	Meaning
2	Enable backing store at all times

Value	Meaning
1	Enable backing store only when window is mapped
0	Indeterminate; depends on the server and the device being used

The value for DECW$SERVER_DEFAULT_BACKING_STORE can be overridden in applications on a per-window basis.

A True value for DECW$SERVER_DISABLE_BACKING_STORE causes the backing store option not to be used.

Note

Both the backing store and save under options add a significant burden on the DECwindows server and therfore reduce performance.

The following logical definition causes the server to save and restore windows when the windows are mapped to the screen.

Example

```
$ DEFINE/SYSTEM DECW$SERVER_DEFAULT_BACKING_STORE 1
```

Note

This parameter is not supported on VMS Version 5.5–2.

DECW$SERVER_DISABLE_BACKING_STORE

Use this parameter to disable backing store support. The default value is False so that backing store is enabled by default.

When DECW$SERVER_DISABLE_BACKING_STORE is set to True (disable backing store), restart the server, at which time the backing store option is disabled. Thereafter, to reenable backing store, deassign or redefine DECW$SERVER_DISABLE_BACKING_STORE and restart the server.

If your application window comes up blank, then the application is incorrectly requesting backing store but is also waiting for an expose event to begin processing. Either modify your application so that it does not request backing store, or set DECW$SERVER_DISABLE_BACKING_STORE to True and restart the DECwindows server.

The following symbol definition disables backing store:

Example

```
$ DECW$SERVER_DISABLE_BACKING_STORE == "T"
```

DECW$SERVER_DISABLE_SAVE_UNDER

The save under option records window information that may be hidden when another window is placed on top. You can disable this option by specifying True for this parameter.

The save under option is similar to the backing store option, but only the occluded portions of windows are saved by the server so that when that portion of the window becomes visible, the server redraws that portion of the window.

Note

Both backing store and save under put a significant burden on the DECwindows server and reduce performance.

The following symbol definition disables the server's save under option:

Example

```
$ DEFINE/SYSTEM DECW$SERVER_DISABLE_SAVE_UNDER T
```

Note

This parameter is not supported on VMS Version 5.5–2.

3.2.12 Error Reporting

The display server creates a log file into which it writes informational and error messages to aid in troubleshooting problems. The log file is called SYS$MANAGER:DECW$SERVER_*n*_ERROR.LOG, where *n* is the server number (usually 0).

The display server uses a condition handler to trap error conditions (such as an access violation) that might otherwise cause the display server to stop. If the condition handler detects a nonfatal error, it tries to allow the display server to continue. The condition handler always records the error in the error log file. If the condition handler detects errors for a single client (2 by default), it disconnects the client.

When the condition handler recovers from an error, the display server may lose track of some resources, such as memory. Therefore, after a number of these exceptions (10 by default), the condition handler broadcasts a message to all users on the workstation indicating that the display server may be running in a degraded mode and suggests that the display server should be restarted. If you see a message like this, restart the display server at the next convenient opportunity. See Section 2.2.1 for instructions for restarting the server.

DECW$SERVER_BUG_COMPATIBILITY

Inconsistencies between pre–X11 R4 servers and the X11 protocol allowed some undefined bits to be set in some X requests. By strictly enforcing this part of the protocol, however, some applications that set undefined bits no longer worked. By setting bug compatibility to True (default), the server will continue to allow these applications to work; however, the applications should be recoded to comply with the X11 protocol.

The following symbol definition allows pre–X11 R4 servers and X11 protocol applications to function correctly:

Example

```
$ DECW$SERVER_BUG_COMPATIBILITY == "T"
```

DECW$SERVER_DUMP

By setting this parameter to True, it adds the /DUMP qualifier to the DCL command RUN, which causes a process dump if the server crashes. Specifying True also automatically disables the server condition handler (sets DECW$SERVER_DISABLE_CH to True).

The following logical definition adds the /DUMP qualifier to the DCL command RUN and disables the server condition handler:

Example

```
$ DEFINE DECW$SERVER_DUMP T
```

DECW$SERVER_ERROR_LOG_TO_KEEP

This parameter defines the number of versions of the error log file to save. The default is two versions.

The following symbol definition causes the DECwindows server to save the three most recent versions of the error log file:

Example

```
$ DECW$SERVER_ERROR_LOG_TO_KEEP == "3"
```

DECW$CLIENT_ERROR_THRESHOLD

This parameter defines the number of protocol errors generated by a single client above which the client will be terminated. The default is one protocol error.

The following logical definition causes the DECwindows server to terminate a client after 10 or more protocol errors have been generated:

Example

```
$ DEFINE/SYSTEM DECW$CLIENT_ERROR_THRESHOLD 10
```

DECW$SERVER_ERROR_THRESHOLD

This parameter defines the total number of server errors you allow (10 by default) before reporting the following message:

```
Server internal runtime error threshold exceeded (code = %x),
server performance may be degraded.
```

The symbol %x is replaced by the condition code (in hexadecimal) that forced the error. The most typical value is "c", specifying an access violation.

The following message is also broadcast to all logged-in terminals:

```
DECW$SERVER_ERROR: internal runtime error threshold exceeded.
Performance may be degraded, restart DECwindows software when
convenient by @SYS$MANAGER:DECW$STARTUP RESTART.
```

After every five server errors that exceed the threshold, the following message is sent to the SYS$MANAGER:DECW$SERVER_0_ERROR.LOG log file and broadcast to terminals connected to the server:

```
Server performance still degrading...
```

Three categories of errors are handled by the condition handler of the server which are reported to the log file.

Note

%d is decimal; %x is hexadecimal.

- **Errors that count toward the client error threshold**
  - If the error count is less than the client error threshold, the following error is reported:

    ```
    opcode %d is ignored due to internal runtime error %x
    for client %d (#error = %d)
    ```

  - If the error count equals or exceeds the client error threshold, the following error is reported:

    ```
    Client %d has made too many runtime errors (%d),
    its connection is marked for termination
    ```

- **Errors that disconnect a client**

  ```
  Client %d has been disconnected due to unrecoverable
  runtime error %x detected while processing opcode %d
  ```

Or, if the server is processing an opcode for a server extension, the following message is displayed:

```
Exception trapped while processing extension opcode %d
(extension id=%d)
```

- **Errors that terminate the server**

```
Unrecoverable server internal error (error code = %d) found,
terminating all connections.
   .
   .
   .
  List of active images and
  call chain at time of the error
   .
   .
   .
** SERVER INTERNAL RUNTIME ERROR ENCOUNTERED,
SERVER HAS JUST CRASHED!! **
*******************************************
```

The following logical definition causes the DECwindows server to report a system degradation message after 20 server errors have accumulated:

Example

```
$ DEFINE/SYSTEM DECW$SERVER_ERROR_THRESHOLD 20
```

DECW$SERVER_CONNECT_LOG

This parameter specifies whether normal client connect/disconnect messages get logged to the server error log file. By default, all successful connect/disconnect messages are not logged, which results in an increase in server performance.

The following symbol definition causes the server to record client connect/disconnect messages in the SYS$MANAGER:DECW$SERVER_0_ERROR.LOG server error log file.

Example

```
$ DECW$SERVER_CONNECT_LOG == "T"
```

3.3 Customization Examples

This section provides several examples that use the symbols described in Section 3.2.

3.3.1 Sample Workstation Setup

Example 3–2 shows some typical workstation-specific setup options.

Example 3–2 Sample Workstation Setup

```
do_MYNODE:
$  DECW$SERVER_DENSITY == 100                    ! 100 dpi font size
$  DECW$COLOR == "T"                             ! Color monitor
$  DECW$DEFAULT_KEYBOARD_MAP == "US_LK201AA"     ! USA LK201 keyboard
$  DECW$SERVER_TRANSPORTS == "DECNET,LOCAL,TCPIP" ! Three transports
$  DECW$SERVER_CONNECT_LOG == "F"                ! Do not record connect log
$ EXIT
```

3.3.2 Setting Up a Dual-Head System on a VAX System (VAX Only)

On VAX systems, you can use the examples and instructions in this section to set up dual-head systems. Section 3.3.3 provides instructions for setting up a dual-head system on an AXP system.

3.3.2.1 Sample Setup for a Dual-Head System (VAX Only)

Example 3–3 shows a sample setup for a dual-head VAXstation 2000 or VAXstation 3100 system that has one keyboard, one mouse, and two monitors (one color and one monochrome).

Example 3–3 Sample Setup for a Dual-Head System (VAX Only)

```
$do_VS2HD:
$ RUN SYS$SYSTEM:SYSGEN                  ! Configure the devices
$ CONNECT GCA0/ADAP=0/CSR=%x5000/VECT=%O400/NUMV=01/DRIVER=GCBDRIVER
$ DECW$COLOR == "T,F"                    ! Screen 0 (color) Screen 1 (B&W)
$ if f$getdvi("GAA0","EXISTS") then DECW$DEVICE == "GAA0,GCA0"
$ if f$getdvi("GEA0","EXISTS") then DECW$DEVICE == "GEA0,GCA0"
$ DECW$SERVER_DENSITY == 100             ! Use 100-dpi fonts  ❶
$ DECW$MONITOR_DENSITY == 100,75         ! Screen 0 (100 dpi) Screen 1 (75 dpi) ❷
$
$ exit
```

❶ The font density is chosen from the dpi value of the first screen. There is only one font path for a server, regardless of the number of screens. Set both screens to the same dpi value for consistency.

❷ Sets the monitor density to 100 dpi for screen 0 and 75 dpi for screen 1.

3.3.2.2 Sample Setup for a Dual-Head System With Different Monitors (VAX Only)

Example 3–4 shows a sample setup for a dual-head VS35*x*0 and GPX system. This configuration has one keyboard, one mouse, and two different color monitors. GPX systems support 16x16 cursor fonts, so the DECW$DEVICE font is defined to take this into account.

Example 3–4 Sample Setup for a Dual-Head System With Different Monitors (VAX Only)

```
$do_VS35x02HD:
$ SYSGEN = "$SYSGEN"                          ! Run SYSGEN configure the devices
$ IF .NOT. F$GETDVI("gaa0","exists") THEN EXIT
$ IF (F$GETDVI("gaa0","sts") .and. 16) .EQ. 0 THEN $ SYSGEN RELOAD GAADRIVER.EXE
$ IF (F$GETDVI("gba0","sts") .and. 16) .EQ. 0 THEN $ SYSGEN RELOAD GBBDRIVER.EXE
$ DECW$KEYBOARD :== TTA0:                     ! Keyboard device
$ DECW$POINTER  :== TTA1:                     ! Mouse device
$ DECW$COLOR == "T,T"                         ! Both are color monitors
$ DECW$DEVICE == "GBA0,GAA0"                  ! Monitor Devices
$ DECW$SERVER_DENSITY == 100                  ! 100 dpi fonts
$ DECW$MONITOR_DENSITY == 100,100             ! 100 dpi monitors
$ EXIT
```

3.3.3 Setting Up a Dual-Head System on an AXP System (AXP Only)

On AXP systems, the DECW$PRIVATE_SERVER_ SETUP.TEMPLATE includes the following command to set up your system for dual-head use:

```
$ IF DECW$DEVICE_COUNT .GT. 1 THEN DECW$MULTI_HEAD == 1
```

The template file is located in the SYS$MANAGER directory. To invoke dual-head support, copy the template file to SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM. Editing of this file is not needed.

3.3.4 Sample Setup for Transport Connections

Example 3–5 shows a sample setup for a system to use TCP/IP and local connections but not DECnet connections.

Example 3–5 Sample Setup for Transport Connections

```
$do_TCPIP:
$ decw$server_transports == "TCPIP,LOCAL"
$ exit
$ !
```

3.3.5 Changing the Default Keyboard Layout

This section describes how to change the default keyboard setting. See Appendix B for a listing of all languages, keyboard models, and keymap names.

The symbol DECW$KEYMAP translates to the directory specification where the keyboard mapping file resides. It is provided for your reference only. Do not modify its value.

To override the default keyboard layout, determine the correct name from the model number of your keyboard, as follows:

1. Turn the keyboard upside down and look for a label that specifies the model number. The model number should be in a format similar to LK401–*xx*.

   The model number may also be listed as simply LK401. In this case, the information you need is in another part of the label, where there will be a number that has the following format *nn-nnnnn-xx*.

2. Use the *xx* part of this number to choose a keymap name from Appendix B. The table is arranged based on the language for which each keyboard is designed. Choose the keymap ending in:

   _DP for the data processing keyboard layout
   _TW for the typewriter layout

3. After you choose a keymap name, modify the DECW$PRIVATE_SERVER_SETUP.COM file. For example, to change the keyboard layout to a Dutch typewriter layout, add the following line to DECW$PRIVATE_SERVER_SETUP.COM:

   ```
   $ DECW$DEFAULT_KEYBOARD_MAP == "DUTCH_LK201LH_TW"
   ```

4. Add a line similar to this for each workstation that does not have a North American keyboard layout, or add it to the common section for all workstations (if all the workstations use the same keyboard layout).

Example 3–6 shows a sample setup for two workstations with Dutch keyboards and 100-dpi monitors.

Example 3–6 Sample Setup for Dutch Keyboards and 100-dpi Monitors

```
$do_DUTCH:
$do_DUTCH2:
$ decw$server_density == 100
$ decw$default_keyboard_map == "DUTCH_LK201LH_DP"
$ exit
```

3.4 Specifying New Fonts

You can add new fonts by following these steps:

1. Place the DECW$FONT file in the DECW$SYSCOMMON:[SYSFONT.DECW.USER_*nn*DPI] directory.
2. Update the font database for the server by entering the following command from a privileged account:

   ```
   $ @SYS$UPDATE:DECW$MKFONTDIR
   ```

   This command creates new font directories for all layered-product fonts and runs automatically when DECwindows is installed.

3.4.1 Using Third-Party Fonts

If you want to use a third party font, choose one of the following options:

Font File is in BDF Format

1. Obtain the font source in bitmap distribution format (BDF) format.
2. Compile the font with the FONT command, as follows:

   ```
   $ FONT file-name
   ```

   See the *VMS DECwindows Guide to Xlib (Release 4) Programming: MIT C Binding* for a full description of the FONT command.
3. On VAX systems, copy the resulting *.DECW$FONT file to one of the following directories:

   ```
   SYS$COMMON:[SYSFONT.DECW.USER_75DPI]
   SYS$COMMON:[SYSFONT.DECW.USER_100DPI]
   SYS$COMMON:[SYSFONT.DECW.COMMON_USER]
   ```

4. On AXP systems, copy the resulting *.PCF file to one of the following directories:

```
SYS$COMMON:[SYSFONT.DECW.USER_75DPI]
SYS$COMMON:[SYSFONT.DECW.USER_100DPI]
SYS$COMMON:[SYSFONT.DECW.COMMON_USER]
```

5. Create new font directories from a privileged account using the following command:

```
$ @SYS$UPDATE:DECW$MKFONTDIR
```

End the current session and start a new session for the new font to become available.

Font is Charter or Lucida

If you want to use Charter or Lucida fonts and they do not reside on your system, use the example font-alias files provided by DECwindows to give the font name aliases to existing DECwindows font names.

For 75-dpi monitors, copy the following files to the SYS$COMMON:[SYSFONT.DECW.USER_75DPI] directory:

```
DECW$EXAMPLES:DECW$FONT_ALIAS_CHARTER.DAT
DECW$EXAMPLES:DECW$FONT_ALIAS_LUCIDA.DAT
```

For 100-dpi monitors, copy the following files to the SYS$COMMON:[SYSFONT.DECW.USER_100DPI] directory:

```
DECW$EXAMPLES:DECW$FONT_ALIAS_CHARTER_100DPI.DAT
DECW$EXAMPLES:DECW$FONT_ALIAS_LUCIDA_100DPI.DAT
```

Other Third-Party Fonts

To use other third-party fonts, create the DECW$FONT_ALIAS.DAT file in either the SYS$COMMON:[SYSFONT.DECW.USER_75DPI] directory or the SYS$COMMON:[SYSFONT.DECW.USER_100DPI] directory, depending on whether the third-party font is 75 dpi or 100 dpi.

The format of the font alias file is:

alias-font-name actual-font-name

If the font names contain embedded spaces, enclose the names in double quotes (" ").

The *alias-font-name* value is the third-party font name, and the *actual-font-name* value is a DECwindows font name.

See the *VMS DECwindows Guide to Xlib (Release 4) Programming: MIT C Binding* for DECwindows font names.

3.4.2 Using Nonstandard Fonts

If you use a font file name that is not a standard font name (standard font files names are those with a .DECW$FONT or .PCF file type), then perform one of the following actions:

- For fixed-width fonts, such as 6x10 fonts, copy the DECW$EXAMPLES:DECW$FONT_ALIAS_FILENAMES.DAT file to the SYS$COMMON:[SYSFONT.DECW.USER_COMMON] directory. End the current session and start a new session for the change to take effect.
- Modify your application program to use standard font names. See the *VMS DECwindows Guide to Xlib (Release 4) Programming: MIT C Binding* for standard DECwindows font names.

3.4.3 Font Server Support (AXP Only)

On AXP systems, the R5 display server includes a font server renderer that supports the use of font servers to access fonts. This allows font files to be distributed on systems other than the system on which the display server is running.

You can add font servers to the font path by defining the symbol DECW$FONT_SERVERS in the site-specific server setup file SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM. Depending on the transport type, use one of the following methods:

- To add support for the TCP transport, specify TCP as the network connection type that will communicate with the font server. Replace *node* with the Internet node name. Specify the TCP/IP port number for *port_number*, as shown in the following example:

  ```
  $ DECW$FONT_SERVERS == "TCP/node::port_number"
  ```

- To add support for the DECnet transport, specify DECNET as the network connection type that will communicate with the font server. Replace *node* with the DECnet node name. Specify the DECnet object name for *network_object*.

  ```
  $ DECW$FONT_SERVERS == "DECNET/node::network_object"
  ```

Multiple font servers can be added by defining the symbol as a comma-separated list. The symbol is not case sensitive.

3.5 DECwindows Transport

DECwindows transport is a general data-transfer mechanism for moving X Window protocol requests between the client and the server. DECwindows Motif provides support for the following transports:

- Local
- DECnet
- DEC TCP/IP

The default transports that DECwindows uses are Local and DECnet. This section describes these transports. Section 3.6 provides information about using alternative transports.

3.5.1 Using Local Transport

You should use local transport when a DECwindows client and the server are executing on the same OpenVMS system. Local transport improves performance because data is transferred between client and server through shared memory. Using local transport reduces the number of data copies in the system, and extra overhead is not incurred by accessing the network.

3.5.2 Using DECnet Transport

DECwindows also provides support for DECnet for OpenVMS as a transport mechanism. Before you can use DECnet, you must have DECnet for OpenVMS installed and running on your system. If DECnet or DEC TCP/IP is shut down while the server is attached to it, the transport will continuously poll the network to reattach when the network is restarted.

3.6 Using Alternate Transport

This section describes how to use alternate transports and includes information about writing and installing a transport interface layer.

3.6.1 Using DEC TCP/IP

DECwindows Motif contains support for the TCP/IP transport. This support requires DEC TCP/IP Services for OpenVMS or a functionally compatible TCP/IP product. Before you can use DECwindows TCP/IP transport interface, you must install DEC TCP/IP Services for OpenVMS.

If you want to use the TCP/IP as a transport, you must first have the OpenVMS/ULTRIX Connection software (UCX) installed on your system. You do not need a UCX Product Authorization Key (PAK) to use UCX with DECwindows. For information about UCX, see the appropriate UCX documents. DECwindows Motif for OpenVMS supports only UCX for TCP/IP transport use.

If you intend to use DEC TCP/IP Services for OpenVMS as a network transport for DECwindows Motif for OpenVMS, you must also include the following line in SYS$MANAGER:SYSTARTUP_VMS.COM file (or the SYS$MANAGER:SYSTARTUP_V5.COM file in VMS Version 5.5–2 systems) :

```
$ @SYS$STARTUP:UCX$STARTUP
```

Also, the TCP/IP transport does not pass user names for authorization to the server. If you are using TCP/IP transport for a given node, use an asterisk (*) in the Username entry box of the Security Options dialog box. To get the Security Options dialog box, pull down the Options menu from the Session Manager's menu, and choose the Security... menu item.

If you are using TCP/IP to display an application on your own system, authorize node zero (0), which is the local node.

Note

Note the following aspects concerning TCP/IP transport:

- Other vendors' implementation of TCP/IP transports may use different call interfaces. Confirm that the interface is compatible before using an alternative TCP/IP product.
- You can conserve memory and process slots by configuring DEC TCP/IP Services for OpenVMS software for the minimum DECwindows requirement to support the X protocol. DECwindows only requires that INET_ACP be running. DECwindows does not

require that the NFS server, the FTPD server, or the remote terminal servers be running.

For more information about TCP/IP concepts, see the *DEC TCP/IP Services for VMS System Management Guide*.

3.6.2 Using Other Transports with DECwindows

DECwindows Motif supports DECnet for OpenVMS, local, and TCP/IP transports. However, you can use other transports with DECwindows by writing a DECwindows transport interface layer. Third-party vendors supply additional transports, such as other TCP/IP transports.

3.6.2.1 Writing a Transport Interface

For information about writing a transport interface, see the *VMS DECwindows Transport Manual*.

The transport interface image's file name must have one of the following formats, where *name* is the unique transport name:

- DECW$TRANSPORT_*name* (for transports supplied by Digital)
- DECW_TRANSPORT_*name* (for customer-written transports)

You can then specify the *name* in the SET DISPLAY command before invoking a client application, as in the following example:

```
$ SET DISPLAY/CREATE/NODE=MYNODE/TRANSPORT=MYTRANSPORT
```

See Section 4.1 for more information about setting the display.

3.6.2.2 Installing the Transport Interface Image

Install the image as a protected image in the SYS$SHARE directory (as defined by an executive mode logical name). Digital recommends that you install the image by adding the following line to SYS$MANAGER:SYSTARTUP_VMS.COM (or the SYS$MANAGER:SYSTARTUP_V5.COM file in VMS Version 5.5–2 systems) :

```
$ INSTALL CREATE SYS$SHARE:DECW_TRANSPORT_name/OPEN/SHARED-
_$  /HEADER_RESIDENT/PROTECTED
```

You can also include this command in a startup file that SYSTARTUP_VMS.COM (SYSTARTUP_V5.COM file in VMS Version 5.5–2) calls. To use this transport with the DECwindows server, customize the server startup as described in the Section 3.1.

3.7 Determining Current Server Parameters

This section lists the commands that you can use to determine the current server settings.

3.7.1 Displaying the Server Logical Name Table

Use the following command to display all of the logical names in the server logical name table that are defined from symbols specified in the DECwindows startup process:

```
$ SHOW LOGICAL/TABLE=DECW$SERVERn_TABLE
```

Example 3–7 illustrates a typical display.

Example 3–7 Displaying the Server Logical Name Table

```
$ SHOW LOGICAL/TABLE=DECW$SERVER0_TABLE
(DECW$SERVER0_TABLE)
```

(continued on next page)

Example 3–7 (Cont.) Displaying the Server Logical Name Table

```
  "DECW$COLOR" = "TRUE"
  "DECW$DEFAULT_KEYBOARD_MAP" = "NORTH_AMERICAN_LK201LA"
  "DECW$FONT" = "DECW$SYSCOMMON:[SYSFONT.DECW.USER_CURSOR16]"❶
= "DECW$SYSCOMMON:[SYSFONT.DECW.CURSOR16]"
= "DECW$SYSCOMMON:[SYSFONT.DECW.USER_100DPI]"
= "DECW$SYSCOMMON:[SYSFONT.DECW.100DPI]"
= "DECW$SYSCOMMON:[SYSFONT.DECW.USER_75DPI]"
= "DECW$SYSCOMMON:[SYSFONT.DECW.75DPI]"
= "DECW$SYSCOMMON:[SYSFONT.DECW.USER_COMMON]"
= "DECW$SYSCOMMON:[SYSFONT.DECW.COMMON]"
  "DECW$KEYBOARD" = "TTA0:"❷
  "DECW$KEYMAP" = "DECW$SYSCOMMON:[SYS$KEYMAP.DECW.USER]"
= "DECW$SYSCOMMON:[SYS$KEYMAP.DECW.SYSTEM]"
  "DECW$MONITOR_DENSITY" = "100"
  "DECW$POINTER" = "TTA1:"❸
  "DECW$SERVER_BUG_COMPATIBILITY" = "Y"
  "DECW$SERVER_CONNECT_LOG" = "F"
  "DECW$SERVER_DISABLE_BACKING_STORE" = "F"
  "DECW$SERVER_DISABLE_CH" = "N"
  "DECW$SERVER_EXTENSIONS" = "Adobe-DPS-Extension"
= "X3D_PEX"
= "Xie"
= "DEC-XTRAP"
= "Multi-Buffering"
  "DECW$SERVER_SCREENS" = "GEA0"❹
  "DECW$SERVER_TRANSPORTS" = "DECNET"
= "LOCAL"
  "DECW$SYSCOMMON" = "SYS$SYSROOT"
  "DECW$XSIZE_IN_PIXELS" = "1280"
  "DECW$YSIZE_IN_PIXELS" = "1024"
```

❶ DECW$FONT specifies the font path. For more information, see Section 3.2.4.1.

❷ DECW$KEYBOARD is the keyboard device name.

❸ DECW$POINTER is the mouse pointer device name.

❹ DECW$SERVER_SCREENS is the list of graphics devices.

3.7.2 Using the XDPYINFO Utility

You can use the DECwindows utility XDPYINFO (X Display Information) to query the server directly and report various server parameters.

The following example shows how to invoke the XDPYINFO utility. Before executing this command, make sure you have the correct display selected by using the SET DISPLAY command. See Section 4.1 for more information about showing and setting the display.

```
$ SET DISPLAY/CREATE/NODE=node_name
$ RUN DECW$UTILS:XDPYINFO
```

The following example illustrates a typical display:

```
name of display:    _WSA1:
version number:    11.0
vendor string:    DECWINDOWS DigitalEquipmentCorp. VAX
vendor release number:    6000
maximum request size:  4096 longwords (16384 bytes)
motion buffer size:  0
bitmap unit, bit order, padding:    32, LSBFirst, 32
image byte order:    LSBFirst
number of supported pixmap formats:    4
supported pixmap formats:
    depth 1, bits_per_pixel 1, scanline_pad 32
    depth 4, bits_per_pixel 8, scanline_pad 32
    depth 8, bits_per_pixel 8, scanline_pad 32
    depth 24, bits_per_pixel 32, scanline_pad 32
keycode range:    minimum 8, maximum 255
number of extensions:    8
    MIT-SUNDRY-NONSTANDARD
    DEC-Server-Mgmt-Extension
    ServerManagementExtension
    Adobe-DPS-Extension
    X3D_PEX
    Xie
    DEC-XTRAP
    Multi-Buffering
default screen number:    0
number of screens:    1
```

```
screen #0:
  dimensions:    1280x1024 pixels (325x260 millimeters)
  resolution:    100x100 dots per inch
  depths (2):    1, 8
  root window id:    0x2a
  depth of root window:    8 planes
  number of colormaps:    minimum 1, maximum 1
  default colormap:    0x25
  default number of colormap cells:    256
  preallocated pixels:    black 0, white 1
  options:    backing-store YES, save-unders NO
  current input event mask:    0x0
  number of visuals:    4
  default visual id:  0x21
  visual:
    visual id:    0x21
    class:    PseudoColor
    depth:    8 planes
    size of colormap:    256 entries
    red, green, blue masks:    0xff, 0xff, 0xff
    significant bits in color specification:    8 bits
  visual:
    visual id:    0x22
    class:    GrayScale
    depth:    8 planes
    size of colormap:    256 entries
    red, green, blue masks:    0xffffffff, 0xffffffff, 0xffffffff
    significant bits in color specification:    8 bits
  visual:
    visual id:    0x23
    class:    StaticGray
    depth:    8 planes
    size of colormap:    2 entries
    red, green, blue masks:    0xff, 0xff, 0xff
    significant bits in color specification:    8 bits
  visual:
    visual id:    0x24
    class:    TrueColor
    depth:    8 planes
    size of colormap:    256 entries
    red, green, blue masks:    0xe0, 0x1c, 0x3
    significant bits in color specification:    3 bits
```

3.7.3 Using the XSET Utility

Use the XSET utility to query the server directly for parameter settings. Running the XSET utility is a good method to determine the current font path.

The following command shows how to invoke the XSET utility. Before executing this command, make sure you have the correct display selected by using the SET DISPLAY command. See Section 4.1 for more information about showing and setting the display.

```
$ SET DISPLAY/CREATE/NODE=node_name
$ MCR DECW$UTILS:XSET Q
```

The following example illustrates a typical display:

```
Keyboard Control:
  auto repeat:  on    key click percent:  50    LED mask:  00000000
  auto repeating keys:  0000000000000000
                        00000000f0011e20
                        0ffcfeffff0700d0
                        ffffffffffffffff
  bell percent:  50    bell pitch:  400    bell duration:  100
Pointer Control:
  acceleration:  7 = 7 / 1    threshold:  3
Screen Saver:
  prefer blanking:  no    allow exposures:  no
  timeout:  600    cycle:  600
Colors:
  default colormap:  0x25    BlackPixel:  0    WhitePixel:  1
Font Path:
  DECW$FONT:
Bug Mode: compatibility mode is enabled
```

4
Using DECwindows

This chapter includes the following information that you can use after DECwindows has started.

Topic	Section
Setting the Display	Section 4.1
Using the SHOW DISPLAY Command	Section 4.1.2
The Login Sequence	Section 4.2
Customizing the Session Manager Environment	Section 4.3
Creating a Custom Bookreader Directory	Section 4.3.1
Changing Your Logo	Section 4.3.2
Creating Dedicated Accounts	Section 4.3.3
Modifying Session Manager Behavior	Section 4.4
Modifying System Resource Files	Section 4.5
Customizing Print Formats	Section 4.6

4.1 Setting the Display

When a client application starts up, it opens one or more connections to a display server. The display server can be on the local machine or on a remote node. The application needs to tell the underlying toolkit, Intrinsics, or Xlib routines the node name and other connection information. This information is contained in a string called the **display name**.

The application can explicitly pass the display name to the underlying routines. Usually, though, the application passes a null display name to the underlying routines. When Xlib opens the display, it translates the logical name DECW$DISPLAY to find the display name. If DECW$DISPLAY is undefined, then Xlib translates the logical name SYS$OUTPUT.

The display name can be in one of the following forms. The brackets ([ ]) represent optional syntax and are not included as part of the display name.

node::*server*[.*screen*]
node:*server*[.*screen*]
WSA*n*:

node	Specifies the name of the node on which the display server is running. If the node name is followed by two colons, then Xlib uses the DECnet transport to establish the connection to the display server. If the node name is followed by one colon, then Xlib will use the TCP/IP transport. Note that the single-colon syntax is not supported in versions of DECwindows earlier than DECwindows Motif Version 1.2 for OpenVMS.
server	Specifies the (decimal) number of the display server on the host machine. A single CPU can have more than one display. Multiple displays are usually numbered starting with zero.
screen	Optional. Specifies, as a decimal number, the screen to be used on that server. A single server can display on multiple screens, and screens are numbered starting with zero.
WSA*n*:	Specifies the device name of a WS device that contains the transport, node, server number, and screen number.

Normally, the display is either inherited from a parent process or specified using the DCL command SET DISPLAY. The SET DISPLAY command creates or modifies a WSA*n*: device and sets the transport, node name, server number, and screen number.

SET DISPLAY also defines the DECW$DISPLAY logical name to point to this device. When Xlib sees that DECW$DISPLAY points to a WSA*n*: device, Xlib reads the transport, node name, and other information just as if the application had specified them explicitly.

4.1.1 Examples for Setting the Display

The following examples illustrate several methods to set the display.

Example 1

Either of the following two command sequences can be used to run the Clock application displaying to node DOG using the default server and screen numbers and using the DECnet transport.

```
$ DEFINE DECW$DISPLAY DOG::0
$ RUN SYS$SYSTEM:DECW$CLOCK
```

```
$ SET DISPLAY/CREATE/NODE=DOG
$ RUN SYS$SYSTEM:DECW$CLOCK
```

Note that in the SET DISPLAY command the transport defaults to DECnet, and the server and screen numbers both default to zero. The /CREATE qualifier means that a new WSA*n*: device will be created.

Example 2

Either of the following two command sequences can be used to run the Clock application on node DOG using screen 1 and using the TCP/IP transport:

```
$ DEFINE DECW$DISPLAY DOG:0.1
$ RUN SYS$SYSTEM:DECW$CLOCK
```

```
$ SET DISPLAY/TRANSPORT=TCPIP/SCREEN=1
$ RUN SYS$SYSTEM:DECW$CLOCK
```

The SET DISPLAY command assumes that the WSA*n*: device has already been created and the node name has been set to DOG.

Example 3

The following example runs Clock in a detached process and displays the output on the local workstation display:

```
$ SET DISPLAY/CREATE/TRANSPORT=LOCAL
$ RUN/DETACH/OUTPUT='F$TRNLNM("DECW$DISPLAY")' -
_$ SYS$SYSTEM:DECW$CLOCK
```

This method takes advantage of the fact that Xlib translates the logical name SYS$OUTPUT to find the display name if DECW$DISPLAY is undefined. Xlib creates a WSA*n*: device using the SET DISPLAY command and then translates the DECW$DISPLAY logical name to find the device name of the WS device. Xlib passes this device name as the SYS$OUTPUT of the detached process.

The WS device is similar to the NL device in that QIO reads from the device return end-of-file and QIO writes to the device are ignored.

Example 4

A somewhat more complicated implementation of Example 3 takes advantage of a feature of LOGINOUT.EXE.

If SYS$ERROR points to a WS device, LOGINOUT defines DECW$DISPLAY to be the translation of SYS$ERROR and then redefines SYS$ERROR to be the translation of SYS$OUTPUT. This allows error messages from the detached process to be written to a log file.

```
$ SET DISPLAY/CREATE/TRANSPORT=LOCAL
$ CREATE TEST.COM
$ RUN SYS$SYSTEM:DECW$CLOCK
$ RUN/DETACH/ERROR='F$TRNLNM("DECW$DISPLAY")' -
_$ /INPUT=TEST.COM/OUTPUT=TEST.LOG -
_$ SYS$SYSTEM:LOGINOUT
```

4.1.2 Using the SHOW DISPLAY Command

You can display information from a WS device by using the SHOW DISPLAY command. Example 1 shows the command sequence.

Example 1

```
$ SET DISPLAY/CREATE
$ SHOW DISPLAY

Device:    WSA9:  [super]
Node:      0
Transport: DECNET
Server:    0
Screen:    0
```

You can use the /SYMBOL qualifier to allow the information from the SHOW DISPLAY command to be stored in DCL symbols, as shown in Example 2.

Example 2

```
$ SHOW DISPLAY/SYMBOL
$ SHOW SYMBOL DECW$*

DECW$DISPLAY_NODE == "0"
DECW$DISPLAY_SCREEN == "0"
DECW$DISPLAY_SERVER == "0"
DECW$DISPLAY_TRANSPORT == "DECNET"
```

In Example 1, WSA9: was created in supervisor mode.

A user with SYSNAM privilege can also create a display in executive mode. A supervisor-mode WS device is deleted when the last channel to it is deassigned, and when the creating process is logged out. However, an executive-mode WS device is not deleted unless the user explicitly deletes it with the SET DISPLAY/DELETE command.

Example 3 shows how a user with SYSNAM privilege can create a WS device in executive mode.

Example 3

```
$ SET DISPLAY/CREATE/EXEC
$ SHOW DISPLAY

      Device:    WSA10:  [exec]
      Node:      0
      Transport: DECNET
      Server:    0
      Screen:    0
```

When DECwindows is started after a system reboot, the command procedure DECW$STARTAPPS.COM creates an executive mode WS device (initially WSA1:), which is passed to the DECwindows login procedure to create the Start Session display. When the user logs in, this same WS device is used to create Session Manager.

When Session Manager starts an application, the WS device is passed to the application using the DECW$DISPLAY mechanisms described in Section 4.1. An example of this is a DECTERM window created from Session Manager (assuming that no SET DISPLAY commands have been issued in that window).

Note that a DECwindows application started from that DECTERM window displays on the local node without the user having to issue a SET DISPLAY command.

4.2 The Login Sequence

This section describes the DECwindows login process that begins when the Start Session dialog box appears to after Session Manager starts.

When you enter a correct user name and password in the Start Session dialog box, the following events occur:

1. **LOGINOUT.EXE calls routines within DECW$LOGINOUT.EXE to perform the following functions:**

   a. Checks the DW-MOTIF license to verify that the system is licensed to open connections to the display server. If not, DECW$LOGINOUT.EXE displays a warning message, restarts itself, and exits. A new Start Session dialog box displays.

   b. Changes the input pointer to a watch cursor.

   c. Modifies the security list to allow you to run applications that connect to the display server.

   d. Starts a process that executes DECW$WAITFORSM.EXE, which holds a connection to the server during the startup process.

   e. Sets the process name to DECW$SESSION.

   f. Intializes the Start Session process's accumulated CPU time to zero so that the user process is not charged for time spent displaying the Start Session dialog box.

   g. Passes SYS$MANAGER:DECW$STARTSM.COM as the command procedure that DCL executes.

   h. Exits. DCL takes control of the process.

2. **SYS$MANAGER:DECW$STARTSM.COM executes as Session Manager's default DCL command procedure. It performs the following functions:**

   a. If the logical name DECW$SM_WSQUOTA is defined, sets the Session Manager process's working set quota to the value of the logical name.

   b. Executes the system login command procedure as determined by the SYS$SYLOGIN logical name. By default, this is SYLOGIN.COM.

   c. If you did not specify /NOCOMMAND after the user name in the Start Session dialog box, executes your LOGIN.COM (or other login command procedure as specified in the UAF record). You can redefine the DECW$USER_DEFAULTS logical name in your LOGIN.COM to change the directory that is used for your resource files.

d. Executes SYS$SYSTEM:DECW$WSINIT.EXE to read your workstation customization files. This program opens a connection to the display server and makes Xlib calls to apply your customizations, setting the screen background, pointer shape and color, and other workstation settings. It also creates a property on the root window to communicate your customizations to DECwindows applications that are started later.

Note

If you are logging in to a DECwindows Motif system for the first time after having used the XUI version of DECwindows, DECW$WSINIT.EXE will read your XUI resource files and convert them to DECwindows Motif format.

e. Executes SYS$MANAGER:DECW$SYLOGIN.COM, if it exists.

f. Executes SYS$LOGIN:DECW$LOGIN.COM, if it exists.

g. Runs SYS$SYSTEM:DECW$SESSION.EXE to start Session Manager.

4.3 Customizing the Session Manager Environment

DECwindows Motif provides a template file that you can modify to customize the login sequence. The file is named:

```
SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.TEMPLATE
```

By using this file, you can modify the client areas listed in Table 4–1.

Table 4–1 Client Areas That Can Be Modified

Client Area	Global Symbol	Default
Directory used by Bookreader to find online books	DECW$BOOK	SYS$SYSROOT:[DECW$BOOK]
Directory for FileView command files	VUE$LIBRARY	DECW$SYSCOMMON: [VUE$LIBRARY.USER], DECW$SYSCOMMON: [VUE$LIBRARY.SYSTEM]
Directory for keymap files	DECW$KEYMAP	DECW$SYSCOMMON: [SYS$KEYMAP.DECW.USER], DECW$SYSCOMMON: [SYS$KEYMAP.DECW.SYSTEM]
Directory for DECwindows example programs	DECW$EXAMPLES	SYS$SYSROOT: [SYSHLP.EXAMPLES.DECW]
Directory for XDPS example programs	XDPS$EXAMPLES	SYS$SYSROOT: [SYSHLP.EXAMPLES.XDPS]
DCL command executed when DECwindows startup is complete	DECW$MAINAPP	Run SYS$SYSTEM: DECW$STARTLOGIN
Login logo	DECW$LOGINLOGO	None
Parameters to the SET DISPLAY command	DECW$APPSNODE DECW$APPSSERVER DECW$APPSSCREEN	Node 0 Server 0 Screen 0
Command procedure that runs after user authorization is complete	DECW$SESSIONCOM	SYS$MANAGER: DECW$STARTSM.COM
DECwindows SYLOGIN command procedure	DECW$SYLOGINCOM	None
Command that initializes the workstation with the correct user resource values	DECW$SESSIONINIT	Run SYS$SYSTEM: DECW$WSINIT
Command that runs a session	DECW$SESSIONMAIN	Run SYS$SYSTEM: DECW$SESSION
Command that resets a session	DECW$SESSIONEND	None
Log file for DECwindows LOGINOUT	DECW$LOGINLOG	None

(continued on next page)

Table 4–1 (Cont.) Client Areas That Can Be Modified

Client Area	Global Symbol	Default
Use of multiple connections during loginout	DECW$LOGINMANY	False
Log file for Session Manager can be turned off	DECW$SESSIONLOG	Log file on
Directory for computer-based instruction files	DECW$CBI	SYS$SYSROOT: [SYSCBI.DECW$CBI]
Directory where FileView creates public profile files	VUE$LIBRARY_WRITE	SYS$COMMON: [VUE$LIBRARY.USER]
Directory for utilities	DECW$UTILS	SYS$SYSROOT:[SYSHLP. EXAMPLES.DECW.UTILS]
Login logo can be run in a subprocess	DECW$LOGINLOGOSUB	False (detached process)
Define logical names for UNIX compatibility	DECW$UNIXLOGICALS	True
Require login or pause to perform snapshot	DECW$SNAPSHOT_ SECURITY	True

To customize any of these areas, copy the SYS$MANAGER: DECW$PRIVATE_APPS_SETUP.TEMPLATE file to SYS$MANAGER: DECW$PRIVATE_APPS_SETUP.COM, as in the following example:

```
$ COPY SYS$COMMON:[SYSMGR]DECW$PRIVATE_APPS_SETUP.TEMPLATE -
_$ SYS$COMMON:[SYSMGR]DECW$PRIVATE_APPS_SETUP.COM/LOG
```

Then edit the file to change the appropriate symbol.

The remaining sections in this chapter contain examples for customizing the DECwindows client environment using the template file.

This file contains two sections, a Cluster Common section, and a Cluster Member Workstation-Specific section. Add the new symbol to the appropriate section.

4.3.1 Creating a Custom Bookreader Directory

When you start Bookreader, the system uses the logical name DECW$BOOK to locate the file LIBRARY.DECW$BOOKSHELF. By default, the logical name DECW$BOOK equates to SYS$SYSROOT:[DECW$BOOK]. You can create a private directory containing the file LIBRARY.DECW$BOOKSHELF.

To do this, you need to create the directory and then define a global symbol in the DECW$PRIVATE_APPS_SETUP.COM file.

To create a private Bookreader directory called SYS$COMMON:[DECW$BOOK_LOCAL], you need to:

1. Create the directory, giving the world read access as in the following example:

```
$ CREATE/DIRECTORY/PROTECTION=WORLD:R -
_$ SYS$COMMON:[DECW$BOOK_LOCAL]
```

2. Copy the Bookreader LIBRARY.DECW$BOOKSHELF file to the new directory.

```
$ COPY SYS$SYSROOT:[DECW$BOOK]LIBRARY.DECW$BOOKSHELF -
_$ SYS$COMMON:[DECW$BOOK_LOCAL]/LOG
```

3. If the SYS$MANAGER directory does not contain the DECW$PRIVATE_APPS_SETUP.COM command file, copy the template file to a command file.

```
$ COPY SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.TEMPLATE -
_$ SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.COM/LOG
```

4. Edit the file and define the global symbol DECW$BOOK to point to your personal Bookreader directory.

   Add the following line:

```
$ DECW$BOOK == "SYS$COMMON:[DECW$BOOK_LOCAL]"
```

5. Restart DECwindows Motif using the following command:

```
$ @SYS$MANAGER:DECW$STARTUP RESTART
```

4.3.2 Changing Your Logo

You must use the Start Session screen for logging in to DECwindows, but you can substitute your own logo for the Digital logo by running an application that displays your logo.

To do this, you need to define a global symbol in the DECW$PRIVATE_APPS_SETUP.COM file and create a DCL command file that contains the commands to display your logo.

To display your own logo on the login screen, you need to:

- Create a command file (SYS$MANAGER:LOGO.COM, for example).
- Define the global symbol DECW$LOGINLOGO in SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.COM, with its value set to the name of your command file. Note that this command file is run by the system account.

To create a custom logo, execute the following steps:

1. If the SYS$MANAGER directory does not contain a command file, copy the template file to a command file:

   ```
   $ COPY SYS$COMMON:[SYSMGR]DECW$PRIVATE_APPS_SETUP.TEMPLATE -
   _$ SYS$SPECIFIC:[SYSMGR]DECW$PRIVATE_APPS_SETUP.COM/LOG
   ```

2. Edit the file and define the symbol DECW$LOGINLOGO to point to the command file that displays your custom logo.

   Add the following line:

   ```
   $ DECW$LOGINLOGO == "SYS$MANAGER:MYLOGO.COM"
   ```

3. Create the command file that displays your logo. This needs to be the file name and directory specified in step 2. Note that the command file runs as a detached process under the SYSTEM account. Be careful not to give users access to the system through your custom logo. Do not run applications like FileView or DECterm as a custom logo.

   For example, the file SYS$MANAGER:MYLOGO.COM could contain this command to display the icosahedron that is shipped as an example program:

   ```
   $ RUN DECW$EXAMPLES:ICO
   ```

4. Restart DECwindows by running DECW$STARTUP, or reboot your workstation. Ending the session does not display the logo unless you execute the DECW$STARTUP command procedure that starts DECwindows.

   ```
   $ @SYS$STARTUP:DECW$STARTUP RESTART
   ```

   Caution

   Restarting the DECwindows server disconnects all current client processes.

   To restore the Digital logo, you need to remove the definition of DECW$LOGINLOGO from your DECW$PRIVATE_APPS_SETUP.COM file and restart DECwindows.

4.3.3 Creating Dedicated Accounts

If you are a system manager, you can set up user accounts to run only certain applications. For example, you might want users to use only the Bookreader application.

This section describes the following three methods for setting up dedicated DECwindows accounts:

- Modifying the Session Manager command procedure
- Modifying the Session Manager executable file
- Modifying the Session Manager profile file

4.3.3.1 Modifying the Session Manager Command Procedure

You can define the global symbol DECW$SESSIONCOM in the SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.COM file so that when a user logs in, LOGINOUT runs a command file other than DECW$STARTSM.COM. This action bypasses Session Manager altogether.

In Example 4–1, DECW$SESSIONCOM is defined in DECW$PRIVATE_APPS_SETUP.COM to point to the private command procedure SYS$MANAGER:PRIVATE_SESSIONCOM.COM. The default command file is SYS$MANAGER:DECW$STARTSM.COM.

Because DECW$SESSIONCOM is defined in the system logical name table, it affects Session Manager startup for every user name. Your private command file must check the user name under which it is running and run the normal DECW$STARTSM.COM procedure for nondedicated accounts.

Note

With this method, the following startup command procedures (normally executed from DECW$STARTSM.COM) do not run: SYLOGIN.COM, LOGIN.COM, DECW$SYLOGIN.COM, and DECW$LOGIN.COM.

To modify the Session Manager command procedure:

1. Create the command procedure shown in Example 4–1.

Example 4–1 Using a Private Session Manager Command File

```
$ CREATE SYS$MANAGER:PRIVATE_SESSIONCOM.COM
$!
$! Check to see whether this is the dedicated DECwindows account.
$! If it isn't, run the normal procedure to start Session Manager.
$!
$ username = F$USER()
$ IF F$LOCATE("BOOK_READER", username) .NE. F$LENGTH(username) -
 THEN GOTO dedicated
$!
$! Normal, nondedicated DECwindows login
$!
$ @SYS$MANAGER:DECW$STARTSM
$ EXIT
$!
```

(continued on next page)

Example 4–1 (Cont.) Using a Private Session Manager Command File

```
$! Dedicated DECwindows login.  Run Bookreader and then exit.
$!
$ dedicated:
$!
$ RUN SYS$SYSTEM:DECW$WSINIT
$ display = F$TRNLNM("DECW$DISPLAY")
$ RUN/DETACHED/OUTPUT='display' SYS$SYSTEM:DECW$MWM  ❶
$ RUN SYS$SYSTEM:DECW$BOOKREADER
$ endsession := $DECW$ENDSESSION ❷
$ endsession -noprompt ❸
$ STOP/ID=0  ❹
```

❶ DECW$MWM is the Motif Window Manager. Its SYS$OUTPUT definition needs to be set to the translation of DECW$DISPLAY so that DECW$DISPLAY is defined correctly in the detached process.

❷ DECW$ENDSESSION resets the server and displays a Start Session dialog login box.

❸ The -noprompt qualifier prevents DECW$ENDSESSION from asking the user to confirm whether to end the session. (If the user cancels the End Session operation, the workstation will be unusable because Bookreader will exit and the login box will not be displayed.)

❹ STOP/ID=0 logs out the process without writing an error message to SYS$OUTPUT.

2. Define the DECW$SESSIONCOM symbol in DECW$PRIVATE_APPS_SETUP.COM as follows:

   ```
   $ DECW$SESSIONCOM :== SYS$MANAGER:PRIVATE_SESSIONCOM.COM
   ```

   If SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.COM does not already exist, create it as follows from the template file:

   ```
   $ COPY SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.TEMPLATE -
   _$ SYS$COMMON:[SYSMGR]DECW$PRIVATE_APPS_SETUP.COM
   ```

3. Restart DECwindows by entering the following command:

   ```
   $ @SYS$MANAGER:DECW$STARTUP RESTART
   ```

This step is necessary for the changes to DECW$PRIVATE_APPS_SETUP.COM to take effect.

4. Log in to the dedicated account on a DECwindows system.

   After entering the user name and password of the dedicated account, Bookreader starts and Session Manager is not displayed. When the user exits from Bookreader, the PRIVATE_SESSIONCOM.COM procedure ends the session operation and then logs out. The End Session command displays the Start Session screen.

4.3.3.2 Modifying the Session Manager Executable File

You can define DECW$SESSIONMAIN in the SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.COM file so that DECW$STARTSM.COM runs the specified command file instead of DECW$SESSION.EXE. Session Manager reads resource files and executes login command files but does not display the Session Manager menu bar. With this method, ensure that the End Session command does not prompt for confirmation.

When DECW$STARTSM.COM starts (that is, if DECW$SESSIONCOM has its default value), it starts Session Manager by executing the DCL command that is stored in the logical name
DECW$SESSIONMAIN. To define this logical name, edit SYS$MANAGER:
PRIVATE_APPS_SETUP.COM so that it defines the global symbol DECW$SESSIONMAIN as the DCL command to execute.

This method is similar to the one described in the Section 4.3.3.1 except that, with this method, DECW$STARTSM.COM executes the SYLOGIN.COM, LOGIN.COM, DECW$SYLOGIN.COM, and DECW$LOGIN.COM command procedures. Also, DECW$STARTSM.COM executes DECW$WSINIT.EXE, so the private command procedure does not need to do so.

To modify the Session Manager executable file:

1. Create the command procedure shown in Example 4–2.

Example 4–2 Modifying the Session Manager Executable File

```
$ CREATE SYS$MANAGER:PRIVATE_SESSIONMAIN.COM
$!
$! Check to see whether this is the dedicated DECwindows account.
$! If it isn't, run the normal procedure to start Session Manager.
$!
$ username = F$USER()
$ IF F$LOCATE("BOOK_READER", USERNAME) .NE. F$LENGTH(USERNAME) -
 THEN GOTO dedicated
$!
$! Normal, nondedicated DECwindows login
$!
$ RUN SYS$SYSTEM:DECW$SESSION
$ EXIT
$!
$! Dedicated DECwindows login.  Run Bookreader and then exit.
$!
$ dedicated:
$!
$ display = F$TRNLNM("DECW$DISPLAY")
$ RUN/DETACHED/OUTPUT='display' SYS$SYSTEM:DECW$MWM    ❶
$ RUN SYS$SYSTEM:DECW$BOOKREADER
$ endsession := $DECW$ENDSESSION    ❷
$ endsession -noprompt    ❸
$ STOP/ID=0    ❹
```

❶ DECW$MWM is the Motif Window Manager. Its SYS$OUTPUT definition needs to be set to the translation of DECW$DISPLAY so that DECW$DISPLAY is defined correctly in the detached process.

❷ DECW$ENDSESSION resets the server and displays a new Start Session screen.

❸ The -noprompt qualifier prevents DECW$ENDSESSION from displaying the End Session prompt. (If the user cancels the End Session prompt, the workstation will be unusable because Bookreader will exit and the Start Session screen will not be displayed.)

❹ STOP/ID=0 logs out the process without writing a message to SYS$OUTPUT.

2. Define the DECW$SESSIONMAIN symbol in DECW$PRIVATE_APPS_SETUP.COM.

Edit this command procedure to define DECW$SESSIONMAIN. Note that the symbol must be defined as a DCL command (unlike DECW$SESSIONCOM, which is defined as the name of a command procedure).

```
$ DECW$SESSIONMAIN :== @SYS$MANAGER:PRIVATE_SESSIONMAIN.COM
```

If SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.COM does not already exist, create it as follows from the template file:

```
$ COPY SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.TEMPLATE -
_$  SYS$COMMON:[SYSMGR]DECW$PRIVATE_APPS_SETUP.COM
```

3. Restart DECwindows by entering the following command:

   ```
   $ @SYS$MANAGER:DECW$STARTUP RESTART
   ```

   This step is necessary for the changes to DECW$PRIVATE_APPS_SETUP.COM to take effect.

4. Log in to the dedicated account on a DECwindows system.

After entering the user name and password of the dedicated account, Bookreader starts and the Session Manager menu bar is not displayed. When the user exits Bookreader, the PRIVATE_SESSIONMAIN.COM procedure performs an End Session operation and then logs out. The End Session command displays a DECwindows Start Session screen.

4.3.3.3 Modifying the Session Manager Profile File

This method starts Session Manager normally, but the system manager customizes its menus to remove any applications that are not started automatically. By removing certain applications, you can limit user access to applications.

With this method, the Session Manager menu bar is displayed, and the user can interact with whatever dialog boxes the system manager does not disable.

To modify the Session Manager profile file:

1. Log in to the dedicated account on a DECwindows system.

   When you enter the user name and password of the dedicated account, Session Manager starts.

2. Add the applications to the automatic startup list that you want to be displayed when a user logs in.

Choose Automatic Startup... from the Session Manager Options menu. Session Manager displays the Automatic Startup dialog box.

Use this dialog box to remove any undesired applications, such as FileView, from the automatic startup list. (For more information about using the Automatic Startup dialog box, see *Using DECwindows Motif for OpenVMS*.)

Note

Do not remove Window Manager from the automatic startup list.

For this example add Bookreader to the list and click on OK.

3. Remove Applications from the Session Manager menu bar.

   Choose Menu Bar... from the Session Manager Options menu. Session Manager displays the Menu Bar dialog box.

   Use this dialog box to remove the applications you do not want to be displayed. (For more information about using the Menus dialog box, see *Using DECwindows Motif for OpenVMS*.)

   Click on OK to save your changes and to dismiss the dialog box.

4. Choose Save Session Manager from the Session Manager Options menu to save your menu bar settings.

5. Remove any menu items that you do not want to be displayed on the Options menu.

   Choose Menus... from the Session Manager Options menu and remove the following menu items from the Session Manager Options menu:

   Automatic Startup...
   Menus...
   Menu Bar...
   Save Session Manager

   You should also remove any other menu items that you do not want users to have access to in the dedicated account, such as Security.

   After removing the items, click on OK to dismiss the dialog box and to save your menu settings.

6. Choose End Session from the Session Manager Session menu.

   Session Manager logs you out of the dedicated account and displays the DECwindows login box.

7. Log back in to the dedicated account.

To test your changes, log in to the dedicated account from the DECwindows login box. Bookreader and Session Manager should start up by default, and you should not be able to start any other application. To log out, choose End Session from the Session Manager menu.

If you decide to make the account nondedicated again, or if you want to make changes in a dialog box that you have removed from the system menu bar, delete or rename the file VUE$PROFILE.VUE$DAT in the dedicated account's SYS$LOGIN directory.

4.4 Modifying Session Manager Behavior

You can modify the behavior of the DECwindows Session Manager by using several logical names. These logicals are not normally used; however, they may help in special circumstances, such as running two Session Managers on the same system. Table 4–2 lists the logicals and their meaning.

Table 4–2 Session Manager Logicals

Logical Name	Meaning
DECW$VUENOAUTOSTART	Prevents applications in the Session Manager autostart list from being started.
DECW$VUENORESET	Prevents Session Manager from resetting the display server when Session Manager exits.

(continued on next page)

Table 4–2 (Cont.) Session Manager Logicals

Logical Name	Meaning
DECW$VUELOGINOUTPUT	Saves any output from the login command files (SYSLOGIN, LOGIN, DECW$LOGIN) as applications start up from Session Manager. You can access this information by clicking on the SHOW OUTPUT button in the Work in Progress dialog box, which is located in the Session menu in the Session Manager window.

You can define these logicals in a DECterm or in the login command files (SYSLOGIN, LOGIN, DECW$LOGIN). To set any of these logical, define the logical name to any value, as in the following example:

```
$ DEFINE DECW$VUENOAUTOSTART 1
```

4.5 Modifying System Resource Files

Most DECwindows applications provide dialog boxes that contain options for customizing your DECwindows environment and saving your settings. For example, using Session Manager Options menu, you can look at many predefined settings and then choose and save new settings.

However, not all applications provide dialog boxes for changing and saving settings. For example, some applications let you change and save the size or location of the application's main window and others do not. To change and save settings that are not available from dialog boxes, you can specify resources in a DECW$XDEFAULTS.DAT file.

Caution

Use extreme caution when modifying resources for the following reasons:

- Options dialog boxes cannot be used to change settings that are specified in the DECW$XDEFAULTS.DAT file.

- In future DECwindows releases, application dialog boxes might contain the options you have added to your Xdefaults file. You will have to modify your DECW$XDEFAULTS.DAT file to reflect these changes.
- Error handling for resources is not available in this version of DECwindows. Syntax errors in resource files might cause DECwindows to fail without providing error messages.
- Using a large DECW$XDEFAULTS.DAT file significantly degrades application startup performance.

To use resources to change application settings, create a file (DECW$XDEFAULTS.DAT) in the directory specified by the logical name DECW$USER_DEFAULTS. In this file, specify the resources for the application settings you want to change. Alternatively, if your changes are specific to one application, you can modify that application's resource file (for example, DECW$MAIL.DAT).

Note that each resource specification in the DECW$XDEFAULTS.DAT file follows explicit syntax rules. For more information about the resource syntax, see the *VMS DECwindows Guide to Xlib (Release 4) Programming: MIT C Binding* or *X Window System*.

See *Using DECwindows Motif for OpenVMS* for resource information for the following applications:

- Bookreader
- Calculator
- Calendar
- Cardfiler
- CDA Viewer
- Clock
- DECterm
- FileView
- Session Manager
- LinkWorks Manager

- Mail
- Motif Window Manager

4.6 Customizing Print Formats

This section describes how to define print formats and lists the logical names and associated print formats for DECwindows Motif software. Depending on your configuration, DECwindows Motif customization tasks can include defining logical names to specify print formats.

Before you start up the system, edit the OpenVMS startup procedure to define the logical names that associate print queues with print formats. This startup procedure will subsequently call the DECwindows Motif startup procedure. You can also add these logicals to your login command file.

4.6.1 Defining Print Formats

Many DECwindows Motif applications use the Print dialog box to queue files or screens to a printer. By default, all printing devices on the system are displayed in the Printer list box. However, print queues can be associated with print formats through the definition of logical names. You define the logical name in the OpenVMS startup command procedure before the call to the DECwindows Motif startup procedure. Separate the list of print queues with commas or spaces, with the first queue being the default choice.

You can customize the list of printer queues displayed in the Print dialog box by defining any of the logical names in Table 4–3. This method is faster than making the Print dialog box derive the names of all the queues on the system, most of which do not apply to the print format under consideration. The following example shows how to define logical names for print formats in the OpenVMS startup files:

```
$ Define DECW$PRINTER_FORMAT_TEXT   "CLUSTER_LN03,CLUSTER_PRINT, -
_$ ANSI_ARTWRK,ANSI_PROTON"
$ Define DECW$PRINTER_FORMAT_LINE   "CLUSTER_PRINT"
$ Define DECW$PRINTER_FORMAT_ANSI2  "CLUSTER_LN03,ANSI_ARTWRK,ANSI2_PROTON"
$ Define DECW$PRINTER_FORMAT_ANSI   "CLUSTER_LN03,ANSI_ARTWRK,ANSI_PROTON"
$ Define DECW$PRINTER_FORMAT_PS     "PS_ARTWRK,PS_PROTON"
$ Define DECW$PRINTER_FORMAT_REGIS  "SYS$NULL"
```

For example, if DECW$PRINTER_FORMAT_ANSI2 is defined as CLUSTER_LN03, ANSI_ARTWRK, ANSI2_PROTON, then when you select ANSI2 from the Print Format list box, only CLUSTER_LN03, ANSI_ARTWRK, and ANSI2_PROTON are shown in the Printer list box, with CLUSTER_LN03 being the default choice.

4.6.2 Logical Names and Print Formats

Table 4–3 lists the logical names and the associated print formats in the DECwindows Motif product.

Table 4–3 Logical Names for the Print Dialog Box

Logical Name	Print Format
DECW$PRINTER_FORMAT_DEFAULT	Default
DECW$PRINTER_FORMAT_TEXT	Text
DECW$PRINTER_FORMAT_LINE	Line printer
DECW$PRINTER_FORMAT_TERM	Terminal
DECW$PRINTER_FORMAT_ANSI2	ANSI2
DECW$PRINTER_FORMAT_ANSI	ANSI
DECW$PRINTER_FORMAT_PS	PostScript
DECW$PRINTER_FORMAT_REGIS	ReGIS
DECW$PRINTER_FORMAT_TEK	TEKTRONIX
DECW$PRINTER_FORMAT_DDIF	DDIF

You can define the logical names in systemwide logical name tables and users can override the logicals when necessary.

A

Tuning the DECwindows System

This appendix lists the recommended values for quotas and system parameters for DECwindows Motif systems.

A.1 Establishing UAF Parameters for DECwindows Applications

DECwindows applications can be sensitive to user authorization file (UAF) limits. If problems occur while starting a session or while starting applications, or if an application disappears without an error message, check the UAF limits of the account under which the session was started.

Table A–1 describes suggested UAF limits. The specific numbers given are intended only as guidelines. The correct UAF parameters depend on your applications and processes. For more information see the *HP OpenVMS System Manager's Manual*.

Table A–1 Recommended Settings for UAF Limits

Limit	VAX Setting	AXP Setting	Usage
ASTLM	100	100	Asynchronous system trap (AST) queue limit
BIOLM	100	100	Buffered I/O count limit
BYTLM	40000	100000	I/O byte limit
DIOLM	100	100	Direct I/O count limit
ENQLM	200	300	Enqueue lock limit

(continued on next page)

Table A–1 (Cont.) Recommended Settings for UAF Limits

Limit	VAX Setting	AXP Setting	Usage
FILLM	100	100	Open file limit
MAXDETACH	0	0	Maximum detached processes for a single user name (0 = no limit)
MAXJOBS	0	0	Maximum active processes for a single user name (0 = no limit)
MAXACCTJOB	0	0	Maximum active processes for a single account (0 = no limit)
PGFLQUOTA	32768	32768	Paging file limit
PRCLM	8	10	Subprocess creation limit
WSDEFAULT	512	1024	Default working set size
WSEXTENT	2048	8192	Working set extent
WSQUOTA	1024	2048	Working set quota

A.2 Establishing System Parameters for DECwindows Applications and Display Server

Because most DECwindows processes, especially DECterm windows, run as detached processes, you need to increase certain system parameters.

Table A–2 lists these parameters, their minimum settings, add values, and usage for VAX systems. Table A–3 lists these parameters, their minimum settings, add values, and usage for AXP systems. An **add value** is the increment by which the system increases a parameter until it reaches the minimum amount.

A.2 Establishing System Parameters for DECwindows Applications and Display Server

See the *HP OpenVMS System Management Utilities Reference Manual* for information about modifying system parameters.

Table A–2 Recommended Settings for System Parameters on VAX Systems

Parameter	Minimum	Add Value	Description
SYSMWCNT		195	System working set limit
GBLSECTIONS	420	130	System global sections
GBLPAGES	62000	24000	System global page table entries
GBLPAGFIL	6024	5000	System global page-file sections limit
NPAGEDYN	735000	305000	System nonpaged dynamic pool
PAGEDYN	274000	64000	System paged dynamic pool
CHANNELCNT	255	–	System permanent I/O channel limit
SWPOUTPGCNT	500	–	Minimum process page size before swapping
VIRTUALPAGECNT	50000	–	Process virtual page limit
PROCSECTCNT	40	–	Process image-section descriptor limit

(continued on next page)

Table A–2 (Cont.) Recommended Settings for System Parameters on VAX Systems

Parameter	Minimum	Add Value	Description
WSMAX	4096	–	Process working set maximum
PQL_DPGFLQUOTA	32768	–	Default page file quota
PQL_MASTLM	100	–	Minimum AST limit
PQL_MBIOLM	100	–	Minimum buffered I/O limit
PQL_MDIOLM	100	–	Minimum direct I/O limit
PQL_MPRCLM	8	–	Minimum subprocess limit
PQL_MFILLM	100	–	Minimum open file limit
PQL_MBYTLM	40000	–	Minimum buffered I/O byte limit
PQL_MENQLM	200	–	Minimum enqueued lock limit
PQL_MWSDEFAULT	512	–	Minimum working set default
PQL_MWSQUOTA	1024	–	Minimum working set quota
PQL_MWSEXTENT	2048	–	Minimum working set extent

Table A–3 lists the parameters, minimum settings, add values, and usage for AXP systems.

A.2 Establishing System Parameters for DECwindows Applications and Display Server

Table A–3 Recommended Settings for System Parameters on AXP Systems

Parameter	Minimum	Add Value	Description
GBLSECTIONS	600	180	System global sections
GBLPAGES	80000	44000	System global page table entries
GBLPAGFIL	1024	768	System global page-file sections limit
NPAGEDYN	670000	150000	System nonpaged dynamic pool
PAGEDYN	289000	76000	System paged dynamic pool
CHANNELCNT	255	–	System permanent I/O channel limit
SWPOUTPGCNT	512	–	Minimum process page size before swapping
VIRTUALPAGECNT	73536[1]	–	Process virtual page limit
PROCSECTCNT	64	–	Process image section descriptor limit
WSMAX	12000	–	Process working set maximum
PQL_MPGFLQUOTA	32768	–	Minimum page file quota

[1]Requirements may vary depending on your graphics hardware.

(continued on next page)

Table A–3 (Cont.) Recommended Settings for System Parameters on AXP Systems

Parameter	Minimum	Add Value	Description
PQL_MASTLM	100	–	Minimum AST limit
PQL_MBIOLM	100	–	Minimum buffered I/O limit
PQL_MDIOLM	100	–	Minimum direct I/O limit
PQL_MPRCLM	10	–	Minimum subprocess limit
PQL_MFILLM	100	–	Minimum open file limit
PQL_MBYTLM	100000	–	Minimum buffered I/O byte limit
PQL_MENQLM	300	–	Minimum enqueued lock limit
PQL_MWSDEFAULT	1024	–	minimum working set default
PQL_MWSQUOTA	2048	–	Minimum working set quota
PQL_MWSEXTENT	8192	–	Minimum working set extent

B

DECwindows Motif Keymap Names

Table B–1 lists the DECwindows Motif keymap names. The table is arranged based on the language for which each keyboard is designed. Choose the keymap ending in _DP for the data processing keyboard layout and the keymap ending in _TW for the typewriter layout.

For information about changing the default keyboard layout, see Section 3.3.5.

Table B–1 DECwindows Motif Keymap

Language	Model	DECwindows Keymap Name
Austrian/German	LK201–(AG,LG,BG,MG)	AUSTRIAN_GERMAN_LK201LG_DP
		AUSTRIAN_GERMAN_LK201LG_TW
	LK201–(NG,PG)	AUSTRIAN_GERMAN_LK201NG_DP
		AUSTRIAN_GERMAN_LK201NG_TW
	LK401–(AG)	AUSTRIAN_GERMAN_LK401AG_TW
Belgian/French	LK201–(AP,LP,BP,MP)	BELGIAN_FRENCH_LK201LP_DP
		BELGIAN_FRENCH_LK201LP_TW
Belgian/French	LK401–(AP)	BELGIAN_FRENCH_LK401AP_DP
		BELGIAN_FRENCH_LK401AP_TW

(continued on next page)

DECwindows Motif Keymap Names

Table B–1 (Cont.) DECwindows Motif Keymap

Language	Model	DECwindows Keymap Name
British	LK201–(AE,LE,BE,ME)	BRITISH_LK201LE_DP
		BRITISH_LK201LE_TW
British	LK401–(AA,PA)	BRITISH_LK401AA_DP
		BRITISH_LK401AA_TW
Canadian/French	LK201–(AC,LC,BC,MC)	CANADIAN_FRENCH_LK201LC_DP
		CANADIAN_FRENCH_LK201LC_TW
Canadian/French	LK401–(AC,LC,BC,MC)	CANADIAN_FRENCH_LK401AC_DP
		CANADIAN_FRENCH_LK401AC_TW
Danish	LK201–(AD,LD,BD,MD)	DANISH_LK201LD_DP
		DANISH_LK201LD_TW
	LK201–(ED,RD,FD)	DANISH_LK201RD_DP
		DANISH_LK201RD_TW
Danish	LK401–(AD,LD,BD,MD)	DANISH_LK401AD_DP
		DANISH_LK401AD_TW
Dutch	LK201–(AH,LH,BH,MH)	DUTCH_LK201LH_DP
		DUTCH_LK201LH_TW
	LK201–(NH,PH)	DUTCH_LK201NH
	LK401–(NH,PH)	DUTCH_LK401AH
Finnish	LK201–(AF,LF,BF,MF)	FINNISH_LK201LF_DP
		FINNISH_LK201LF_TW
	LK201–(NX,PX)	FINNISH_LK201NX_DP
		FINNISH_LK201NX_TW
Finnish	LK401–(AF,LF,BF,MF)	FINNISH_LK401AF_DP
		FINNISH_LK401AF_TW
Flemish	LK201–(AB,LB,BB,MB)	FLEMISH_LK201LB_DP
		FLEMISH_LK201LB_TW

(continued on next page)

Table B–1 (Cont.) DECwindows Motif Keymap

Language	Model	DECwindows Keymap Name
Flemish	LK401–(AB,LB,BB,MB)	FLEMISH_LK401AB_DP
		FLEMISH_LK401AB_TW
Icelandic	LK201–(AU,LU,BU,MU)	ICELANDIC_LK201LU_DP
		ICELANDIC_LK201LU_TW
Italian	LK201–(AI,LI,BI,MI)	ITALIAN_LK201LI_DP
		ITALIAN_LK201LI_TW
Italian	LK401–(AI,LI,BI,MI)	ITALIAN_LK401AI_DP
		ITALIAN_LK401AI_TW
North American	LK201–(AA,LA,BA,MA)	NORTH_AMERICAN_LK201LA
North American	LK401–(AA,LA,BA,MA)	NORTH_AMERICAN_LK401AA
Norwegian	LK201–(AN,LN,BN,MN)	NORWEGIAN_LK201LN_DP
		NORWEGIAN_LK201LN_TW
	LK201–(EN,RN,FN)	NORWEGIAN_LK201RN_DP
		NORWEGIAN_LK201RN_TW
Norwegian	LK401–(AN,LN,BN,MN)	NORWEGIAN_LK401AN_DP
		NORWEGIAN_LK401AN_TW
Portuguese	LK201–(AV,LV,BV,MV)	PORTUGUESE_LK201LV
Portuguese	LK401–(AV,LV,BV,MV)	PORTUGUESE_LK401AV
Spanish	LK201–(AS,LS,BS,MS)	SPANISH_LK201LS_DP
		SPANISH_LK201LS_TW
Spanish	LK401–(AS,LS,BS,MS)	SPANISH_LK401AS_DP
		SPANISH_LK401AS_TW
Swedish	LK201–(AM,LM,BM,MM)	SWEDISH_LK201LM_DP
		SWEDISH_LK201LM_TW
	LK201–(NM,PM)	SWEDISH_LK201NM_DP
		SWEDISH_LK201NM_TW
Swedish	LK401–(AM,LM,BM,MM)	SWEDISH_LK401AM_DP
		SWEDISH_LK401AM_TW
Swiss/French	LK201–(AK,LK,BK,MK)	SWISS_FRENCH_LK201LK_DP

(continued on next page)

DECwindows Motif Keymap Names

Table B–1 (Cont.) DECwindows Motif Keymap

Language	Model	DECwindows Keymap Name
		SWISS_FRENCH_LK201LK_TW
Swiss/French	LK401–(AK,LK,BK,MK)	SWISS_FRENCH_LK401AK_DP
		SWISS_FRENCH_LK401AK_TW
Swiss/German	LK201–(AL,LL,BL,ML)	SWISS_GERMAN_LK201LL_DP
		SWISS_GERMAN_LK201LL_TW
Swiss/German	LK401–(AL,LL,BL,ML)	SWISS_GERMAN_LK401AL_DP
		SWISS_GERMAN_LK401AL_TW
Combined US/UK	LK201–(EE,RE,PE)	UK_LK201RE
		US_LK201RE
Combined US/UK	LK401–(EE,RE,PE)	UK_LK401AA
		US_LK401AA

C
Troubleshooting Guidelines

This appendix contains information that can help you determine the cause of some common problems that you might encounter while using DECwindows. Section C.1 contains general troubleshooting techniques. Section C.2 lists problems by category.

C.1 General Troubleshooting Techniques

This section discusses some actions that you can take to diagnose and to help prevent the problems discussed in the previous sections. The items listed here are intended as general techniques for troubleshooting.

- After the system reboots, check the server error log file SYS$MANAGER:DECW$SERVER_0_ERROR.LOG for possible problems. The server process runs with the name DECW$SERVER_0. It creates the SYS$MANAGER:DECW$SERVER_0_ERROR.LOG log file every time DECwindows is activated. Always check this log file for error messages when you are experiencing problems. Always save the complete contents of the log file in case you must contact your Customer Support Center.
- To find the exit status of the DECW$LOGINOUT process, follow these steps:

  1. Use the DCL command ACCOUNTING/SINCE=TODAY and look for the most recent detached process termination running under the SYSTEM account.
  2. Issue the command WRITE SYS$OUTPUT F$MESSAGE(%x*nnnnnnn*), where *nnnnnnn* is the hexadecimal exit status.

3. If the error message is "Message number *nnnnnnn*", look at the facility code in the error message and issue a SET MESSAGE command for that facility's message file.

   For example, for a -DWT- message, issue the following command:

   ```
   $ SET MESSAGE SYS$MESSAGE:DECW$DWTMSG
   ```

4. Enter WRITE SYS$OUTPUT F$MESSAGE(%x*nnnnnnn*)

The message file for XLIB is SYS$MESSAGE:DECW$XLIBMSG. The message file for DECW is one of the following files:

SYS$MESSAGE:DECW$LOGINMSG
SYS$MESSAGE:DECW$TRANSPORTMSG
SYS$MESSAGE:DECW$TERMINALMSG

Note that DECW$LOGINOUT does not generate a DECW$TERMINALMSG exit status.

To create a DECwindows login log file, follow these steps:

1. If SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.COM doesn't exist, create it from SYS$MANAGER: DECW$PRIVATE_APPS_SETUP.TEMPLATE.
2. Edit SYS$MANAGER:DECW$PRIVATE_APPS_ SETUP.COM and remove the comment character (!) from the line:

   ```
   $! DECW$LOGINLOG == "SYS$MANAGER:DECW$LOGIN.LOG"
   ```

3. Restart DECwindows with the following command:

   ```
   $ @SYS$MANGER:DECW$STARTUP RESTART
   ```

After the DECW$LOGINOUT process exits, it will create the log file SYS$MANAGER:DECW$LOGIN.LOG.

- If the Start Session dialog box does not start, you can define a logical name to create a log file for the Start Session process. Edit the file SYS$MANAGER:DECW$PRIVATE_APPS_ SETUP.COM and define a log file location. If that file does not exist, copy the SYS$MANAGER:DECW$PRIVATE_APPS_ SETUP.TEMPLATE file to SYS$MANAGER:DECW$PRIVATE_ APPS_SETUP.COM and edit the .COM file. Use the symbol DECW$LOGINLOG to define a log file location, as in the following example:

  ```
  $ DECW$LOGINLOG == "SYS$MANAGER:DECW$LOGIN.LOG"
  ```

Restart DECwindows by using the following DCL command:

```
$ @SYS$MANGER:DECW$STARTUP RESTART
```

Check the log file for error messages.

- Use the Accounting utility to get information about the process. See the *HP OpenVMS System Management Utilities Reference Manual* for more information about using the Accounting utility. Enter the following command from a privileged account to get the accounting log for the DECterm process:

  ```
  $ ACCOUNTING/SINCE
  ```

  Enter the following command to find out the text for the completion status of the process that failed:

  ```
  $ EXIT %x000187DC
  ```

  An exit status of 1 indicates normal completion. Any other completion status might indicate a problem. Substitute the completion status for the number 000187DC.

- Session Manager creates the SYS$LOGIN:DECW$SM.LOG log file. If your session is terminated abnormally or if you notice strange behavior during your session, check the contents of this file. Always save the contents of this file in case you must contact your Customer Support Center.

- You can run many of the Session Manager files from the DCL command level. If you are experiencing problems during the session, try running the following executable files and see whether you get different results:

  - SYS$SYSTEM:DECW$PAUSESESSION.EXE (pausing a session)
  - SYS$SYSTEM:DECW$ENDSESSION.EXE (ending a session)
  - SYS$SYSTEM:DECW$SESSION.EXE (Session Manager menu bar)
  - SYS$SYSTEM:DECW$WSINIT.EXE (reading resource files and setting workstation customization options)
  - SYS$SYSTEM:DECW$WSCUST.EXE (viewing workstation customization dialog boxes)

- If you are part of a cluster environment and one workstation cannot run DECwindows but another can, be sure to check the SYS$SPECIFIC directories [SYSLIB], [SYSMGR], and [SYSEXE] for DECW$*.* files and delete any that you find. There should be no DECW$*.* files in any of those directories unless a user placed them there. The installation procedure does not place any DECW$*.* files in the SYS$SPECIFIC directories. Also, compare that system's SYSGEN parameters with those of other working systems.
- If you are having problems with one account but others on the same machine work fine, check the following:
  - Rename the resource files, as follows:

    ```
    $ RENAME DECW$USER_DEFAULTS:DECW*.DAT -
    _$ SYS$LOGIN:DECW*.OLD_DAT
    ```

    End the current session, start a new session, and try to use the account again.
  - Profile files are causing problems.

    Rename DECW$USER_DEFAULTS:*.VUE$DAT files and try again, as follows:

    ```
    $ RENAME SYS$LOGIN:*.VUE$DAT SYS$LOGIN:*.OLD_VUE$DAT
    ```
  - Session Manager cannot execute a command in your DECW$LOGIN.COM file or your DECW$SYLOGIN.COM file.

    Rename the SYS$LOGIN:DECW$LOGIN.COM file or the system file and try to run the application again, as follows:

    ```
    $ RENAME SYS$LOGIN:DECW$LOGIN.COM -
    _$ SYS$LOGIN:DECW$LOGIN.OLD
    ```
  - Session Manager cannot execute a command in your LOGIN.COM file or your SYLOGIN.COM file.

    Rename the SYS$LOGIN:LOGIN.COM file or the system file and try to run the application again, as follows:

    ```
    $ RENAME SYS$LOGIN:LOGIN.COM SYS$LOGIN:LOGIN.OLD
    ```

- The user authorization file (UAF) values for this user differ from those of other users.

  Compare the values with those of working accounts and make adjustments to find the problem.

- The user's device and directory are incorrect in the UAF record. Check whether any of the following conditions exist:

  1. The account has recently been moved.
  2. The disk is on line and working.
  3. The user directory has the correct protections for the user to create files in the directory.
  4. The user is the owner of their SYS$LOGIN directory.

- If you see a "Can't Open Display" error message when you run an application, you might be able to get a more specific error message by running the following program:

  ```
  $ RUN DECW$EXAMPLES:ICO.EXE
  ```

  This program is on your system if your system manager installed the programming support during the DECwindows installation. If not, you can restore it to your system from the DECwindows installation distribution media. See the *HP OpenVMS System Management Utilities Reference Manual* for more information on restoring a file from a save set using the Backup utility.

- When DECterm windows disappear with no trace and you want to find out why, place the following command in your LOGIN.COM file.

  ```
  $ DEFINE DECW$DECTERM_OUTPUT DECTERM.LOG
  ```

  End the current session and start a new session. Extra output is written to the DECTERM.LOG file when the DECterm windows run and when they stop working.

- If you find a hexadecimal error status in one of the log files (for example, 02DBA002), you can translate it to a text message by creating the following DECWMSG.COM file:

```
$ SET MESSAGE SYS$MESSAGE:DECW$TRANSPORTMSG
$ WRITE SYS$OUTPUT F$MESSAGE(P1)
$ SET MESSAGE SYS$MESSAGE:DECW$XLIBMSG
$ WRITE SYS$OUTPUT F$MESSAGE(P1)
```

For example:

```
$ @DECWMSG %X02DBA002
%DECW-E-CNXABORT, connection aborted
%NONAME-E-NOMSG, Message number 02DBA002
```

In this example, the %NONAME-E-NOMSG message means that status 02DBA002 was not found in the DECW$XLIBMSG.EXE file. Therefore, the status 02DBA002 translates to "DECW-E-CNXABORT, connection aborted."

- Periodically use AUTOGEN with feedback to increase efficiency and prevent quota-related problems. AUTOGEN makes adjustments to your system after evaluating your hardware configuration and estimating typical work loads. See the *HP OpenVMS System Management Utilities Reference Manual* for more information about AUTOGEN. Section A.1 and Section A.2 in this manual include tables of suggested settings for UAF limits and system parameters.

C.2 Common Problems

This section lists common problems in the following categories:

- Start Session dialog box
- Licensing
- FileView
- Window Manager
- Font access
- Network
- Using Applications
- Session Manager
- DECterm
- X Terminals and third-party workstations
- X server

- Keyboard, monitor, and pointer

After each topic heading, one or more problem statements appear in ***boldface italic type.*** Each problem statement is followed by a brief list of possible causes and solutions.

C.2.1 Start Session Dialog Box

The Start Session dialog box does not appear when the system reboots after an installation.

- Check the SYS$SPECIFIC directories [SYSLIB], [SYSMGR], and [SYSEXE] for DECW$*.* files and delete any that you find. There should be no DECW$*.* files in any of those directories unless a user placed them there. The installation procedure does not place any DECW$*.* files in the SYS$SPECIFIC directories.
- Make sure that the protection code on all DECwindows shareable images is set to world read and world execute (W:RE). This is the default protection used during the installation of DECwindows files. See the *DECwindows Motif Version 1.2 for OpenVMS Installation Guide* for a list of files provided by DECwindows. See the *HP OpenVMS DCL Dictionary* for information about setting protection with the DCL command SET FILE/PROTECTION or SET PROTECTION.

  The following commands might help you identify a file protection problem. Execute the commands from a privileged account.

  ```
  $ SET AUDIT/ALARM/ENABLE=FILE=FAILURE
  $ REPLY/ENABLE=SECURITY
  $ @SYS$MANAGER:DECW$STARTUP RESTART
  ```

  The first command causes file access failures to be logged in the operator log file. The second command requests OPCOM to display security violation messages on your screen. The last command restarts DECwindows Motif.

  See the *HP OpenVMS DCL Dictionary*, the *HP OpenVMS System Management Utilities Reference Manual*, and the security guide for more information about auditing.
- Check the server error log file SYS$MANAGER:DECW$SERVER_0_ERROR.LOG for error messages that might give more details about the problem.

- Check the SYS$MANAGER:NETSERVER.LOG file for error messages that relate to a DECnet transport problem. See the *DECnet for OpenVMS Networking Manual* for information about error messages contained in the NETSERVER.LOG file.
- See Section C.1 for more help in diagnosing the problem.

C.2.2 Licensing

When you try to start a new session, the dialog box indicates that "No license is active for this software product."

- A license on your workstation is invalid. If a new license is available, reboot the operating system by performing a conversational boot using minimum startup. (See the installation and operations guide for your VAX computer for information about performing a conversational boot using minimum startup.) Log in at the console and issue the LICENSE DISABLE command to disable the expired license. Register a new license using the License Management Facility (LMF). See the *OpenVMS License Management Utility Manual* for information about using LMF.

C.2.3 FileView

When you start FileView, you receive a message indicating that your directory does not exist.

- If your system manager has recently moved your directory, follow these steps to instruct FileView to use the new directory specification:
  1. Start the FileView application.
  2. Type in your new directory specification.
  3. Choose Save Startup (FileView) from FileView's Options menu.

C.2.4 Window Manager

You cannot iconify any windows. The command buttons are missing from the top of the windows.

- The Motif Window Manager has stopped working. Choose either of the following options to restart the Motif Window Manager program:
  - Use the following commands to restart the Motif Window Manager program:

    ```
    $ SET DISPLAY/CREATE/TRANSPORT=LOCAL
    $ RUN SYS$SYSTEM:DECW$MWM
    ```

    Enter these commands from a DECterm window, from a FileView DCL command window, or from another node by setting host to that workstation. You will be unable to use the window for any other operation.
  - Add the Window Manager menu item to Session Manager's Applications menu using the Menu dialog box from Session Manager's Options menu. Then choose Window Manager from Session Manager's Applications menu. See *Using DECwindows Motif for OpenVMS* for more information about adding a menu item to a menu.

C.2.5 Font Access

Your program can neither access nor find a font.

- The fonts are located in the wrong directory. Make sure that layered-product fonts are in one of the following disk directories:

  ```
  SYS$COMMON:[SYSFONT.DECW.USER_75DPI]
  SYS$COMMON:[SYSFONT.DECW.USER_100DPI]
  SYS$COMMON:[SYSFONT.DECW.USER_COMMON]
  SYS$COMMON:[SYSFONT.DECW.USER_CURSOR16]
  SYS$COMMON:[SYSFONT.DECW.USER_CURSOR32]
  ```

  The USER_75DPI directory is for fonts displayed on 75-dpi monitors, the USER_100DPI directory is for fonts displayed on 100-dpi monitors, and the USER_COMMON directory is for fonts that can be displayed on either 75-dpi or 100-dpi monitors. The USER_CURSOR16 directory is for 16x16 pixel cursors, and the USER_CURSOR32 directory is for 32x32 pixel cursors.

- The font directory files are not updated. To update the font directories, enter the following command from a privileged account:

  ```
  $ @SYS$UPDATE:DECW$MKFONTDIR
  ```

  This command creates new font directories for all layered-product fonts and runs automatically when DECwindows is installed. The command might not have been run by a layered product that supplies its own fonts.

- A third-party font is being used. Choose any of the following options:

  - Obtain the font source in bitmap distribution format (BDF) format. Compile the font with the FONT command. The *VMS DECwindows Guide to Xlib (Release 4) Programming: MIT C Binding* provides a full description of the FONT command.

    On VAX systems, copy the resulting *.DECW$FONT file to either the SYS$COMMON:[SYSFONT.DECW.USER_75DPI] directory or the SYS$COMMON:[SYSFONT.DECW.USER_100DPI] directory, as appropriate.

    On AXP systems, copy the resulting *.PCF file to either the SYS$COMMON:[SYSFONT.DECW.USER_75DPI] directory or the SYS$COMMON:[SYSFONT.DECW.USER_100DPI] directory, as appropriate.

    Then create new font directories using the following command:

    ```
    $ @SYS$UPDATE:DECW$MKFONTDIR
    ```

    You must end the current session and start a new session in order for the new font to become available.

  - A file name is used instead of a standard font name.

    - For fixed-width fonts, such as 6x10, copy the file DECW$EXAMPLES:DECW$FONT_ALIAS_FILENAMES.DAT to the SYS$COMMON:[SYSFONT.DECW.USER_COMMON]

directory. You must end the current session and start a new session for the change to take effect.

- Modify your program to use standard font names. See the *VMS DECwindows Guide to Xlib (Release 4) Programming: MIT C Binding* for standard DECwindows font names.

- If your program requires either the Charter or Lucida family of fonts and they do not exist on your system, use the example font alias files provided by DECwindows to give the font name aliases to existing DECwindows font names.

  For 75-dpi monitors, copy the following files to the SYS$COMMON:[SYSFONT.DECW.USER_75DPI] directory:

  ```
  DECW$EXAMPLES:DECW$FONT_ALIAS_CHARTER.DAT
  DECW$EXAMPLES:DECW$FONT_ALIAS_LUCIDA.DAT
  ```

  For 100-dpi monitors, copy the following files to the SYS$COMMON:[SYSFONT.DECW.USER_100DPI] directory:

  ```
  DECW$EXAMPLES:DECW$FONT_ALIAS_CHARTER_100DPI.DAT
  DECW$EXAMPLES:DECW$FONT_ALIAS_LUCIDA_100DPI.DAT
  ```

- Give the third-party font name an alias to an existing DECwindows font name. The third party font must be an ISO Latin1 font. Create the file DECW$FONT_ALIAS.DAT in either the SYS$COMMON:[SYSFONT.DECW.USER_75DPI] directory or the SYS$COMMON:[SYSFONT.DECW.USER_100DPI] directory, depending on whether the third party font is 75 dpi or 100 dpi. The format of the font alias file is:

  alias-font-name actual-font-name

  If the font names contain embedded spaces, enclose the names in double quotes (" "). The *alias-font-name* value is the third-party font name, and the *actual-font-name* value is a DECwindows font name. See the *VMS DECwindows Guide to Xlib (Release 4) Programming: MIT C Binding* for DECwindows font names.

C.2.6 Keyboard, Monitor, and Pointer

The spacing and fonts look fine on screen 0 but not on screen 1.

- You might be using monitors with different densities. Edit the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM file and define DECW$MONITOR_DENSITY so that the dpi values are the same for both screens, as in the following example:

  ```
  $ DECW$MONITOR_DENSITY == "100,100"
  ```

  If you do not have that file, copy the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.TEMPLATE file to SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM, and edit the .COM file.

  Enter the following command to restart the display server:

  ```
  $ @SYS$MANAGER:DECW$STARTUP RESTART
  ```

  This command stops all applications and restarts DECwindows. Both screens should now use the same dpi values. All clustered workstations also do so when they are restarted. See Chapter 3 for more information about the private server setup file and for information about how to change individual workstations in a cluster.

You cannot compose characters on OpenVMS workstations.

- If you are using an LK201 keyboard, the Compose Character key functions as the Alt function key. Hold down the Compose Character key and press the space bar. The Compose indicator lights up. Then enter the compose character sequence (for example, a ' to produce á).
- If you are using an LK401 keyboard, choose one of these options:
  - Pull down Session Manager's Options menu and choose the Keyboard menu item. A dialog box is displayed. From the Keyboard Type list, choose LK401, then click on OK.
  - Use the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM file to tell the server to default to the LK401 keyboard, as in the following example:

```
$ DECW$DEFAULT_KEYBOARD_MAP == "US_LK401AA"
```

If you do not have that file, copy the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.TEMPLATE file to SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM. Add the following line to the .COM file:

```
$ DECW$DEFAULT_KEYBOARD_MAP == "US_LK401AA"
```

See Section 3.1 for an example.

Enter the following command to restart the DECwindows server:

```
$ @SYS$MANAGER:DECW$STARTUP RESTART
```

This command stops all applications and restarts DECwindows. The server now treats the keyboard as an LK401 keyboard. All clustered workstations also do so when they are restarted. See the Chapter 3 for more information on the private server setup file and for information about how to change individual workstations in a cluster.

The arrow keys move the mouse pointer instead of working with the application.

- If you pressed Shift+Ctrl+F3 to enter pseudomouse mode, you must press Shift+Ctrl+F3 to exit pseudomouse mode.
- Click with any mouse button to exit pseudomouse mode.

You cannot dismiss the Pseudo Mouse Mode dialog box without exiting pseudomouse mode.

- To dismiss the dialog box without exiting pseudomouse mode, position the cursor over the OK button in the Pseudo Mouse Mode dialog box and then press the Return key.

The top part of the screen goes black and some text messages are visible.

- If you pressed Ctrl+F2 to enter console display mode, you must press Ctrl+F2 to exit console display mode.
- If OPCOM wrote a message to the console, either press Ctrl+F2 to exit console display mode, or disable OPCOM messages to the console.

To disable OPCOM messages, log in to a privileged account and enter the following commands:

```
$ SET TERMINAL/PERMANENT/NOBROADCAST OPA0:
$ DEFINE/USER SYS$COMMAND OPA0:
$ REPLY/DISABLE
```

You might want to add those commands to the SYS$MANAGER:SYSTARTUP_V5.COM file so that OPCOM messages are disabled after a system reboot.

C.2.7 Network

A program that is displayed on another node exits with the following error message:

```
XIO:  fatal IO error 65535  on X server "_WSA3:"
      after 600 requests (597 known processed) with 0 events remaining.
%XLIB-F-IOERROR, xlib io error
-SYSTEM-F-LINKABORT, network partner aborted logical link
```

- The user on the other workstation logged out. That user must log back in.
- The display server stopped on the other workstation. Restart DECwindows by setting host to the other workstation, logging in to a privileged account, and entering the following command:

  ```
  $ @SYS$MANAGER:DECW$STARTUP RESTART
  ```

- Network congestion caused the program to exit. If possible, use the fastest available CPU for DECnet routing. See the *DECnet for OpenVMS Networking Manual* for information about network configuration and routing node requirements.

A program that is displayed on another node exits with the following error message:

```
XIO:  fatal IO error 65535  on X server "_WSA3:"
      after 267 requests (229 known processed) with 0 events remaining.
%XLIB-F-IOERROR, xlib io error
-SYSTEM-F-PATHLOST, path to network partner node lost
```

- The other workstation was powered off or halted, or the operating system crashed. You must power on, restart, or reboot the other workstation.

- Network congestion caused the program to exit. If possible, use the fastest available CPU for DECnet routing. See the *DECnet for OpenVMS Networking Manual* for information about network configuration and routing node requirements.

A program that is displayed on another node exits with the following message:

```
XIO:  fatal IO error 65535  on X server "_WSA3:"
      after 259 requests (259 known processed) with 0 events remaining.
%XLIB-F-IOERROR, xlib io error
-SYSTEM-F-THIRDPARTY, network logical link disconnected by a third party
```

- The user on the other workstation might have shut down DECnet. Restart the DECnet software on the other workstation.

C.2.8 Applications

The application window comes up but the window is blank.

- The application incorrectly asks for backing store but also waits for an expose event to begin processing. Choose one of the following options to disable backing store:
  - If possible, modify the program so that it does not request backing store.
  - Disable backing store by editing the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM file. If you do not have that file, copy the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.TEMPLATE file to SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM, and edit the .COM file. Add the following line to the cluster common or standalone workstation section:

    ```
    $ DECW$SERVER_DISABLE_BACKING_STORE == "TRUE"
    ```

    Exit the file and enter the following command to restart the server:

    ```
    $ @SYS$MANAGER:DECW$STARTUP RESTART
    ```

    This command stops all applications and restarts DECwindows. Applications, including those running on other clustered workstations, will not use backing store when they are restarted.

See Chapter 3 for more information about the private server setup file and for information about how to change individual workstations in a cluster.

When you run an application, you see a "Can't open display" message, or the application can run with DECnet transport but cannot run with local transport.

- The number of global sections or global pages might be insufficient.

  Enter the DCL command SET DISPLAY/CREATE/TRANSPORT=DECNET and run the application. If you can run the application with DECnet transport but not with local transport, you might need to increase the global sections or global pages or both. Local transport (the default) uses global sections and global pages. For each DECwindows application, you need at least one global section and about 300 global pages.

  Use the F$GETSYI lexical function to determine the number of free global pages and global sections, as follows:

```
$ GBLPAGES = F$GETSYI("FREE_GBLPAGES")
$ SHOW SYMBOL GBLPAGES
$ GBLSECT = F$GETSYI("FREE_GBLSECTS")
$ SHOW SYMBOL GBLSECT
```

  See the *HP OpenVMS DCL Dictionary* for more information on using the F$GETSYI lexical function.

  Increase the SYSGEN parameter GBLSECTION or GBLPAGES. See the *HP OpenVMS System Management Utilities Reference Manual* for information about modifying system parameters.

  Use the Digital-supplied command procedure SYS$UPDATE:AUTOGEN.COM to help determine the correct values and parameters for your system. AUTOGEN makes adjustments to your system after evaluating your hardware configuration and estimating typical work loads. See the *HP OpenVMS System Management Utilities Reference Manual* for more information about AUTOGEN.

- If you see a "Can't Open Display" error message when you run an application, you might be able to get a more specific error message by running the following program:

  ```
  $ RUN DECW$EXAMPLES:ICO.EXE
  ```

  This program is on your system if you installed the programming support during the DECwindows installation. If the program is not on your system, you can restore it from the DECwindows installation distribution media. See the *HP OpenVMS System Management Utilities Reference Manual* for more information about restoring a file from a save set using the Backup utility.

- See Section C.1 for more help in diagnosing the problem.

You cannot display more than 16 applications on a workstation.

- Determine the transport type by entering the following command:

  ```
  $ SHOW DISPLAY
  ```

  – If you are using DECnet transport, you might have exceeded the maximum number of DECnet links.

    Use the following commands to increase the maximum number of DECnet links:

    ```
    $ RUN SYS$SYSTEM:NCP
    NCP> SET EXECUTOR MAXIMUM LINKS 32
    NCP> EXIT
    ```

  – If you are using the OpenVMS/ULTRIX Connection product (UCX) transport, you might have exceeded the maximum number of TCP/IP connections.

    See the appropriate UCX documentation.

  – If you are using local transport, the number of global pages or global sections might be insufficient.

    Increase the SYSGEN parameter GBLPAGES or GBLSECTIONS. See the *HP OpenVMS System Management Utilities Reference Manual* for information about modifying system parameters.

See the previous problem statement titled "When you run an application, you see..." for information about global pages and global sections.

Your DECwindows application does not run.

- Check the SYS$SPECIFIC directories [SYSLIB], [SYSMGR], and [SYSEXE] for DECW$*.* files and delete any that you find. There should be no DECW$*.* files in any of those directories unless a user placed them there. The installation procedure does not place any DECW$*.* files in the SYS$SPECIFIC directories.
- Resource files that you have edited might be causing problems. Editing resource files is not encouraged because any syntax errors or typographical errors can cause applications to stop functioning.

  Rename the resource files as follows:

  ```
  $ RENAME SYS$LOGIN:DECW*.DAT SYS$LOGIN:DECW*.OLD_DAT
  ```

  End the current session, and start a new session. Try to run the application again.
- If you see a "Can't Open Display" error message when you run an application, you might be able to get a more specific error message by running the following program:

  ```
  $ RUN DECW$EXAMPLES:ICO.EXE
  ```

  This program is on your system if you installed the programming support during the DECwindows installation. If the program is not on your system, you can restore it from the DECwindows installation distribution media. See the *HP OpenVMS System Management Utilities Reference Manual* for more information on restoring a file from a save set using the Backup utility.
- Use the Digital-supplied command procedure SYS$UPDATE:AUTOGEN.COM to help determine the correct values and parameters for your system. AUTOGEN makes adjustments to your system after evaluating your hardware configuration and estimating typical work loads. See the *HP OpenVMS System Management Utilities Reference Manual* for more information about AUTOGEN.
- See Section C.1 for more help in diagnosing the problem.

When you run an application, you see error messages that say, "Client is not authorized to access server."

- Check Session Manager's Security Options box to ensure that the user is authorized with the correct node name, user name, and transport.
- For DECnet transport, the node names need to be defined in the network node database of both the system running the server and the system running the application. Use the Network Control Program utility (NCP) to define the node names. See the *DECnet for OpenVMS Network Management Utilities* for more information about using NCP.
- TCP/IP transport does not pass user names for authorization to the server.

  Be sure to use an asterisk (*) in the Username entry box of the Security Options box if you are using TCP/IP transport for a given node.
- If you are using TCP/IP to display an application on your own system, you must authorize node zero (0), which is the local node.
- Check the server error log file SYS$MANAGER:DECW$SERVER_0_ERROR.LOG for error messages that might provide more details about the problem.
- Check the SYS$MANAGER:NETSERVER.LOG file for error messages that relate to a DECnet transport problem. See the *DECnet for OpenVMS Networking Manual* for information about error messages contained in the NETSERVER.LOG file.

C.2.9 Session Manager

When you start an application from Session Manager, the whole session is terminated. The session is logged out and a new Start Session dialog box appears.

- Search the SYS$LOGIN:DECW$SM.LOG file for either of the following error messages:

```
X Toolkit Error: Cannot perform calloc

Session Error: %XLIB-E-INSFMEM,
       insufficient dynamic memory
```

If either error message is present, you might need to increase the PGFLQUOTA value for the accounts that are experiencing the problem. See the *HP OpenVMS System Management Utilities Reference Manual* for information about modifying user authorization file (UAF) values using the Authorize utility.

You might also need to increase the SYSGEN parameter PQL_DPGFLQUOTA. See the *HP OpenVMS System Management Utilities Reference Manual* for information about modifying system parameters.

Use the Digital-supplied command procedure SYS$UPDATE:AUTOGEN.COM to help determine the correct values and parameters for your system. AUTOGEN makes adjustments to your system after evaluating your hardware configuration and estimating typical workloads. See the *HP OpenVMS System Management Utilities Reference Manual* for more information on AUTOGEN.

- See Section C.1 for more help in diagnosing the problem.

The Start Session dialog box disappears for no apparent reason.

- An "idle process killer" has assumed that the process _WSA*n*: is idle and has terminated it.

  Do not run an "idle process killer" on your workstation. This could inadvertently terminate Session Manager or Window Manager even though other applications are running on the system.

  To restart the Start Session dialog box, follow the directions given in the X server section under the problem statement titled "You cannot start a session on the workstation."

- Your system disk has reached a threshold that caused the AUDIT_SERVER process either to suspend or to delete processes.

  Take measures to recover the system by determining why the AUDIT_SERVER exceeded the threshold. This might require rebooting the system with a conversational boot and specifying a minimum startup.

After login, the Session Manager menu bar does not start.

- Check the SYS$SPECIFIC directories [SYSLIB], [SYSMGR], and [SYSEXE] for DECW$*.* files and delete any that you find. There should be no DECW$*.* files in any of those directories unless a user placed them there. The installation procedure does not place any DECW$*.* files in the SYS$SPECIFIC directories.
- If you have redefined your logical name table list, make sure that you have included DECW$LOGICAL_NAMES in the logical name table search list. Verify that LNM$FILE_DEV includes the table. Also verify that the protection code of any added table allows the user read access or that the table is listed after DECW$LOGICAL_NAMES.
- Make sure that the protection code on all DECwindows shareable images is set to world read and world execute (W:RE). This is the default protection used during the installation of DECwindows files. See the *DECwindows Motif Version 1.2 for OpenVMS Installation Guide* for a list of files provided by DECwindows. See the *HP OpenVMS DCL Dictionary* for information about setting protection with the DCL command SET FILE/PROTECTION or SET PROTECTION.
- Resource files that you have edited might be causing problems. Editing resource files is not encouraged because any syntax or typographical errors can cause applications to stop functioning.

  Rename the resource files, as follows:

  ```
  $ RENAME SYS$LOGIN:DECW*.DAT SYS$LOGIN:DECW*.OLD_DAT
  ```

  End the current session, start a new session, and try to run the application again.
- Profile files are causing problems.

  Rename the SYS$LOGIN:*.VUE$DAT files, as follows:

  ```
  $ RENAME SYS$LOGIN:*.VUE$DAT SYS$LOGIN:*.OLD_VUE$DAT
  ```

  Try to run the application again.

- Session Manager cannot execute a command in your DECW$LOGIN.COM file or your DECW$SYLOGIN.COM file.

  Rename the SYS$LOGIN:DECW$LOGIN.COM file or the system file, as follows:

  ```
  $ RENAME SYS$LOGIN:DECW$LOGIN.COM -
  _$ SYS$LOGIN:DECW$LOGIN.OLD
  ```

  Try to run the application again.

- Session Manager cannot execute a command in your LOGIN.COM file or your SYLOGIN.COM file.

  Rename the SYS$LOGIN:LOGIN.COM file or the system file, as follows:

  ```
  $ RENAME SYS$LOGIN:LOGIN.COM SYS$LOGIN:LOGIN.OLD
  ```

  Try to run the application again.

- See Section C.1 for more help in diagnosing the problem.

C.2.10 DECterm

Your DECterm window does not display characters.

- You pressed either Ctrl+S or F1.

  Check whether the Hold Screen indicator is lit. If it is, press either Ctrl+Q or F1.

- The DECwindows server is "frozen."

  Select another window and try to type characters to it. If characters are not displayed, refer to the problem statement titled "The screen seems to be frozen" in Section C.2.12.

- The DECterm controller process is stuck.

  Enter the DCL command SHOW PROCESS and look for the process named DECW$TE_*nnn*. If it is in an RWAST state, either wait for it to complete or create a new terminal controller process. To create a new controller process, execute the following commands from a privileged account:

  ```
  $ SET DISPLAY/CREATE/TRANSPORT=LOCAL
  $ RUN SYS$SYSTEM:DECW$TERMINAL
  ```

You might want to execute these commands in a batch job that has interactive priority, since the process running the terminal controller cannot be used for anything else.

The application looks different in a DECterm window than on a VT340 terminal.

- The DECterm emulates only VT52, VT100, VT200, and VT320 class terminals.

  If possible, modify the application.

The DECterm fonts are small and hard to read.

- Small fonts might be enabled in the DECterm Options menu.

  Pull down the DECterm Options menu and choose the Window... menu item. Select the Big Font option and click on OK.

- Fonts intended for 75-dpi monitors are displayed on a 100-dpi monitor.

  Use the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM file to tell the server to use 100-dpi fonts. If you do not have that file, copy the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.TEMPLATE file to SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM. Add the following line to the .COM file:

  ```
  $ DECW$SERVER_DENSITY == "100"
  ```

  See Section 3.1 for an example.

  Enter the following command to restart the DECwindows server:

  ```
  $ @SYS$MANAGER:DECW$STARTUP RESTART
  ```

  This command stops all applications and restarts DECwindows. All applications will now use the larger, 100-dpi fonts. All clustered workstations will use 100-dpi fonts when restarted.

  See Chapter 3 for more information about the private server setup file and for information about how to change individual workstations in a cluster.

You are not getting as many DECterm windows from automatic startup as you requested.

- Use the Accounting utility to get information about the process. See the *HP OpenVMS System Management Utilities Reference Manual* for information about using the Accounting utility. Enter the following command from a privileged account to get the accounting log for the application that is having problems:

  ```
  $ ACCOUNTING/SINCE
  ```

  Enter the following command to find out the text for the completion status of the process that failed:

  ```
  $ EXIT %x00000124
  ```

  An exit status of 1 indicates normal completion. Any other completion status might indicate a problem. Substitute the completion status for the number 00000124.

  If the accounting log for the process shows an exit status of 00000124 (SYSTEM-F-INSFMEM, insufficient dynamic memory), you need to increase either the PAGEDYN or the NPAGEDYN parameter. This error message usually means that you ran out of nonpaged dynamic memory and that the NPAGEDYN SYSGEN parameter needs to be increased. However, sometimes DECterm exits with this status even when paged dynamic memory is depleted.

  Enter the DCL command SHOW MEMORY/FULL to identify which type of memory is insufficient. The Free column should display a low number for either Nonpaged Dynamic Memory or Paged Dynamic Memory, as follows:

```
Dynamic Memory Usage (bytes):      Total        Free      In Use     Largest
  Nonpaged Dynamic Memory         931840       49040      882800       27936
  Paged Dynamic Memory            949760           0      552800      393792
```

  In this case, increase the PAGEDYN system parameter. See the *HP OpenVMS System Management Utilities Reference Manual* for information about modifying system parameters.

- See Section C.1 for more help in diagnosing the problem.

You cannot start DECterm application windows.

- Use the Accounting utility to get information on the process. See the *HP OpenVMS System Management Utilities Reference Manual* for more information about using the Accounting utility. Enter the following command from a privileged account to get the accounting log for the DECterm process:

  ```
  $ ACCOUNTING/SINCE
  ```

  Enter the following command to find out the text for the completion status of the process that failed:

  ```
  $ EXIT %x000187DC
  ```

  An exit status of 1 indicates normal completion. Any other completion status might indicate a problem. Substitute the completion status for the number 000187DC. If the accounting log shows a completion status of 000187DC (RMS-F-EXENQLM, exceeded enqueue quota), you might need to increase the ENQLM value of the account that is experiencing problems. See the *HP OpenVMS System Management Utilities Reference Manual* for information about modifying user authorization file (UAF) values using the Authorize utility.

- See Section C.1 for more help in diagnosing the problem.

DECterm windows disappear if the DECW$TE_nnn process paging file quota becomes less than 420 pages. No error message is displayed.

- Each default DECterm with 500 lines saved needs 420 pages of paging file quota.

  Increase the PGFLQUO value in the user's UAF record using the Authorize utility. Increase the PQL_DPGFLQUOTA system parameter (the default is 8192 pages) using the OpenVMS System Generation utility.

  See the *HP OpenVMS System Management Utilities Reference Manual* for information about modifying UAF values with the Authorize utility and for information about modifying system parameters.

  Use the Digital-supplied command procedure SYS$UPDATE:AUTOGEN.COM to help determine the correct values and parameters for your system. AUTOGEN makes adjustments to your system after evaluating your hardware configuration and estimating typical work loads. See the *HP OpenVMS System Management*

Utilities Reference Manual for more information about AUTOGEN.

- Reduce or eliminate lines saved off the top of the DECterm screen. Choose the Display item on the Options menu and decrease the value of the Record Lines Off Top option.

If the process BUFIOCNT becomes 0, DECW$TE_nnn enters the MWAIT state and generates the following error:

```
%X2DBC072
%DECW-E--TIMEOUT_CONTROL, Timeout waiting for DECterm controller to start
```

- Each DECterm creation requires 2 BUFIO.

  Increase the SYSGEN parameter PQL_MBIOLM (the default is 40). See the *HP OpenVMS System Management Utilities Reference Manual* for information about modifying system parameters.

  Use the Digital-supplied command procedure SYS$UPDATE:AUTOGEN.COM to help determine the correct values and parameters for your system. AUTOGEN makes adjustments to your system after evaluating your hardware configuration and estimating typical work loads. See the *HP OpenVMS System Management Utilities Reference Manual* for more information about AUTOGEN.

After creating some number of DECterm windows, you get either of the following errors while trying to create a new one:

```
%X1000039C

%SYSTEM-F-NOSLOT, no PCB available
```

- Increase the SYSGEN parameter MAXPROCESSCOUNT (the default is 32). See the *HP OpenVMS System Management Utilities Reference Manual* for information about modifying system parameters.

C.2.11 X Terminals and Third-Party Workstations

You cannot display from a DECwindows workstation to a third-party workstation.

- The third-party workstation is not running the X Window System.

  Make sure the third-party workstation is running the X Window System.

- The network software is incompatible.

  If you are running DECnet software, make sure that the third-party workstation is also running DECnet.

  If you are using TCP/IP, install the OpenVMS/ULTRIX Connection software (UCX) on the OpenVMS workstation, or install a third-party vendor's TCP/IP networking product. You do not need a UCX Product Authorization Key (PAK) to use UCX with DECwindows.

- The network database definitions are incorrect.

  Make sure the DECnet or TCP/IP node name and node number definitions for both workstations are correct on both workstations.

- The node name contains lowercase letters.

  Add an uppercase node name alias for the node name, or enclose the node name in double quotes, as in the following example:

  ```
  $ SET DISPLAY/CREATE/TRANSPORT=TCPIP/NODE="ultrixnode"
  ```

- Access control is not enabled on the third-party workstation.

  Consult the documentation for the third-party workstation for information about adding the OpenVMS node name to the server's access control list.

You cannot display from a third-party workstation to a DECwindows workstation.

- The third-party workstation is not running the X Window System.

  Make sure the third-party workstation is running the X Window System.

- The network software is incompatible.

  If you are running DECnet software, make sure the third-party workstation is also running DECnet.

  If you are using TCP/IP, install the OpenVMS/ULTRIX Connection software (UCX) on the OpenVMS workstation, or install a third-party vendor's TCP/IP networking product. You do not need a UCX Product Authorization Key (PAK) to use UCX with DECwindows.

- The network database definitions are incorrect.

  Make sure the DECnet or TCP/IP node name and node number definitions for both workstations are correct on both workstations.

- The network software is not running.

  If you are using UCX, use the UCX$UCP loop command to verify that the other workstation responds to notification from your workstation, as follows:

  ```
  $ RUN SYS$SYSTEM:UCX$UCP
  UCX> LOOP DECW2
  DECW2 is alive
  UCX> EXIT
  ```

  The message "DECW2 is alive" means that UCX was started properly. If UCX was not started properly, a message similar to the following is displayed:

  ```
  $ RUN SYS$SYSTEM:UCX$UCP
  UCX> LOOP DECW2
  DECW2 does not respond
  ```

UCX must be started with the command file UCX$STARTUP. Add the following line to your SYS$MANAGER:SYSTARTUP_V5.COM file:

```
@SYS$STARTUP:UCX$STARTUP
```

UCX should be started after DECnet has been started.

- A network-specific OpenVMS DECwindows transport interface is not supplied.

  If you are using a third-party TCP/IP networking product, it must supply the file SYS$SHARE:DECW_TRANSPORT_*name*, where *name* is the product name.

- The transport interface is not installed.

  If you are using a third-party TCP/IP networking product, the DECW_TRANSPORT_*name* file must be installed from the SYS$MANAGER:SYSTARTUP_V5.COM file with the following command:

  ```
  $ INSTALL CREATE SYS$SHARE:DECW_TRANSPORT_name -
  _$ /OPEN/SHARED/HEADER_RES/PROTECTED
  ```

  Use the following command from a privileged account to verify that the file is installed:

  ```
  $ INSTALL LIST SYS$SHARE:DECW_TRANSPORT_name
  ```

- The server is not listening for connections. Choose either of the following options:

  – If you are using TCP/IP, you must use the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM customization file. If you do not have that file, copy the SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.TEMPLATE file to SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM. Add the following line to the .COM file:

    ```
    DECW$SERVER_TRANSPORTS == "DECNET,LOCAL,TCPIP"
    ```

    Restart the server with the following command:

    ```
    $ @SYS$MANAGER:DECW$STARTUP RESTART
    ```

    This command stops all applications and restarts DECwindows. All workstations in the cluster will listen for TCP/IP connections.

See Chapter 3 for more information about the private server setup file and for information about how to change individual workstations in a cluster.

After restarting DECwindows, you should see messages similar to the following in SYS$MANAGER: DECW$SERVER_0_ERROR.LOG:

```
Now attach all known txport images
%DECW-I-ATTACHED, transport DECNET attached to its network
%DECW-I-ATTACHED, transport TCPIP attached to its network
```

There should be one message for each transport you listed in DECW$SERVER_TRANSPORTS, except for local.

- Access control is not enabled on the OpenVMS workstation.

  Pull down Session Manager's Options menu and choose the Security menu item. Type the third-party node name in the Node entry box. If you are using DECnet, type the user name in the Username entry box and add it. Then type the user ID (UID) in the Username entry box and add it. If you are using the OpenVMS/ULTRIX Connection software (UCX) TCP/IP, type an asterisk (*) in the Username entry box, and type TCPIP in the Transport entry box. If you are using a third-party TCP/IP, follow the directions supplied by the third-party vendor.

Note

All node names are shown in the security list in uppercase, but they will match both uppercase and lowercase node names.

You cannot display PostScript images on X terminals or on third-party workstations.

- The X terminal or third-party workstation does not support the Display PostScript extension.

  The only solution is to get an X terminal workstation that supports the Display PostScript extension. The PostScript commands are performed by an extension to the server, not by a library on the client (application) side.

C.2.12 X Server

You cannot start a session on the workstation. Your keystrokes are not displayed on the screen.

- The DECwindows X server has stopped working.

  Press Return. A user name prompt should appear. If it does not appear, set host to the workstation from another node. Log in to a privileged account and restart the server with the following command:

  ```
  $ @SYS$MANAGER:DECW$STARTUP RESTART
  ```

  Then log out. If you cannot log in again, halt the workstation and reboot the operating system.

  After the system reboots, check the server error log file SYS$MANAGER:DECW$SERVER_0_ERROR.LOG for possible problems. The server process runs with the name DECW$SERVER_0. It creates the SYS$MANAGER:DECW$SERVER_0_ERROR.LOG log file every time DECwindows Motif is activated. Always check this log file for error messages when you are experiencing problems. Always save the complete contents of the log file in case you must contact your Customer Support Center.

The screen seems to be frozen. Your keystrokes are not displayed and cursors do not blink.

- The DECwindows X server is performing a long operation for PEX, XIE, or PostScript.

  Wait for the operation to complete only if you are using one of the three server extensions. Some PEX operations might last minutes. XIE and PostScript should hold the server for no more than a few seconds before yielding.

- The DECwindows X server is waiting for network I/O.

  Wait 30 seconds. The DECwindows X server will wait 30 seconds after its transport buffers fill. If the I/O still does not complete, the server will terminate the client connection and continue without intervention.

- A client has grabbed the X server.
  - Identify the client application and respond to any outstanding output requests. If the client application does not respond, set host to the workstation from another node. Use the SHOW SYSTEM command to identify the process id number of the application. Stop the application with the following command:

    ```
    $ STOP/ID=process_id_number
    ```

    If the screen does not free itself at this time, continue with the next problem resolution.
  - The DECwindows X server has stopped working.
    - Press Return. A user name prompt should appear. If it does not appear, set host to the workstation from another node. Log in to a privileged account and restart the server with the following command:

      ```
      $ @SYS$MANAGER:DECW$STARTUP RESTART
      ```

      Then log out. If you cannot log in, halt the workstation and reboot the operating system.

      After the system reboots, check the server error log file SYS$MANAGER:DECW$SERVER_0_ERROR.LOG for possible problems. The server process runs with the name DECW$SERVER_0. It creates the SYS$MANAGER:DECW$SERVER_0_ERROR.LOG log file every time DECwindows Motif is activated. Always check this log file for error messages when you are experiencing problems. Always save the complete contents of the log file in case you must contact your Customer Support Center.
  - See the problem statement titled "The top of the screen goes black..." in Section C.2.6.

Index

A

Accounts
  dedicated, 4–12
Applications
  problems and solutions, C–15
Architecture, 1–2
AUTOGEN utility, 2–2

B

Backing store option, 3–27
Bitmap distribution format (BDF), 3–38
Bookreader directory, 4–8
  changing, 4–10
Bundled components, 1–5

C

Caching fonts, 3–17
  dynamically (VAX only), 3–18
Charter font, 3–39
Client, 1–3
Client/server model, 1–1
Client areas
  modifying, 4–7
Color database, 3–25
Command procedure
  DECW$DEVICE_xx.COM, 3–11
Command procedures
  used for startup, 2–2
Components, 1–4
  bundled with operating system, 1–5
Computer-based instruction files directory, 4–9
Cursor size, 3–17
Customization
  Bookreader directory, 4–10
  client areas, 4–9
  print formats, 4–22
  specific workstations, 3–2, 3–3
  startup, 4–5
  using DECW$PRIVATE_SERVER_SETUP file, 3–1
Customizing characteristics
  backing store, 3–27
  color database, 3–25
  keyboard, 3–21
  keymap, 3–21
  mouse, 3–24
  save under characteristics, 3–27
  Screen Saver, 3–25

D

Data files
  color database, 1–5
  font, 1–5
  keymap, 1–5
DECterm
  problems and solutions, C–22
DECW$BITONAL parameter, 3–13
DECW$BOOK symbol, 4–10
DECW$CHECK_PARAMS.COM command procedure, 2–5

DECW$CLIENT_ERROR_THRESHOLD parameter, 3–31
DECW$COLOR parameter, 3–12
DECW$CURSOR_SIZE parameter, 3–17
DECW$DEFAULT_KEYBOARD_MAP parameter, 3–21
DECW$DEFAULT_VISUAL_CLASS parameter, 3–11
DECW$DEVICE.COM command procedure, 2–4
DECW$DEVICE parameter, 3–11
DECW$DEVICE_*xx*.COM command procedure, 3–11
DECW$DISPLAY, 4–1
DECW$FM_INIT_NUM_CACHE_ROWS parameter, 3–18
DECW$FM_MAX_ROW_CACHE_SIZE parameter, 3–19
DECW$FM_MIN_ROW_CACHE_SIZE parameter, 3–19
DECW$FM_NO_CACHE_AUTOTUNING parameter, 3–19
DECW$FM_ROW_CACHE_EXTEND parameter, 3–20
DECW$FM_TIME_TO_SHRINK_CACHE parameter, 3–20
DECW$LOGICALS.COM command procedure, 2–4
DECW$MONITOR_DENSITY parameter, 3–13
DECW$MULTI_HEAD parameter, 3–10
DECW$PRIMARY_DEVICE parameter, 3–11
DECW$PRIVATE_APPS_SETUP.COM command procedure, 2–7
DECW$PRIVATE_APPS_SETUP.TEMPLATE , 4–9
DECW$PRIVATE_SERVER_SETUP.COM command procedure, 2–5
DECW$PRIVATE_SERVER_SETUP.TEMPLATE file, 3–1
DECW$RGBPATH parameter, 3–25
DECW$SERVER_BELL_BASE_VOLUME parameter, 3–22
DECW$SERVER_BUG_COMPATIBILITY parameter, 3–30
DECW$SERVER_CONNECT_LOG parameter, 3–33
DECW$SERVER_DEFAULT_BACKING_STORE parameter, 3–27
DECW$SERVER_DENSITY parameter, 3–17
DECW$SERVER_DISABLE_BACKING_STORE parameter, 3–28
DECW$SERVER_DISABLE_SAVE_UNDER parameter, 3–29
DECW$SERVER_DUMP parameter, 3–31
DECW$SERVER_ENABLE_KB_AUTOREPEAT parameter, 3–23
DECW$SERVER_ENQUEUE_LIMIT parameter, 3–10
DECW$SERVER_ERROR_THRESHOLD parameter, 3–32
DECW$SERVER_FILE_LIMIT parameter, 3–9
DECW$SERVER_KEYCLICK_VOLUME parameter, 3–22
DECW$SERVER_MAIN_KB_UPDOWN parameter, 3–23
DECW$SERVER_MOUSE_ACCELERATION parameter, 3–24
DECW$SERVER_MOUSE_THRESHOLD parameter, 3–24
DECW$SERVER_PAGE_FILE parameter, 3–9
DECW$SERVER_RETRY_WRITE_INTERVAL parameter, 3–15
DECW$SERVER_RETRY_WRITE_TIMEOUT parameter, 3–15
DECW$SERVER_SCALE_BITMAP_FONTS parameter, 3–21
DECW$SERVER_SCREEN_SAVER_INTERVAL parameter, 3–26
DECW$SERVER_SCREEN_SAVER_PREFER_BLANKING parameter, 3–25

DECW$SERVER_SCREEN_SAVER_
TIMEOUT parameter, 3–26
DECW$SERVER_TRANSPORTS parameter, 3–14
DECW$SERVER_WSDEF parameter, 3–8
DECW$SERVER_WSEXTENT parameter, 3–9
DECW$SERVER_WSQUOTA parameter, 3–8
DECW$STARTAPPS.COM command procedure, 2–7
DECW$STARTLIBS.COM command procedure, 2–6
DECW$STARTSERVER.COM command procedure, 2–5
DECW$STARTUP.COM command procedure, 2–3
DECW$STARTXTERMINAL.COM command procedure, 2–5
DECW$XPORT_SYNC_TIMEOUT parameter, 3–15
DECwindows system
See System
Dedicated accounts
creating with Session Manager command procedure, 4–12
creating with Session Manager executable file, 4–15
creating with Session Manager profile file, 4–17
Defining symbols, 3–2
Density
monitor, 3–13
Density of screen, 3–17
Device drivers, 1–5, 2–4
Devices
graphics, 3–10
WSA, 2–7, 4–2
Device setup
symbols for, 3–10
Directories
Bookreader, 4–10
computer-based instruction files, 4–9
example programs, 4–8
FileView command files, 4–8
Directories (cont'd)
keymap files, 4–8
online books, 4–8
public profile files, 4–9
utilities, 4–9
XDPS example programs, 4–8
Display
setting, 4–1
Displaying information
server logical name table, 3–44
server settings, 3–44
using XDPYINFO, 3–46
using XSET utility, 3–48
Display name, 4–1
Display server
See Server
Dual-head setups
on AXP systems, 3–36
on VAX systems, 3–34

E

Error condition handler, 3–30
Error log files, 3–30
Error reporting, 3–30
Error threshold, 3–31
Example programs directory, 4–8

F

File limit, 3–9
FileView
directory, 4–8
problems and solutions, C–8
Font access
problems and solutions, C–9
Font alias files, 3–16, 3–40
Font path, 3–16
Fonts
adding new, 3–38
bitmap distribution format (BDF), 3–38
caching (VAX only), 3–18
Charter, 3–39
creating new directories, 3–38
cursor size, 3–17

Fonts (cont'd)
determining size, 3–17
directories used in path, 3–16
file, 3–17
glyphs, 3–17
Lucida, 3–39
nonstandard fonts, 3–40
scaling bitmap (AXP only), 3–20
servers, 3–17
specifying, 3–38
third-party, 3–38
user-supplied, 3–17
Font server support, 3–40
Font setup, 3–16

I

Images
input driver, 1–5
shareable, 1–5
video driver, 1–5

K

Keyboard
changing default layout, 3–37
customizing, 3–37
model number, 3–37
problems and solutions, C–12
Keyboard setup, 3–21
Keymap files directory, 4–8
Keymap name
determining, 3–37
Keymaps
setting up, 3–21

L

License
problems and solutions, C–8
Logging in, 4–5
Logical names
defining, 4–23
for print formats, 4–22
overriding, 4–23
print formats, 4–22, 4–23
Login process, 4–5
Logo
customizing Start Session, 4–11
Lucida font, 3–39

M

Memory
adjusting for high usage, 3–7
defining, 3–8
physical pages, 3–9
requirements for backing store, 3–27
requirements for save under, 3–27
shared, 1–3
virtual, 3–9
Modifying client areas, 4–7
Modifying resource files, 4–20
Modifying Session Manager, 4–19
Monitor
blanking with Screen Saver, 3–25
defining density, 3–13
problems and solutions, C–12
support for color, 3–12
support for grayscale, 3–12
support for multihead, 3–10, 3–11
on VAX, 3–10
Mouse setup, 3–24
Multihead support, 3–10
on AXP systems, 3–10
on VAX systems, 3–11
Multiple font servers, 3–41
Multiple monitors, 3–10, 3–11

N

Network
problems and solutions, C–14
Node name, 4–2

O

OpenVMS/ULTRIX Connection software (UCX), 3–42

P

Parameters
  See System parameters
  for customization, 3–4
Pointer
  problems and solutions, C–12
Postinstallation tasks
  customizing the software, 4–22
Print dialog box
  logical names, 4–23
Print formats
  customizing, 4–22
  defining, 4–22
  logical names, 4–22, 4–23
Print queues, 4–23
Process
  starting up, 2–4
Process quotas, A–1
  ENQUEUE_LIMIT parameter, 3–10
  for AXP, 3–7
  for VAX, 3–7
  PAGE_FILE parameter, 3–9
  WSDEF parameter, 3–8
  WSEXTENT parameter, 3–9
  WSQUOTA parameter, 3–8
Profile files directory, 4–9

R

Resource files
  modifying, 4–20
Restarting DECwindows, 2–5
Restoring obscured windows, 3–27
Restoring the DECwindows logo, 4–12
RGB database, 3–25

S

Save under option, 3–27
Scaling bitmap fonts (AXP only), 3–20
Screen
  defining width and height, 3–13
  density, 3–17
Screen number, 4–2
Screen Saver characteristics, 3–25
Security
  with TCP/IP, 3–42
Security Options dialog box, 3–42
Server, 1–1
  asynchronous support to display, 1–3
  customization parameters, 3–4
  defining transport timeout, 3–15
  density value, 3–17
  displaying information
    about server, 3–44
    logical name table, 3–44
  error condition handler, 3–30
  error reporting, 3–30
  error threshold, 3–31
  private setup file, 3–2
  processing requests, 1–3
  shareable images, 1–5
  special requirements, 3–2
  synchronous support to application, 1–3
  using backing store, 3–27
  using save under, 3–27
  X11 display, 1–3
Server logical name table, 3–44
Server number, 4–2
Server process
  symbols, 3–7
Session Manager
  executable file, 4–15
  modifying behavior, 4–19
  problems and solutions, C–19
SET DISPLAY command, 4–2
Setting the display, 4–1
  examples, 4–2
Shared memory, 1–3
SHOW DISPLAY command, 4–4
Snapshot, 4–9
Starting DECwindows
  license verification, 4–6
  security list, 4–6
  SYLOGIN.COM, 4–6
Start Session dialog box
  problems and solutions, C–7

Startup
command procedures, 2–2
process, 2–2
sequence, 4–5
Symbols
Bookreader directory, 4–10
device setup, 3–10
server process, 3–7
writing with SHOW DISPLAY command, 4–4
SYSTARTUP_VMS.COM, 2–1
System
logicals used for startup, 2–1
overview, 1–1
running AUTOGEN, 2–2
startup file, 2–1
System parameters, 2–2, 2–5
establishing for server, A–1
recommended settings, A–2
recommended settings for AXP systems, A–4
recommended settings for VAX systems, A–3

T

TCP/IP transport, 3–42
Third-party fonts, 3–38
Third-party workstations
problems and solutions, C–27
Transport
DECnet, 1–3, 3–41
defining timeout, 3–15
installing the interface, 3–43
interface image, 3–43
local, 1–3, 3–41
sample setup, 3–36
shareable images, 1–5
specifying with symbol, 3–14
TCP/IP, 1–3, 3–41, 3–42
writing an interface, 3–43
Transport layer, 1–3
Troubleshooting techniques, C–1
Tuning, A–1
for high memory usage, 3–7

U

UAF
recommended settings, A–1
UNIX compatibility, 4–9
User authorization file (UAF)
See UAF
Utilities directory, 4–9

V

Video
device driver, 1–5

W

Window Manager
problems and solutions, C–9
Working set
defining, 3–8
Workstation
customizing, 3–3
setting up specific, 3–34
WSA devices, 2–7, 4–2, 4–3, 4–5

X

X11 display server
See Server
X Display Information utility (XDPYINFO), 3–46
XDPS example programs directory, 4–8
X protocol, 1–3
X server
problems and solutions, C–31
XSET utility, 3–48
X terminals
problems and solutions, C–27