

HP DECwindows Motif for OpenVMS

New Features

Order Number: AA-RT2CC-TE

January 2005

This manual describes new features and enhancements that pertain to the DECwindows Motif Version 1.5 software.

Revision/Update Information:	This manual supersedes the <i>HP DECwindows Motif for OpenVMS Alpha New Features</i> for Version 1.3-1.
Operating Systems:	OpenVMS I64 Version 8.2 OpenVMS Alpha Version 8.2
Software Versions:	DECwindows Motif for OpenVMS I64 Version 1.5 DECwindows Motif for OpenVMS Alpha Version 1.5

Hewlett-Packard Company
Palo Alto, California

© Copyright 2005 Hewlett-Packard Development Company, L.P.

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

Intel and Itanium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Java is a US trademark of Sun Microsystems, Inc.

UNIX is a registered trademark of The Open Group.

The X device is a trademark of X/Open Company Ltd. in the UK and other countries.

Printed in the US

ZK6663

Contents

Preface	ix
1 Introduction	
2 General User Features	
2.1 General DECwindows Motif Environment	2-1
2.1.1 Support for Internet Protocol Version 6 (IPv6) Transport	2-1
2.1.2 Extended File Specification (EFS) Support	2-2
2.1.2.1 File Selection Popup Window	2-2
2.1.2.2 New Desktop and the File Manager Application	2-2
2.1.2.3 Traditional Desktop and the FileView Application	2-2
2.1.2.4 The Programming Libraries	2-2
2.1.2.5 Translated Image Support (TIS) Library	2-2
2.1.3 Color Customizer Example Program	2-3
2.1.3.1 Supported Displays	2-3
2.1.3.2 Supported Applications	2-3
2.1.3.3 Building the Color Customizer on OpenVMS Systems	2-3
2.1.3.4 Running the Color Customizer	2-4
2.1.3.5 Modifying the DECW\$LOGIN.COM File	2-4
2.1.3.6 Command Interface Summary	2-4
2.1.3.7 Changing the Mapping Between Color Resources and Color Cells	2-5
2.1.3.8 Using the Customizer with DECterm Windows	2-6
2.1.3.9 Changing the Default Value of the Automatic Shadowing Toggle Button	2-6
2.1.3.10 Using the Customizer on Multihead Systems	2-6
2.1.3.11 Using the XSETROOT_CUST.EXE Demonstration Program	2-7
2.1.4 Drag-and-Drop Support	2-7
2.1.5 Tear-Off Menu Support	2-8
2.2 New Desktop Environment	2-8
2.2.1 Updated Welcome Message	2-8
2.2.2 Setting the File Manager Refresh Rate	2-8
2.2.3 Selecting Screens on Application Launch	2-8
2.2.4 Front Panel Icons Support MB3 Operations	2-9
2.3 Traditional Desktop Environment	2-9
2.3.1 Resource Added for DECwindows XUI Applications	2-9
2.4 Applications	2-9
2.4.1 CDA Viewer	2-9
2.4.1.1 Using the CDA Viewer to View Asian-Language Text	2-9
2.4.1.1.1 Specifying an Options File	2-10
2.4.1.1.2 Defining Logical Names	2-10
2.4.1.2 Converting Files That Contain Asian-Language Characters	2-11
2.4.1.3 Dynamic Font Support	2-12

2.4.1.4	Enhanced Display Performance	2-13
2.4.1.5	Pack and Unpack Applications	2-13
2.4.1.5.1	Pack Application Syntax	2-14
2.4.1.5.2	Unpack Application Syntax	2-14
2.4.1.5.3	Error Messages	2-15
2.4.1.6	New CDA Viewer Error Message	2-16
2.4.2	DECterm	2-16
2.4.2.1	New Default Font Sizes	2-16
2.4.2.2	Scrolling Using the Keyboard	2-16
2.4.2.3	ReGIS Input Cursors and Escape Sequences	2-17
2.4.2.4	Automatic Window Positioning	2-17
2.4.3	Window Manager	2-17
2.4.3.1	Overlay Support	2-17
2.5	Tools and Utilities	2-18
2.5.1	AccessX Keyboard Utility (accessx)	2-19
2.5.1.1	The AccessX Configuration File	2-20
2.5.1.2	Default Resource Settings	2-21
2.5.2	X Authority Utility (xauth)	2-22
2.5.2.1	The X Authority File	2-24
2.5.2.1.1	Format of an X Authority File Entry	2-25
2.5.2.1.2	Specifying an X Authority File	2-26
2.5.2.2	Invoking xauth and Entering Commands	2-26
2.5.2.3	IPv6 Considerations	2-27
2.5.2.4	Accessing Online Help	2-27
2.5.2.5	Creating an X Authority File	2-27
2.5.2.6	Displaying File Information	2-27
2.5.2.7	Viewing and Editing File Entries	2-28
2.5.2.8	Generating Authorization Keys	2-31
2.5.3	X Keyboard Compiler Utility (xkbcomp)	2-31
2.5.4	Window Dump to Print File (xpr) Utility	2-33

3 System Management Features

3.1	Installation and Upgrade Information	3-1
3.1.1	DECW\$STARTUP Log File	3-1
3.2	Security and Authorization	3-1
3.2.1	Support for ACME Subsystem (Alpha Only)	3-1
3.3	X Display Server Management	3-2
3.3.1	New Parameter for Enabling IPv6 Support in the Font Server	3-2

4 Programming Features

4.1	General Run-Time and Programming Environment	4-1
4.1.1	Multithreading Support	4-1
4.1.2	Binary Compatibility	4-1
4.1.2.1	Use of Asynchronous System Traps (ASTs)	4-2
4.1.2.2	Levels of Thread Safety and Concurrency	4-2
4.1.2.3	Enabling Support for Multithreading	4-4
4.1.2.4	Developing Applications with Thread-Aware Images	4-5
4.2	Application Programming	4-6
4.2.1	CDA Programming	4-6
4.2.1.1	Changes to the CDA Programming Interface	4-6
4.2.1.2	Changes to CDA External Reference Processing	4-7
4.2.1.3	Restructuring CDA Shareable Images	4-8

4.2.2	DECTerm Programming	4-8
4.2.2.1	ReGIS Input Cursors	4-8
4.2.2.2	Page-Movement Escape Sequences	4-9
4.3	OSF/Motif Toolkit (Xm) Programming	4-9
4.3.1	Drag-and-Drop Enabled Widgets	4-9
4.4	HP Extensions to Motif (DXm) Programming	4-10
4.4.1	SVN Widget Supports Extended Selection	4-10
4.4.2	DXmCSText Input Method Support	4-10
4.5	X Toolkit Intrinsics (Xt) Programming	4-11
4.5.1	New Functions Available with X11R6.6 Upgrade	4-11
4.5.2	Support for Easy Resource Configuration	4-12
4.5.3	New Option for CompositeClassExtensionRec	4-13
4.5.4	New Default Format for XtResolvePathname	4-13
4.5.5	XtAppMainLoop Routine	4-13
4.6	X Window System Library (Xlib)	4-14
4.6.1	IPv6 Considerations	4-14
4.6.2	New Functions Available with X11R6.6 Upgrade	4-14
4.6.3	Updated Client-Side Extension Library	4-15
4.6.4	Support for LCNs	4-16
4.6.5	Updated X11 Environment Variable Parsing	4-17
4.6.6	Additional Non-C Language Bindings Available with X11R6.6	4-17
4.6.6.1	CLOSE OM	4-17
4.6.6.2	CONTEXTUAL DRAWING	4-18
4.6.6.3	CONVERT CASE	4-18
4.6.6.4	DESTROY OC	4-18
4.6.6.5	DIRECTIONAL DEPENDENT DRAWING	4-18
4.6.6.6	DISPLAY OF OM	4-19
4.6.6.7	EXTENDED MAX REQUEST SIZE	4-19
4.6.6.8	INIT IMAGE	4-19
4.6.6.9	INIT THREADS	4-19
4.6.6.10	INTERNAL CONNECTION NUMBERS	4-20
4.6.6.11	LOCALE OF OM	4-20
4.6.6.12	LOCK DISPLAY	4-20
4.6.6.13	OPEN OM	4-20
4.6.6.14	PROCESS INTERNAL CONNECTION	4-21
4.6.6.15	REGISTER IM INSTANTIATE Callback	4-21
4.6.6.16	SET AUTHORIZATION	4-21
4.6.6.17	UNLOCK DISPLAY	4-21
4.6.6.18	UNREGISTER IM INSTANTIATE Callback	4-22
4.6.7	Support for Additional Fonts	4-22
4.6.7.1	Additional 75-dpi Fonts	4-23
4.6.7.2	Additional 100-dpi Fonts	4-27
4.6.7.3	Additional Common Fonts	4-32
4.6.7.4	Bitstream Speedo Scalable Fonts	4-34
4.6.7.5	Agfa Monotype TrueType Scalable Fonts	4-34
4.6.7.6	Adobe Type1 Fonts	4-35
4.6.8	UIDPATH Environment Variable	4-35
4.6.9	Client Side Extension Library	4-36
4.7	X Window System Extensions and Protocols	4-36

4.7.1	Additional X Display Server Extensions Supported with X11R6.6	4-36
4.7.1.1	Application Group Extension (XC-APPGROUP)	4-37
4.7.1.2	Big Requests Extension (BIG-REQUESTS)	4-38
4.7.1.3	Colormap Utilization Policy Extension (TOG-CUP)	4-38
4.7.1.4	Extended Visual Information Extension (EVI)	4-39
4.7.1.5	Low-Bandwidth X Extension (LBX)	4-39
4.7.1.6	Security Extension (SECURITY)	4-40
4.7.1.7	XC-MISC Extension	4-41
4.7.1.8	X Double Buffer Extension (DBE)	4-41
4.7.1.9	XINERAMA Extension	4-42
4.7.1.10	X Keyboard Extension (XKB)	4-42
4.7.1.11	X Synchronization Extension (SYNC)	4-43
4.7.2	Server Extensions Updated for X11R6.6	4-44
4.7.3	Inter-Client Exchange (ICE) Protocol Support	4-44
4.7.3.1	Multithreading Considerations	4-45
4.7.3.2	Differences from the Standard Implementation	4-46
4.7.3.3	IPv6 Considerations	4-47
4.7.4	X Session Management Protocol (XSMP) Support	4-47
4.7.4.1	Multithreading Considerations	4-48
4.7.4.2	Differences from the Standard Implementation	4-48
4.7.5	MIT Shared Memory Extension (MIT-SHM) Support	4-49
4.7.5.1	How to Use Shared Memory Extension	4-49
4.7.5.2	Using Shared Memory XImages	4-50
4.7.5.3	Using Shared Memory Pixmaps	4-53
4.7.6	X Image Extension (XIE) Support	4-53
4.8	Transport Programming	4-54
4.8.1	Support for the LAT Transport Interface Available	4-54
4.8.2	Support for the Logical Connection Number (LCN) Interface	4-54
4.8.2.1	LCN Functions	4-55
4.8.2.1.1	Initializing Thread Support	4-55
4.8.2.1.2	Allocating Connection Numbers	4-55
4.8.2.1.3	Querying Status and Signaling Input	4-55
4.8.3	LCN Routines	4-56
4.8.3.1	DECW\$LCN_ALLOCATE	4-56
4.8.3.2	DECW\$LCN_CLEAR_x_READY	4-57
4.8.3.3	DECW\$LCN_FREE	4-58
4.8.3.4	DECW\$LCN_SELECT	4-59
4.8.3.5	DECW\$LCN_SELECT_ONE	4-61
4.8.3.6	DECW\$LCN_SET_x_READY	4-63
4.8.3.7	DECW\$LCN_THREAD_INIT	4-64

Index

Tables

1-1	Directory of Features for DECwindows Motif Version 1.5	1-1
2-1	Asian Language Codes for Options Files	2-10
2-2	Logical Names for Specifying Text Encoding	2-11
2-3	Languages and Associated Basic Fonts	2-12
2-4	AccessX Keyboard Utility Options	2-20
2-5	Default AccessX Resource Settings	2-21
2-6	X Authority Utility Options	2-23

2-7	X Authority Utility Commands	2-23
2-8	xauth Transport Actions	2-27
2-9	Keyboard Compiler Options	2-32
2-10	Window Dump to Print File Options	2-34
4-1	Level of Thread Safety for DECwindows Motif Images	4-2
4-2	Multithreading Functions	4-4
4-3	New Header File Names	4-7
4-4	Names of Shareable Images	4-8
4-5	ReGIS Input Cursors—Cursor styles and Values	4-9
4-6	Drag-and-Drop Widgets	4-10
4-7	New Xt Functions Supported for X11R6.6	4-11
4-8	New Xlib Functions Supported for X11R6.6	4-14
4-9	Additional 75-dpi Fonts (.PCF File Extension)	4-23
4-10	Additional 100-dpi Fonts (.PCF File Extension)	4-27
4-11	Additional Common Fonts (.PCF File Extension)	4-32
4-12	Bitstream Speedo Scalable Fonts (.SPD File Extension)	4-34
4-13	Agfa Monotype TrueType Scalable Fonts (.TTF File Extension)	4-34
4-14	Adobe Type1 Scalable Fonts (.PCA File Extension)	4-35

Preface

This document describes the new features introduced with the HP DECwindows Motif for OpenVMS Version 1.5 (DECwindows Motif) software on the OpenVMS I64 and OpenVMS Alpha platforms. For information about how these features might affect your system, read the release notes before you install, upgrade, or use the DECwindows Motif Version 1.5 software.

The features in this manual are cumulative from DECwindows Motif for OpenVMS Version 1.0 and indicate any undocumented items that still pertain to the software. For previous features, a label within the description indicates when the feature was introduced.

Intended Audience

This manual is intended for system managers, users, and programmers who work with DECwindows Motif.

Document Structure

This manual is structured as follows:

- Chapter 1 provides an overview of the current release.
- Chapter 2 describes features of interest to general users of the DECwindows Motif software.
- Chapter 3 describes features related to system and network management.
- Chapter 4 describes features that support application and system programmers.

Related Documents

For additional information about OpenVMS or DECwindows Motif products and services, visit the following web site:

<http://www.hp.com/go/openvms>

Reader's Comments

HP welcomes your comments on this manual. Please send comments to either of the following addresses:

Internet	openvmsdoc@hp.com
Mail	Hewlett-Packard Company OSSG Documentation Group, ZKO3-4/U08 110 Spit Brook Rd. Nashua, NH 03062-2698

How To Order Additional Documentation

For information about how to order additional documentation, visit the following World Wide Web address:

<http://www.hp.com/go/openvms/doc/order>

Conventions

The following product names may appear in this manual:

- HP OpenVMS Industry Standard 64 for Integrity servers
- OpenVMS I64
- I64

All three names—the longer form and the two abbreviated forms—refer to the version of the OpenVMS operating system that runs on the Intel Itanium architecture.

All uses of DECwindows and DECwindows Motif refer to the HP DECwindows Motif for OpenVMS software; and all uses of X server and X display server refer to the DECwindows X11 Display Server. Additionally, all uses of DECwindows XUI (X User Interface) refer to the DECwindows product prior to DECwindows Motif Version 1.0.

The following typographic conventions may be 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.
<div>Return</div>	<p>In examples, a key name enclosed in a box indicates that you press a key on the keyboard. (In text, a key name is not enclosed in a box.)</p> <p>In the HTML version of this document, this convention appears as brackets, rather than a box.</p>
...	<p>Horizontal ellipsis points in examples indicate one of the following possibilities:</p> <ul style="list-style-type: none">• Additional optional arguments in a statement have been omitted.• The preceding item or items can be repeated one or more times.• Additional parameters, values, or other information can be entered.
.	Vertical ellipsis points indicate 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.
()	In command format descriptions, parentheses indicate that you must enclose the choices in parentheses if you choose more than one.
[]	In command format descriptions, brackets indicate optional elements. You can choose one, none, or all of the options. (Brackets are not optional, however, in the syntax of a directory name in an OpenVMS file specification or in the syntax of a substring specification in an assignment statement.)

[]	In command format descriptions, vertical bars separating items inside brackets indicate that you choose one, none, or more than one of the options.
{ }	In command format descriptions, braces indicate required elements; you must choose one of the options listed.
text style	<p>This text style represents the introduction of a new term or the name of an argument, an attribute, or a reason.</p> <p>In the HTML version of this document, this convention appears as <i>italic text</i>.</p>
<i>italic text</i>	Italic text emphasizes important information and indicates complete titles of manuals and variables. Variables include information that varies in system messages (Internal error <i>number</i>), in command lines (/PRODUCER= <i>name</i>), and in command parameters in text (where <i>dd</i> represents the predefined code for the device type).
UPPERCASE TEXT	Uppercase text indicates a command, the name of a routine, the name of a file, or the abbreviation for a system privilege.
Monospace type	<p>Monospace type indicates code examples and interactive screen displays.</p> <p>In the C programming language, monospace type in text identifies the following elements: keywords, the names of independently compiled external functions and files, syntax summaries, and references to variables or identifiers introduced in an example.</p>
-	A hyphen at the end of a command format description, command line, or code line indicates that the command or statement continues on the following line.
numbers	All numbers in text are assumed to be decimal unless otherwise noted. Nondecimal radices—binary, octal, or hexadecimal—are explicitly indicated.

Introduction

This chapter summarizes the features associated with the DECwindows Motif Version 1.5 software. Version 1.5 is a major release that includes support for the HP OpenVMS Industry Standard 64 for Integrity Servers (OpenVMS I64) platform, the Internet Protocol Version 6 (IPv6) transport, and ACME security (Alpha only).

Table 1–1 lists all the items added during this release and cross-references the sections in which they are described.

For a detailed description of the corrections, changes, and known problems associated with this release, see the *HP DECwindows Motif for OpenVMS Release Notes* manual.

Table 1–1 Directory of Features for DECwindows Motif Version 1.5

Title	Section
General User Features	
Support for Internet Protocol Version 6 (IPv6) Transport	Section 2.1.1
Tool and Utility Features	
xauth: IPv6 Considerations	Section 2.5.2.3
Installation and Upgrade Features	
DECW\$STARTUP Log File	Section 3.1.1
Security and Authorization Features	
Support for ACME Subsystem (Alpha Only)	Section 3.2.1
X Display Server Management Features	
New Parameter for Enabling IPv6 Support in the Font Server	Section 3.3.1
X Window System Library (Xlib) Features	
IPv6 Considerations	Section 4.6.1
X Window System Extension and Protocol Features	
ICE: IPv6 Considerations	Section 4.7.3.3

General User Features

This chapter provides information about new features that pertain to all users of DECwindows Motif.

2.1 General DECwindows Motif Environment

This section describes new features that are common to both the New Desktop and Traditional DECwindows Desktop environments.

2.1.1 Support for Internet Protocol Version 6 (IPv6) Transport

V1.5

DECwindows Motif now supports the Internet Protocol Version 6 (IPv6) transport in addition to the existing support for the Internet Protocol Version 4 (IPv4) transport. Included in this release is support for the transport names described in the X11R6.6 and X11R6.7 specifications published by the X.Org Foundation (formerly X.Org).

Use of the IPv6 transport is enabled with the new global symbol `DECW$IPV6_SUPPORT`, which can be defined in the startup file `SYS$MANAGER:DECW$PRIVATE_APPS_SETUP.COM`.

Support for the IPv6 transport affects many DECwindows Motif components—from the display device to functions in the Inter-Client Exchange (ICE) library. See one or more of the following manuals (or sections in this manual) to learn about IPv6 support and its use.

For information about...	See the...
X Window System support for transport names	X.Org Foundation web site (http://www.x.org)
Enabling IPv6 support, valid display name formats, host name translation, and using with font and proxy servers	<i>HP DECwindows Motif for OpenVMS Management Guide</i>
Impact on the SET DISPLAY command	<i>HP OpenVMS DCL Dictionary</i> and DCL online help
Impact on the xauth utility	Section 2.5.2.3
Impact on X Library (Xlib) routines	Section 4.6.1
Impact on ICE library routines	Section 4.7.3.3

General User Features

2.1 General DECwindows Motif Environment

2.1.2 Extended File Specification (EFS) Support

V1.2-5

In general, DECwindows Motif supports the Extended File Specifications (EFS) option available with OpenVMS. The interface either supports the new ODS-5 file names, as described in the following sections, or provides an error message indicating that the particular component does not support the new names.

2.1.2.1 File Selection Popup Window

The standard file selection popup window used by most DECwindows Motif applications fully supports ODS-5 style file names. The window supports deep directories, case preservation, and extended-length file names. The window supports entry of file names using the extended file name character and displays files using the extended file name character set using the circumflex character (^).

See the current OpenVMS documentation for more specific information about using deep directories and the extended file name character set.

2.1.2.2 New Desktop and the File Manager Application

With an ODS-5 volume, the File Manager application supports deep directories, case preservation, and extended-length file names. File Manager supports entry of file names using the extended file name character and displays files using the extended file name character set using the circumflex character (^).

Any custom File Manager command extensions must be modified to support EFS.

See the current OpenVMS documentation for more specific information about using deep directories and the extended file name character set.

2.1.2.3 Traditional Desktop and the FileView Application

The FileView application supports case preservation and creates files with the extended file name character set, but does not support any operations on its files.

FileView also supports extended-length file names and deep directory structures with the additional restriction that a total file specification cannot exceed 235 8-bit characters in length.

Exceeding this limit causes an error message for some menu items.

2.1.2.4 The Programming Libraries

The programming libraries fully support deep directories and extended-length file names. However, the libraries do not support case preservation or the extended file name character set.

2.1.2.5 Translated Image Support (TIS) Library

The translated image support (TIS) library has not been updated to support EFS.

2.1.3 Color Customizer Example Program

V1.2

The color customizer example program allows you to dynamically control the colors of your workstation environment. Window, icon, and window manager colors can be changed individually or as part of a palette switch. You can control mapping between resources and color cells, as well as the size and contents of the palette set. Also, automatic shadowing with the standard Motif shadowing algorithms is supported.

2.1.3.1 Supported Displays

The color customizer supports any display using pseudocolor or grayscale visuals. This includes most 4- and 8-plane workstation displays.

2.1.3.2 Supported Applications

The color customizer can affect the colors of any applications that use the current release of the X Toolkit Intrinsic (Xt) included with X Window System Version 11 Release 5 (X11R5) or greater. Applications from other vendors and versions of the library previous to X11R5 are unaffected.

Note

If the color customizer is used to control the colors of applications that have their own color customization dialog boxes (like the Session Manager, Window Manager, and DECwindows Mail), those application-specific color customization dialog boxes may not reflect the correct current color values while the customizer is running the application. This is normal; use the customizer instead of the application-specific dialog box to change these color values.

2.1.3.3 Building the Color Customizer on OpenVMS Systems

To build the color customizer on OpenVMS systems, perform the following steps:

1. Copy the files to a private directory. For example:

```
$ SET DEFAULT SYS$LOGIN
$ CREATE/DIRECTORY [.CUSTOMIZER]
$ SET DEFAULT [.CUSTOMIZER]
$ COPY DECW$EXAMPLES:CUSTOM.C []
$ COPY DECW$EXAMPLES:CUSTOM.UIL []
$ COPY DECW$EXAMPLES:CUSTOMIMAGE.DAT []
$ COPY DECW$EXAMPLES:XSETROOT_CUST.C []
$ COPY DECW$EXAMPLES:BUILD_CUSTOMIZER.COM []
```

2. Build the customizer using the following command:

```
$ @BUILD_CUSTOMIZER.COM
```

This command procedure creates the following output files:

```
CUSTOM.UID
CUSTOM.EXE
XSETROOT_CUST.EXE
```

General User Features

2.1 General DECwindows Motif Environment

2.1.3.4 Running the Color Customizer

To run the color customizer, perform the following steps:

1. Copy the files CUSTOM.UID and CUSTOM.EXE, which were created during the customizer build, to the directory where the customizer will be run. A typical location is the directory SYS\$LOGIN or the directory DECW\$USER_DEFAULTS.
2. Copy the files CUSTOM.DAT and DXMDEFAULTS.DAT from the directory DECW\$EXAMPLES to the directory DECW\$USER_DEFAULTS.
3. Run the executable file CUSTOM.EXE as follows:

```
$ RUN CUSTOM
```

Note

Only the colors of applications invoked after the customizer starts will be affected. For this reason, start the customizer as the first X application during the login process.

2.1.3.5 Modifying the DECW\$LOGIN.COM File

As noted in Section 2.1.3.4, the color customizer should be the first X application started during the login process. Do this by starting it as a subprocess from within the DECW\$LOGIN.COM file. Add a command to wait approximately 10 seconds between customizer startup and the startup of other applications.

For example, add the following lines to the DECW\$LOGIN.COM file:

```
$! Starting the color customizer
$ DISPLAY = F$LOGICAL("DECW$DISPLAY")
$ SPAWN/NOWAIT/OUTPUT='DISPLAY' RUN SYS$LOGIN:CUSTOM.EXE
$ WAIT 0:0:10
```

See *Using DECwindows Motif for OpenVMS* and *HP DECwindows Motif for OpenVMS Management Guide* for more information on modifying DECW\$LOGIN.COM.

2.1.3.6 Command Interface Summary

A box containing a list of available palettes is in the leftmost section of the Color Customizer window. Click on the desired palette to see the colors take affect.

Below the palettes are two arrays of colored buttons, representing the dynamically allocated color cells for normal and shadow colors. To find out what resources are affected by a color cell, click and hold the arrow button next to the color cell.

Hint

As a shortcut, you can click on the screen facsimile in the rightmost corner of the dialog box. If the portion you click on is colored by one of the resource values controlled by the customizer, the pop-up window for the appropriate color button is displayed.

General User Features

2.1 General DECwindows Motif Environment

To modify a single color cell, click on the corresponding color button. A colormix widget pops up; as you modify the color, these modifications are reflected in your workstation environment. Use the colormix widget reset button to return to the starting color at any time. You can also change the color cell you are modifying by clicking on a different color button while the colormix widget is displayed.

The automatic shadowing option causes shadow and select colors to be automatically updated when their corresponding background colors are changed. The standard Motif shadowing algorithms are used for these calculations.

Use the File menu to modify, add, and delete color palettes as follows:

- To modify an existing palette, select the palette, change the colors, and choose Save Palette from the File menu.
- To add a new palette, select an existing palette, modify the colors as necessary, and choose Save Palette As... from the File menu. A message box prompts you for the name of the new palette.
- To delete a palette, select the palette and choose Delete Palette from the File menu.

Changes made through the File menu automatically update the CUSTOM.DAT file, which contains the resource defaults.

The File menu Exit button causes the customizer application to exit. A warning dialog is displayed first. Note that the color cells allocated by the customizer (and used by the currently running applications) will be deallocated. After the customizer exits, if the colors of the currently running applications are not correct, the applications should be restarted to restore normal colors. Usually, there is no need to exit the color customizer; it is typically kept running at all times, like the Session Manager.

2.1.3.7 Changing the Mapping Between Color Resources and Color Cells

The file DXMDEFAULTS.DAT allows you to control how many dynamic color cells are allocated and what resources are affected. This file contains resource specifications like the following:

```
*background:    DXmDynamicWindowBackground
*foreground:    DXmDynamicWindowForeground
*topShadowColor: DXmDynamicWindowTopShadow
```

When the customizer is started, the file DXMDEFAULTS.DAT is written to a property on the root window. Any application that is subsequently run and that uses the correct version of Xt merges these resources with its normal resource database. Resource specifications in this file take precedence over specifications with equivalent resource names in other resource default files.

The resource values within the file DXMDEFAULTS.DAT have a special format. For each unique color value in this file that begins with the string "DXmDynamic", a color button is created in the color customizer. If the string "Shadow" is encountered in a name, the color button is placed in the shadow button box rather than the normal color button box. If a color value string ends with the suffix "Background", it is linked to any color buttons with identical prefixes and suffixes of "TopShadow", "BottomShadow", or "SelectColor" for purposes of automatic shadowing. If a color value named "DXmDynamicScreenBackground" is encountered, the color cell allocated is used by the customizer to set the root window background color.

General User Features

2.1 General DECwindows Motif Environment

You can edit the file DXMDEFAULTS.DAT and define resources to use the same color cells. You can have separate dynamic color cells, for scroll bar widgets or for your DECwindows Mail application, for example, by adding the following lines to the file DXMDEFAULTS.DAT:

```
Mail*background:      DXmDynamicMyMailBackground
Mail*foreground:      DXmDynamicMyMailForeground
Mail*topShadowColor:  DXmDynamicMyMailTopShadow
Mail*bottomShadowColor: DXmDynamicMyMailBottomShadow
```

Adding the previous lines to the file DXMDEFAULTS.DAT and restarting the customizer causes four new color cells to be allocated and four new color buttons to be added to the customizer interface. These buttons are assigned default color values (usually black or white) for each palette. These defaults can then be modified for each palette through the customizer interface.

Note

The text of the DXMDEFAULTS.DAT file is read and parsed by the color customizer. The parsing algorithm does not allow comments, incorrect spacing, or incorrect resource specifications. If this file or the CUSTOM.DAT resource file become corrupt, the customizer cannot start correctly. To resolve the problem, copy the versions of CUSTOM.DAT and DXMDEFAULTS.DAT from the DECW\$EXAMPLES directory into your login directory.

2.1.3.8 Using the Customizer with DECterm Windows

To change the colors of DECterm windows, copy the DECterm resource specifications from the file DXMDEFAULTS.DAT and add them to the DECterm resource defaults file DECW\$USER_DEFAULTS:DECW\$TERMINAL_DEFAULT.DAT. For example, add the following lines to the DECterm resource defaults file:

```
DECW$TERMINAL.main.terminal.background: DXmDynamicTerminalBackground
DECW$TERMINAL.main.terminal.foreground: DXmDynamicTerminalForeground
```

This allows the DECterm window colors to be customized with the color customizer.

2.1.3.9 Changing the Default Value of the Automatic Shadowing Toggle Button

The default value of the automatic shadowing toggle button is set using the Custom.autoShadow resource in the CUSTOM.DAT file as follows:

```
Custom.autoShadowing: False
```

The default value is True.

2.1.3.10 Using the Customizer on Multihead Systems

The color customizer affects only applications started on the same screen as the customizer. On most multihead systems, you can start a different color customizer for each screen and have a different palette in effect on each screen. On multihead systems using XINERAMA, a single instance of the color customizer affects all applications, since the screens function as a single logical screen.

General User Features

2.1 General DECwindows Motif Environment

The color customizer can be configured so that it is invoked once and affects all applications regardless of where they are started. This mode is invoked by modifying the Custom.multiScreen resource in the CUSTOM.DAT file as follows:

```
Custom.multiScreen: True
```

The default value is False.

2.1.3.11 Using the XSETROOT_CUST.EXE Demonstration Program

The XSETROOT_CUST.EXE demonstration program, created during the customizer build, is a modified version of the X utility xsetroot that is used to set a bitmap on the root window. The XSETROOT_CUST.EXE program uses DXmDynamicScreenBackground and DXmDynamicScreenForeground as the background and foreground colors of the specified bitmap. If your DXMDEFAULTS.DAT file contains entries for these two dynamic colors, then use the customizer to dynamically modify the colors of your bitmap.

For example:

```
$ XSETROOT_CUST :== "$SYS$LOGIN:XSETROOT_CUST.EXE"
$ XSETROOT_CUST -BITMAP your_xbm_file.XBM
```

2.1.4 Drag-and-Drop Support

V1.2

The drag-and-drop feature lets you move or copy screen objects. This feature is provided primarily for programmers who choose to incorporate drag-and-drop into their applications. For example, you can move text from a text entry area and paste it elsewhere.

All DECwindows Motif applications except Notepad support the drag-and-drop feature. DECwindows Mail supports the drag-and-drop feature in all windows except the main message area, where DECwindows Mail has its own drag-and-drop; you can use MB2 to move messages around with the SVN interface.

To drag and drop text into a new location:

1. Select the text to be copied or moved with MB1.
2. To move the text, press and hold MB2; to copy the text, press and hold Ctrl/MB2.

A move or copy icon appears.

3. Drag the icon to the location where you want to drop the text and release MB2.

If the object is highlighted as you drag the icon across it, you can drop the text into that location.

For a list of the widgets that support drag-and-drop functionality, see Section 4.3.1.

General User Features

2.1 General DECwindows Motif Environment

2.1.5 Tear-Off Menu Support

V1.2

Most DECwindows Motif applications allow you to tear off pull-down and popup menus. Tear-off menus let you keep frequently used menus displayed without repeatedly pulling them down or popping them up.

To tear off a menu:

1. Display a pull-down or popup menu.

If the menu is a tear-off menu, a dotted line is displayed at the top of the menu.

2. Click on the dotted line with MB1.

The menu remains active until it is closed or until the parent application is closed.

To close a tear-off menu:

1. Click on the Window menu button in the tear-off menu.
2. Choose the Close menu item.

2.2 New Desktop Environment

This section describes new features related to the New Desktop environment.

2.2.1 Updated Welcome Message

V1.2-6

The welcome message in the Login Screen now displays a host name regardless of the transport. If the DECnet transport is configured, the DECnet host name is displayed. If the TCP/IP transport is configured, the TCP/IP host name is displayed. If neither transport is configured, a default message of "Welcome to OpenVMS" is displayed.

2.2.2 Setting the File Manager Refresh Rate

V1.2-6

You can now specify that the File Manager periodically update its view on the New Desktop by adjusting the `Dtfile.rereadTime` setting in the `DTFIELDAT` resource file. The value of this setting represents the seconds elapsed between checking for changes in the viewed directories. Note that this setting does not work when viewing search lists.

2.2.3 Selecting Screens on Application Launch

V1.2-5

You can graphically select the screen on which a new application is displayed when launched from either the Front Panel, the File Manager, or the Application Manager. By default, the new application appears on the current screen (that is, the screen containing the mouse pointer). The feature allows you to drop an application icon from the File Manager or Application Manager onto one of the numbered screen controls of the Set Default Screen window. This starts the application on the selected screen without changing the current screen.

The Set Default Screen window is activated by selecting the “Set Default Screen” application in the Application Manager’s Desktop Tools folder. You can start a separate instance of the Set Default Screen window on each screen. The highlighting of the default screen is synchronized across all instances of the Set Default Screen window.

2.2.4 Front Panel Icons Support MB3 Operations

V1.2–4

The New Desktop Front Panel supports mouse button 3 (MB3) operations. When the cursor is placed over a Front Panel icon and you press MB3, a subpanel or menu appears. The menu items are as follows:

- Top item—The label of the menu.
- Second item—The application that starts if you single click on the icon.
- Third item—Add or delete a subpanel, depending on whether a subpanel already exists for the control panel.

Caution

If the third item is “Delete Subpanel”, this change is difficult to reverse without reinstalling the kit.

2.3 Traditional Desktop Environment

This section describes new features related to the traditional DECwindows desktop environment.

2.3.1 Resource Added for DECwindows XUI Applications

V1.0

The resource `Mwm*useDECMode` has been added to allow previous versions of DECwindows XUI applications to behave correctly with the Motif Window Manager. In particular, this resource is used to control focus, window placement, multiline icons, and the window’s initial state (normal or minimized).

2.4 Applications

The following sections describe new features related to specific DECwindows Motif applications.

2.4.1 CDA Viewer

This section describes features related to the Compound Document Architecture (CDA) Viewer application.

2.4.1.1 Using the CDA Viewer to View Asian-Language Text

V1.2–3

You can use the CDA Viewer in two ways to view text files that contain Asian characters:

- Specify an options file to the CDA Viewer application.
- Define logical names at the DCL command level or in a LOGIN.COM file.

General User Features

2.4 Applications

Refer to the *DECwindows Motif for OpenVMS Applications Guide* for information about using the CDA Viewer.

2.4.1.1.1 Specifying an Options File Specify an options file by including a one-line entry in the file in the following format:

```
TEXT TEXT_ENCODING text_encoding_value
```

- TEXT is the format.
- TEXT_ENCODING is the option you specify to CDA.
- *text_encoding_value* is the value of the codeset. (See Table 2–1 for a list of values.)

Table 2–1 shows the languages, codesets, and text-encoding values.

Table 2–1 Asian Language Codes for Options Files

Language	Codeset	Text Encoding Value
Japanese	DEC Kanji	DEC_KANJI
Japanese	Super DEC Kanji	SDECKANJI
Traditional Chinese	DEC Hanyu	DEC_HANYU
Simplified Chinese	DEC Hanzi	DEC_HANZI
Korean	DEC Korean	DEC_HANGUL

The following table shows examples of one-line entries.

Options File	One-Line Entry
HANYU.CDA\$OPTIONS	TEXT TEXT_ENCODING DEC_HANYU
HANZI.CDA\$OPTIONS	TEXT TEXT_ENCODING DEC_HANZI
HANGUL.CDA\$OPTIONS	TEXT TEXT_ENCODING DEC_HANGUL

To view the `EXAMPLES_CUSTOMERS.TXT` file that contains Japanese text in DEC Kanji, use your editor to create an options file called `KANJI.CDA$OPTIONS`. Add the following one-line entry to the file:

```
TEXT TEXT_ENCODING DEC_KANJI
```

When you access the file through the Options File dialog box with the CDA Viewer, the `EXAMPLES_CUSTOMERS.TXT` file is viewable in the DEC Kanji codeset (Japanese language).

2.4.1.1.2 Defining Logical Names The second option to enable viewing files in Asian languages is to specify the text file and encoding value by defining two logical names:

- `DDIF$READ_TEXT_GL`
- `DDIF$READ_TEXT_GR`

Table 2–2 shows the logical names and associated encoding values.

Table 2–2 Logical Names for Specifying Text Encoding

DDIF\$READ_TEXT_GL	DDIF\$READ_TEXT_GR	Encoding Value
LATIN1	MCS	MCS
LATIN1	LATIN1	ISO Latin–1
LATIN1	KATAKANA	ASCII–Kana
LATIN1	KANJI	DEC Kanji
ROMAN	MCS	Roman–MCS
ROMAN	LATIN1	Roman
ROMAN	KANJI	Roman–Kanji
ROMAN	KATAKANA	Roman–Kana
LATIN1	HANZI	DEC Hanzi
LATIN1	HANGUL	DEC Hangul
LATIN1	HANYU	DEC Hanyu

You can define the logical names on the DCL command line or in your LOGIN.COM file. For example:

```
$ DEFINE DDIF$READ_TEXT_GL LATIN1
$ DEFINE DDIF$READ_TEXT_GR KANJI
```

Note that this example defines the text encoding for DEC Kanji (see Table 2–2).

2.4.1.2 Converting Files That Contain Asian-Language Characters

V1.2–3

You can convert an Asian-language text file to another format by specifying an options file or by defining the logical names DDIF\$READ_TEXT_GL and DDIF\$READ_TEXT_GR as discussed in Section 2.4.1.1.1 and Section 2.4.1.1.2.

The format for converting a document from TEXT to another format is as follows:

```
$ CONVERT/DOCUMENT/OPTION=language.CDA$OPTIONS filename.TXT/FORMAT=TEXT -
_$ filename.output_extension/FORMAT=output_format
```

For example, to convert a traditional Chinese language text file to a DDIF file, enter the following command line:

```
$ CONVERT/DOCUMENT/OPTION=HANYU.CDA$OPTIONS -
_$ GUIDELINES_PERSONNEL.TXT/FORMAT=TEXT GUIDELINES_PERSONNEL.DDIF
```

Note that this command line does not include the /FORMAT=DDIF qualifier; DDIF is the default.

The output file, GUIDELINES_PERSONNEL.DDIF, contains language data.

General User Features

2.4 Applications

You can also create Asian language PostScript files from a DDIF, DTIF, or text (ASCII) file. For example, to convert a DDIF file to PostScript (.PS) format, enter the following command:

```
$ CONVERT/DOCUMENT filename.DDIF filename.PS/FORMAT=PS
```

Note

Convert only DDIF and DTIF files that contain language data to Asian language PostScript format.

When you print an Asian language PostScript file on a PostScript printer, ensure that the required language fonts are available on the printer. Otherwise, the PostScript file defaults to a basic set of fonts. If these fonts do not exist, the PostScript file defaults to Courier fonts. Table 2–3 shows the languages and their associated basic fonts.

Table 2–3 Languages and Associated Basic Fonts

Language	Basic Fonts
Japanese	Ryumin-Light-EUC-H or Ryumin-Light-Hankaku
Hanyu	Sung-Light-CNS11643, Sung-Light-DTSCS
Hangul	Munjo
Hanzi	XiSong-GB2312-80

Note

Vertical writing is not supported by the CDA converters. All vertical text is printed horizontally.

2.4.1.3 Dynamic Font Support

V1.2

As well as supporting a static-table for the fonts supported by the DECfonts Typeface Collection Version 1.2, the CDA Run-Time Services includes support for dynamic font lookup. This enables the CDA Viewer to use new fonts as they are installed on the system.

Dynamic font support is implemented using the WRITE\$FONTS.INI file, which you can maintain using the Font utility provided with either DECwrite or DECpresent. If a document contains a font not found in the static tables, the CDA Viewer tries to open the WRITE\$FONTS.INI file and search for the font. If the font is not found or if the system does not contain a WRITE\$FONTS.INI file, the viewer uses a fallback font.

The WRITE\$FONTS logical name references an initialization file used to provide font definitions to the CDA Viewer.

The default location for the WRITE\$FONTS.INI file is SYS\$LIBRARY, but, if the logical name WRITE\$FONTS is defined, the CDA Viewer uses the logical name definition to search for the WRITE\$FONTS.INI file.

Full path support is included, so any of the following definitions are valid:

WRITE\$FONTS Logical Name	Resulting File
Undefined	SYS\$LIBRARY:WRITE\$FONTS.INI
DISK:[DIRECTORY]	DISK:[DIRECTORY]WRITE\$FONTS.INI
SYS\$LOGIN:	SYS\$LOGIN:WRITE\$FONTS.INI
.TMP	SYS\$LIBRARY:WRITE\$FONTS.TMP
DISK:[DIRECTORY]FILE	DISK:[DIRECTORY]FILE.INI

2.4.1.4 Enhanced Display Performance

V1.2

The current version of CDA Run-Time Services includes a performance enhancement that decreases the time it takes to display the first page of a CDA document. Other applications that use the CDA Viewer to view documents (for example, DECwindows Mail) also benefit from this enhancement.

The CDA Viewer enables this performance enhancement feature by default. You can disable the feature as follows:

```
$ DEFINE CDA_QUICK_FIRST_PAGE FALSE
```

The CDA Viewer might not display some documents correctly when this feature is enabled. If you encounter such a problem, disable the feature and invoke the CDA Viewer again.

2.4.1.5 Pack and Unpack Applications

V1.2

CDA Run-Time Services includes two standalone applications that can be used for transferring CDA documents across a network. The CDA Pack application packages a CDA document along with all of its externally referenced files into a single file that can be copied between systems or mailed to other users. The CDA Unpack application reads a file that is packaged by the CDA Pack application and creates a copy of the original document file and all its externally referenced files.

These applications allow you to copy CDA documents between systems without copying externally referenced files separately or correcting external file reference information after copying documents.

To use these applications, add the following lines to your LOGIN.COM file (or add the lines to the SYS\$MANAGER:SYLOGIN.COM file):

```
$ PACK == "$SYS$SYSTEM:CDA$PACK.EXE"  
$ UNPACK == "$SYS$SYSTEM:CDA$UNPACK.EXE"
```

These lines enable you to use the symbols PACK and UNPACK to invoke the Pack and Unpack applications, respectively.

General User Features

2.4 Applications

2.4.1.5.1 Pack Application Syntax The CDA Pack application creates a single output file that contains the contents of a .DDIF or .DTIF input file. The single output file also includes the files that are referenced by the .DDIF or .DTIF input file.

The format of the PACK command is as follows:

```
$ PACK input-file-spec output-file-spec
```

The following sections explain the format of the PACK command.

input-file-spec

Specifies the name of the primary .DDIF or .DTIF input file.

output-file-spec

Specifies the name of the output file that is created by the PACK application. If you do not specify a device or directory, the output file is created in the current default directory.

Qualifiers:

/[NO]SKIP_MISSING

Controls whether the Pack application continues processing if it cannot find one or more of the files that are listed as external references in the input file. The names of any missing files are sent to SYS\$ERROR when the Pack application is completed. If you specify /NOSKIP_MISSING, the Pack application does not create an output file if any of the externally referenced files are missing. The default is /SKIP_MISSING.

/[NO]CONTROLLED_COPY

Controls whether the output file includes only those external references that specify COPY_REFERENCE as the value of the ERF_CONTROL item in the input file. If you specify /NOCONTROLLED_COPY, the Pack application includes all referenced files, regardless of the value of the ERF_CONTROL item. The default is /NOCONTROLLED_COPY.

/ALWAYS_ENCODE

Controls whether an output file is created when there are no external references in the input file, or if none of the externally referenced files are found. The default is not to create an output file in these cases. If an output file is not created for these reasons, the Pack application returns the CDA_W_NOOUTFIL status code.

For example:

```
$ PACK MYFILE.DDIF TEST.PACK
```

2.4.1.5.2 Unpack Application Syntax The CDA Unpack application unpacks an input file created by the Pack application. The output files are the .DDIF or .DTIF file that is packed by the Pack application, as well as a file for each external reference in the .DDIF or .DTIF file. The Unpack application sends a list of created files to SYS\$ERROR.

The format of the UNPACK command is as follows:

```
$ UNPACK input-file-spec
```

The following sections explain the format of the UNPACK command.

input-file-spec

Specifies the name of the input file that is created by the Pack application.

Qualifier:

/OUTPUT=output-file-spec

Specifies the file name and location of the files created by the Unpack application.

If you specify an output file name without a directory name, the Unpack application creates the main .DDIF or .DTIF file with the file name you specify in the current default directory. It also creates all externally referenced files in the current default directory.

If you specify a directory name without a file name, the Unpack application creates the main .DDIF or .DTIF file and all externally referenced files in the specified directory. The main .DDIF or .DTIF file has the same name as the file packed by the Pack application.

If you specify a directory name and a file name, the Unpack application creates the main .DDIF or .DTIF file and all the externally referenced files, in the specified directory. The main .DDIF or .DTIF file has the file name you specify.

For example:

```
$ UNPACK TEST.PACK
Output file DISK$:[SMITH]MYFILE.DDIF created.
Output file DISK$:[SMITH]FIGURE_1.DDIF created.

$ UNPACK TEST.PACK/OUTPUT=[SMITH.UNPACK]
Output file DISK$:[SMITH.UNPACK]MYFILE.DDIF created.
Output file DISK$:[SMITH.UNPACK]FIGURE_1.DDIF created.

$ UNPACK TEST.PACK/OUTPUT=[SMITH.UNPACK]NEW_FILE.DDIF
Output file DISK$:[SMITH.UNPACK]MYFILE.DDIF renamed NEW_FILE.DDIF
Output file DISK$:[SMITH.UNPACK]FIGURE_1.DDIF created.
```

2.4.1.5.3 Error Messages This section describes messages associated with the CDA Pack and Unpack applications.

FILESPEC, Missing filespec: *file-name*

Severity: Informational

Explanation: The Pack application cannot locate an external file included as an external reference in the .DDIF or .DTIF file or in one of the files referenced in the .DDIF or .DTIF file.

NOOUTFIL, No output file was created.

Severity: Warning

Explanation: The Pack application cannot find external references in the .DDIF or .DTIF document to be packed, and you did not specify the /ALWAYS_ENCODE qualifier.

OUTFILE, Output file created: *file-name*

Severity: Informational

Source: CDA_UNPACK

Explanation: The Unpack application created the specified file while unpacking a file created by the Pack application.

General User Features

2.4 Applications

2.4.1.6 New CDA Viewer Error Message

V1.2

The CDA Viewer issues the following message if it is unable to create the application context:

DRMCTXFAIL, DVR could not create application context, aborting

Level: Error

Explanation: The CDA Viewer ends because an attempt to create the application context using the Resource Manager failed, which is usually caused by insufficient memory.

User Action: Reduce the system load and start the application again.

2.4.2 DECterm

This section describes features that pertain to the DECterm application.

2.4.2.1 New Default Font Sizes

V1.2–4

In previous releases, the default DECterm font size for the “big” and “little” fonts were chosen by point size. This depended on whether 75 or 100 dpi fonts were installed first in the font path. Starting with DECwindows Motif for OpenVMS Version 1.2–4, DECterm chooses its default fonts by pixel size.

The following table shows the two behaviors.

Default Font Size (in pixels)	Prior Releases		Since Version 1.2–4	
	100	75 ¹	100	75
Big font	25	18	18	18
Little font	18	14	14	14

¹75 dpi fonts or 100 dpi fonts on a 15-inch monitor

In addition, on 100-dpi displays DECterm now uses the big font by default. This results in DECterm using the same font size (18 pixels) on 100-dpi displays as it did in prior releases. On displays that are less than 325 mm wide, DECterm also now uses a bigger default font.

Follow these steps to use the same font as in previous releases (14 pixels):

1. From the Options menu, select the Window... item.
2. Choose Little Font. Click on Apply.
3. If the new default font is too small, click on the Big Font button and change the pixel size in the Other text entry field from 18 to 25.

2.4.2.2 Scrolling Using the Keyboard

V1.2–3

You can now scroll through text using the keyboard by pressing the Ctrl key and arrow keys or by pressing the Prev or Next key on the editing keypad.

2.4.2.3 ReGIS Input Cursors and Escape Sequences

V1.2–3

The DECterm application supports all ReGIS input cursors:

- Crosshair
- Diamond
- Rubber-band line
- Rubber-band rectangle

For a shape other than the diamond cursor when n is equal to 1, define the logical name `DECW$TERM_REGIS_CURSOR` as one of the numbers defined in the `SYS$LIBRARY:DECW$CURSOR` file.

V1.2

The following escape sequences are supported by DECterm:

- All page movement sequences (NP, PP, PPA, PPB, and PPR).
- One rectangular area operation sequence (DECCRA).
- The DECLFKC sequence.
- The ReGIS command `S(C(In))` supports the rubber-band rectangle cursor and the diamond cursor.

See Section 4.2.2.1, ReGIS Input Cursors for additional information about escape sequences in DECwindows Motif software.

2.4.2.4 Automatic Window Positioning

V1.1

A resource has been defined to manage repositioning a DECterm window when a resize operation forces part of the window off the screen. If a DECterm window is enlarged by using the Options/Window dialog box or by entering a `SET TERMINAL/PAGE= nn` or `SET TERMINAL/WIDTH= nn` command, the controller moves the newly resized DECterm window so that it can be viewed in its entirety. If you prefer DECterm not to move, add the following line to your `DECW$TERMINAL_DEFAULT.DAT` file:

```
DECW$TERMINAL.main.terminal.autoAdjustPosition: off
```

2.4.3 Window Manager

This section describes features that pertain to the Motif Window Manager application.

2.4.3.1 Overlay Support

V1.2–3

The latest version of the Motif Window Manager (MWM) supports overlays and utilizes additional planes of memory, which are available on some 3D graphics accelerators. The Window Manager places borders and banners for all the windows into these extra planes of memory and thereby reduces the number of expose events for your applications that use overlays.

You may need to modify your existing applications that use overlays to avoid potential problems with the colormap. HP recommends that you set up your system to share the overlay colormap with the Window Manager, as the hardware supports only one colormap for the overlay planes.

General User Features

2.4 Applications

See the associated documentation for your 3D graphic accelerator to determine if overlays are supported.

Setting Up the Overlay Colormap

To modify your applications to share the overlay colormap with the Window Manager, query the server property name `SERVER_OVERLAY_COLORMAPS`. When you make the query, the server returns the 32-bit value for the overlay Colormap ID.

To set up your system to share the overlay colormap with the Window Manager, edit the files `SYS$COMMON:[VUE$LIBRARY.SYSTEM]VUE$MWM.COM` and `SYS$COMMON:[SYSMGR]DECW$MWM.COM`. Change the following line in each file:

```
$ mwm -multiscreen
```

Edit this line by adding the `-Overlay` command-line option as follows:

```
$ mwm -multiscreen "-Overlay"
```

Note that if you create and install your own colormap, the following problems can result:

- Colors flash on the screen when the colormap is changed.
- Border and banner colors also change when you change the colors of your colormap.

Restrictions

The following restrictions apply when you enable the Window Manager to use overlays:

- The Window Manager supports only single-screen systems and does not function correctly with multiple graphics devices (multihead).
- If you select a Matte Size value other than “None” from the Window Manager options list, the Matte color may not be correct; that is, the color does not match the selection and is occasionally transparent.
- If you select “Show feedback when moving or resizing windows” from the Workspace Options menu, the window with the feedback information causes expose events.
- When you move windows by showing the outline of the window, the outline appears to go below the window borders and banners.
- Window borders are occasionally and randomly displayed in clear or black. If this problem occurs, select the restart option from the Workspace menu to restart the Window Manager.

2.5 Tools and Utilities

The following sections describe new features related to specific X Window System utilities ported to DECwindows Motif.

2.5.1 AccessX Keyboard Utility (accessx)

V1.3

The AccessX Keyboard utility (accessx) is a client application that enables you to set one or more AccessX keyboard enhancements available with the X Keyboard extension (XKB). These enhancements make it easier for users with disabilities to interact with workstation input devices (keyboard and mouse).

Specifically, AccessX enhancements for XKB offer the following capabilities:

- **Sticky Keys** – Allows you to perform multikey operations with one hand, one finger, or a mouth stick. You can use this feature to enter certain uppercase letters and punctuation characters without having to hold down the Shift key. This feature also makes it easier to enter control sequences, such as Ctrl/C.
- **Mouse Keys (also known as Dead Mouse)** – Lets you map actions that you would perform with a mouse to keys on the numeric keyboard or other keys that you specify. With this feature, you can use one finger or a mouth stick to move the cursor to different areas of the screen, manipulate menus, and select, cut, and paste text.
- **Toggle Keys** – Provides audio feedback when the Shift Lock (Caps Lock) key is pressed. This feature helps users who might have difficulty seeing the keyboard light indicator for the Shift Lock key or users who are using a keyboard that does not provide light indicators for any keyboard settings.
- **Repeat Keys** – Allows you to adjust the auto-repeat keyboard mechanism speed or to turn it off entirely. With this feature turned on, you can set your keyboard so that holding down a key for a longer than average time does not cause a repeat entry of that character.
- **Slow Keys** – Makes the keys less likely to respond when brushed accidentally. With this feature turned on, the computer accepts only keystrokes that are held for a certain length of time. The computer ignores light keystrokes that are held only for a moment.
- **Bounce Keys** – Eliminates the problem of pressing a key and then accidentally pressing it again before moving to another key. You can set this feature to tell the computer not to process a second pressing of a key unless a certain length of time elapses between each pressing.
- **Time Out** – Shuts off the AccessX features (except for Repeat Keys) on a workstation after a specified period of time. If you are sharing a workstation and have set AccessX features, the settings are turned off automatically before the next use. To retain the AccessX settings at all times, you can turn off the Time Out feature.

The settings for these capabilities are stored as X resource specifications available from an **AccessX configuration file**. This utility reads the appropriate file (either for the client or server) and adjusts the resource settings depending upon the changes you make.

See Section 2.5.1.1 and Section 2.5.1.2 to learn more about the configuration file and its default values. For information about enabling and using XKB and X Keyboard keymaps, see the *HP DECwindows Motif for OpenVMS Management Guide*.

General User Features

2.5 Tools and Utilities

To run this utility, define `accesssx` as a foreign command, and copy the UID file to `DECW$USER_DEFAULTS`:

```
$ accesssx :== "$DECW$EXAMPLES:ACCESSX.EXE"
$ COPY DECW$EXAMPLES:ACCESSX.UID DECW$USER_DEFAULTS:ACCESSX.UID
```

The command format for `accesssx` is as follows:

```
$ accesssx [-options...]
```

You can run `accesssx` by entering the command at the DCL prompt, with or without options. The options, described in Table 2–4, allow you to specify which configuration file to load and choose whether to display the status of the editing session. If no options are specified, the client configuration file is loaded by default.

Table 2–4 AccessX Keyboard Utility Options

Options	Description
-client	Instructs <code>accesssx</code> to load and use the custom settings in the client configuration file. This is the default option.
-server	Instructs <code>accesssx</code> to load and use the default, system-wide settings maintained in the server configuration file.
-status	Displays the status window when <code>accesssx</code> is started.
-vmods	Uses the names for virtual modifiers in the status window. For example, with the -vmods option, <code>accesssx</code> displays NumLock instead of Mod5.

Once invoked, the `accesssx` dialog box is displayed, which lets you select the AccessX features that you want to enable. It also provides a test area that allows you to preview your settings before applying them to a DECwindows Motif session.

For detailed information on the controls and menu options in the dialog box, see the online help available from the Help menu option.

2.5.1.1 The AccessX Configuration File

The AccessX configuration file is an X resource file used to store and load specific AccessX settings for the client. The default client configuration file is located in `DECW$USER_DEFAULTS:ACCESSX.DAT` and is created the first time you choose Save Settings.

You can choose to edit the file directly or use the `accesssx` utility (with the **-client** or **-server** option) to modify the settings. Note that any comments inserted in this file are deleted when you perform a save settings action.

The format for entries in this file follow the standard X resource specification format, which is:

**resource:value*

For example:

```
*BounceKeysToggle.set:False
```

Since an application can consist of a combination of input widgets (such as, push buttons and a scroll bar), you can use the widget class and name identifiers to create additional resource specifications.

2.5.1.2 Default Resource Settings

Table 2–5 lists the AccessX resources and their default values. These are default settings maintained by the server and also represent the initial settings in the client configuration file, prior to modification.

Table 2–5 Default AccessX Resource Settings

Resource	Value
*BounceKeysToggle.set	False
*EnableAccessXToggle.set	True
*KRGDebounceScale.decimalPoints	1
*KRGDebounceScale.maximum	40
*KRGDebounceScale.minimum	1
*KRGDebounceScale.value	3
*KRGRepeatDelayScale.decimalPoints	2
*KRGRepeatDelayScale.maximum	400
*KRGRepeatDelayScale.minimum	1
*KRGRepeatDelayScale.value	66
*KRGRepeatRateScale.decimalPoints	2
*KRGRepeatRateScale.maximum	400
*KRGRepeatRateScale.minimum	1
*KRGRepeatRateScale.value	4
*KRGSlowKeysDelayScale.decimalPoints	1
*KRGSlowKeysDelayScale.maximum	40
*KRGSlowKeysDelayScale.minimum	1
*KRGSlowKeysDelayScale.value	3
*MouseKeysToggle.set	False
*MouseAccelScale.decimalPoints	1
*MouseAccelScale.maximum	40
*MouseAccelScale.minimum	1
*MouseAccelScale.value	20
*MouseDelayScale.decimalPoints	1
*MouseDelayScale.maximum	40
*MouseDelayScale.minimum	1
*MouseDelayScale.value	3
*MouseMaxSpeedScale.decimalPoints	0
*MouseMaxSpeedScale.maximum	500
*MouseMaxSpeedScale.minimum	1
*MouseMaxSpeedScale.value	300
*RepeatKeysToggle.set	True
*ShowStatusToggle.set	False
*SlowKeysOnAcceptToggle.set	True

(continued on next page)

General User Features

2.5 Tools and Utilities

Table 2–5 (Cont.) Default AccessX Resource Settings

Resource	Value
*SlowKeysOnPressToggle.set	True
*SlowKeysToggle.set	False
*SoundOnOffToggle.set	True
*StickyKeysToggle.set	False
*StickyModSoundToggle.set	True
*StickyTwoKeysToggle.set	True
*TimeOutScale.decimalPoints	0
*TimeOutScale.maximum	10
*TimeOutScale.minimum	1
*TimeOutScale.value	2
*TimeOutToggle.set	False
*ToggleKeysToggle.set	False

2.5.2 X Authority Utility (xauth)

V1.3

The X Authority utility (xauth) enables you to manage the contents of one or more X authority files. The X authority file contains information used to authorize client connections to the X server.

This utility is typically used to extract authorization records from one system and combine them with the records on another system, such as when granting access to additional users or enabling remote logins. The actual record entries vary depending on the authentication scheme currently in use.

In contrast to other X Window System utilities that are available with DECwindows Motif, xauth is included as a part of OpenVMS Alpha and OpenVMS I64 operating systems. The xauth commands are case-insensitive and available directly from the DCL command line, xauth command line, or from a batch file.

The command format for xauth is as follows:

```
$ xauth [-f authfile] [-options...] [command]
```

Table 2–6 defines the available options.

Table 2–6 X Authority Utility Options

Options	Description
<i>-f authfile</i>	Specifies the name of the X authority file. Version numbers are not allowed. If a display device is specified on the command line, xauth will use the X authority file referenced by the display device. Otherwise, xauth will use the default X authority file used by client applications. This file is the X authority file referenced by the DECW\$DISPLAY display device, the DECW\$XAUTHORITY logical, or SYS\$LOGIN:DECW\$XAUTHORITY.DECW\$XAUTH.
<i>-q</i>	Specifies that xauth operate in quiet mode. Status messages are not displayed. This is the default setting if the output from xauth is not directed to a terminal.
<i>-v</i>	Specifies that xauth operate in verbose mode. Status message are printed. This is the default setting if the output from xauth is directed to a terminal.
<i>-i</i>	Specifies that xauth ignore file locks. Normally, xauth will refuse to read or edit any files that have been locked by another program (such as, by another instance of xauth) and not timed out.
<i>-b</i>	Specifies that xauth break file locks before proceeding. Use this option only to clean up stale locks.
<i>-n</i>	Specifies that xauth not interpret the host name when the list command is used. Instead, xauth displays the literal value as it appears in the X authority file.

Table 2–7 defines the available commands.

Table 2–7 X Authority Utility Commands

Commands	Description
add	Adds or replaces the specified entries.
extract	Extracts and writes the specified entries to a new X authority file.
exit	Saves and closes the file and exits the xauth utility. (Available from the xauth command line only.)
remove	Deletes the specified entries.
merge	Appends entries from another X authority file.
nextract	Extracts the specified entries in numerical format.
nmerge	Merges the specified entries presented in numerical format.
list	Displays a listing of entries in the X authority file.
nlist	Displays a listing of entries in numerical format.

(continued on next page)

General User Features

2.5 Tools and Utilities

Table 2–7 (Cont.) X Authority Utility Commands

Commands	Description
generate	Used to generate a new authorization key. Requires that DECwindows Motif be installed and the SECURITY extension be enabled on the X display server. The generate command creates a key that applies to all entries with a matching IPv4 or IPv6 host address. It is not limited to the IP address of the host used to connect to the display server.
help	Displays information about the parameters and options for this utility. Subtopic help is also available by typing a question mark (?) at the command prompt.
info	Provides a brief overview of the X authority file.
quit	Closes the xauth utility without applying any changes. (Available from the xauth command line only.)
source	Runs xauth commands from a command file.

2.5.2.1 The X Authority File

The **X authority file** is a binary data file that contains information used to authorize connections to the X display server on a system running DECwindows Motif Version 1.3 or higher.

Each time a client application attempts to connect to an X server system that uses an authorization protocol, it references the current X authority file to determine the appropriate **authorization key** to apply in order to authenticate the connection. Each authorization key consists of the protocol name and token, which can be one of the following depending on the protocol in use:

- MIT-MAGIC-COOKIE-1 + random numeric code
- MIT-KERBEROS-5 + encrypted string (cached separately)

By default, an X authority file is created automatically the first time a user logs into a desktop on a system configured for MIT-MAGIC-COOKIE-1 or MIT-KERBEROS-5 authentication. The file is stored in that user's OpenVMS login directory (SYS\$LOGIN:DECW\$XAUTHORITY.DECW\$XAUTH). Each time the user subsequently logs into a desktop on that system, a new authorization key is generated, passed to the X server, and written to the user's X authority file. This key controls access to the X server during the DECwindows Motif session.

A separate X authority file can be manually defined on a server level (using the DECW\$SERVER_XAUTHORITY symbol) for those client applications that require access to the X server outside of the normal DECwindows Motif login process.

If the SECURITY extension is enabled, authorization keys can also be manually generated. Manually-generated keys can be used to further restrict server access. The generated key is stored in the X authority file on the client system overwriting any value already present for the specified display server. The key can be distributed to different client systems to allow connections to a specific server and can be revoked to stop subsequent connections.

Generated keys are assigned an **authorization ID** that associates the key with the user who generated the key. As a result, only the user who generated the key can revoke the key.

2.5.2.1.1 Format of an X Authority File Entry Each entry in an X authority file corresponds to a particular X display server and is composed of three main components:

display-name protocol token

display-name

Identifies the name of the X display to which you are authorizing access. The display name follows the supported display name format described in *HP DECwindows Motif for OpenVMS Management Guide*:

[transport/]host[:]server[.screen]

This format enables you to use a single X authority file to grant varying levels of access to different X display servers and connection families.

For example, the following entries grant access to the local display server on node HUBBUB and the remote display server on node ZEPHYR via the DECnet transport:

```
local/HUBBUB:0 MIT-MAGIC-COOKIE-1 cfc5ef98f9718f90154f355c0ae9f62
decnet/ZEPHYR::0 MIT-MAGIC-COOKIE-1 cfc5ef98f9718f90154f355c0ae9f62
```

- **[transport/]**
Identifies the network transport used to connect to an X display server. See *HP DECwindows Motif for OpenVMS Management Guide* for a list of the supported transport values. If a transport value is not specified, the default value is interpreted from the format of the remaining portions of the display-name entry, for example:
 - Host address and one colon (116.94.24.187:0) (TCP/IP)
 - Two colons (::0 or ZEPHYR::0) (DECnet)
 - No host name or address and one colon (:0) (local)
- **host[:]**
Identifies the name of the host system where the X display server is located. A value of 0 is interpreted as the local host, which is the default. The type of host is determined by the transport value. See *HP DECwindows Motif for OpenVMS Management Guide* for a full list of valid host name and address formats.
- **:server**
Identifies the server. This value is required and must be preceded by a single colon (:). Typically the value for a single-server system is :0. If you are specifying a display on a multi-server system (such as when using a proxy server), additional values may apply depending on the number of servers in the configuration. If you have specified a display device (with the SET DISPLAY command), the server portion of the entry is assumed from the device specification.
- **[.screen]**
Identifies the screen. On OpenVMS Alpha and OpenVMS I64 systems, the screen value is not held in the X authority file and is ignored when included in a command. All screens on a single server have the same authorization.

protocol

Indicates the authentication protocol in use. Valid values are MIT-MAGIC-COOKIE-1 and MIT-KERBEROS-5.

General User Features

2.5 Tools and Utilities

token

A random alphanumeric string that functions as a password authorizing a server connection. The format of the token depends on the authorization scheme in use. MIT-MAGIC-COOKIE-1 uses a 128-bit string known as a magic cookie. MIT-KERBEROS-5 uses an encrypted string to authorize server connections. This string is stored separately. The token entry in the X authority file represents the encoded location of the Kerberos keytab file and associated principal name, which is referenced by the server to locate the encrypted string.

2.5.2.1.2 Specifying an X Authority File By default, the X authority file referenced by client applications and the xauth utility is defined as SYS\$LOGIN:DECW\$XAUTHORITY.DECW\$XAUTH. You can override this default and specify an alternate X authority file in either of the following ways:

- You can create alternate X authority files and switch between them using the DECW\$XAUTHORITY logical. For example, the following command changes the X authority file in use for the current DECwindows Motif session to UNTRUSTED.DECW\$XAUTH:

```
$ DEFINE DECW$XAUTHORITY SYS$MANAGER:[SYSMGR]UNTRUSTED.DECW$XAUTH
```

The logical definition remains in use until it is redefined or an alternate value is specified using the SET DISPLAY/XAUTHORITY command.

- If a display device is used to create a client connection to an X server, you can specify an alternate X authority file using the SET DISPLAY/CREATE/XAUTHORITY command. Note that the file specified on this command line overrides both the default and any file referenced by the DECW\$XAUTHORITY logical.

2.5.2.2 Invoking xauth and Entering Commands

You can choose to enter commands interactively from DCL, or enter the utility and issue commands from the xauth command line.

Note that SYS\$LOGIN:DECW\$XAUTHORITY.DECW\$XAUTH is the default X authority file. If you want to work with an alternate file, use the -f option on the command line to specify the filename, as follows:

```
$ XAUTH -f SYS$SYSROOT:[SYSMGR]UNTRUSTED.DECW$XAUTH
Using authority file SYS$SYSROOT:[SYSMGR]UNTRUSTED.DECW$XAUTH
xauth>
```

Tips and Shortcuts

- If you are working with an X authority file other than the default, and plan to enter a series of commands, use the XAUTH -f command to enter the utility; then issue the subsequent commands from the utility command line. Otherwise, you will need to reenter the fully-qualified filename with each xauth command issued from the DCL command line.
- When adding a file entry, you can specify a period (.) in place of the value MIT-MAGIC-COOKIE-1. The period is replaced by the name of the authentication protocol.

2.5.2.3 IPv6 Considerations

V1.5

Note that xauth interprets certain transport values slightly different than in most other DECwindows Motif interfaces. Table 2–8 describes how xauth interprets each transport.

Table 2–8 xauth Transport Actions

With this Transport Value...	xauth Commands Affect...
INET	All entries in the X authority file whose host address matches any of the IPv4 addresses associated with the same TCP/IP host.
INET6	All entries in the X authority file whose host address matches any of the IPv4 or IPv6 addresses associated the same TCP/IP host.
TCP/IP or TCP	All entries in the X authority file as if INET or INET6 had been specified, depending on the setting of DECW\$IPV6_SUPPORT.

2.5.2.4 Accessing Online Help

To display a brief list of the available xauth commands or a summary of their function, issue either the XAUTH ? or XAUTH HELP command.

2.5.2.5 Creating an X Authority File

Use the XAUTH -f ADD command to manually create an X authority file. You must manually create an X authority file for the server when enabling authentication outside of a DECwindows Motif session. You can also choose to create additional user X authority files to store alternate authentication settings, such as for authorizing untrusted network connections.

An X authority file name can consist of any characters currently supported by OpenVMS; however, the file extension is restricted to a maximum of 37 characters and version numbers are not allowed.

The -f option specifies the name of the X authority file, and the ADD command creates the file by adding an entry. If you do not enter a fully-qualified filename, the new X authority file is written to the current directory by default.

For example, the following command creates a new X authority file UNTRUSTED.DECW\$XAUTH to be used to authorize untrusted network connections:

```
$ XAUTH -f UNTRUSTED.DECW$XAUTH ADD :0 .
cfcc5ef98f9718f90154f355c0ae9f62
```

2.5.2.6 Displaying File Information

To assist with debugging file access and write issues, xauth includes a command that displays summary information about a particular X authority file. Use the XAUTH INFO command to display information about an X authority file, such as the current lock status and change history.

General User Features

2.5 Tools and Utilities

For example, the following command displays summary information about the X authority file UNTRUSTED.DECW\$XAUTH:

```
$ XAUTH -f SYS$SYSROOT:[SYSMGR]UNTRUSTED.DECW$XAUTH INFO
Authority file:      SYS$SYSROOT:[SYSMGR]UNTRUSTED.DECW$XAUTH
File new:           no
File locked:        yes
Number of entries:   2
Changes honored:     yes
Changes made:        no
Current input:       command line
```

2.5.2.7 Viewing and Editing File Entries

Each X authority file assumes the default protections of the account and directory in which it resides. If you have the appropriate privileges, you can view or edit the contents of an X authority file. To ensure the appropriate level of security, access to this file is typically limited to either the local SYSTEM account, the file owner, or both.

Note

When an X authority file is open for viewing or editing, one or more lock files are created by adding -L or -C to the file extension (such as, *.DECW\$XAUTH-C). This renders the X authority file locked from further use. When the file is closed, the lock is subsequently removed, and the lock files deleted.

If a DECwindows Motif session is terminated abruptly, one or more locked files can remain. Use the XAUTH command with options -b or -i to either break or ignore the locks and gain access to the file.

Displaying File Entries

Use the XAUTH LIST command to display the contents of an X authority file.

For example, the following XAUTH command displays the entries in the X authority file UNTRUSTED.DECW\$XAUTH:

```
$ XAUTH -f UNTRUSTED.DECW$XAUTH LIST
local/ZEPHYR:0 MIT-MAGIC-COOKIE-1 cfcc5ef98f9718f90154f355c0ae9f62
decnet/ZEPHYR::0 MIT-MAGIC-COOKIE-1 cfcc5ef98f9718f90154f355c0ae9f62
116.94.24.187:0 MIT-MAGIC-COOKIE-1 cfcc5ef98f9718f90154f355c0ae9f62
```

Note

TCP/IP is considered the default transport for X authority file entries. As a result, the transport portion of the display name is assumed and not displayed for entries that use the TCP/IP transport.

To limit the list to entries related to a particular display, enter the display name at the end of the XAUTH LIST command, as follows:

```
$ XAUTH -f UNTRUSTED.DECW$XAUTH LIST ZEPHYR::0
decnet/ZEPHYR::0 MIT-MAGIC-COOKIE-1 cfcc5ef98f9718f90154f355c0ae9f62
```

With the TCPIP transport, multiple addresses may correspond to the same display name. When displaying X authority file entries, it may be difficult to distinguish which entry applies to which address. To differentiate the entries, use the `-n` qualifier to list them in numeric format, as follows:

```
$ XAUTH LIST
test13_2:0 MIT-MAGIC-COOKIE-1 12
test13_2:0 MIT-MAGIC-COOKIE-1 23
$ XAUTH -n LIST
#0006#fe8000000000000000000000200f8fffe101905#:0 MIT-MAGIC-COOKIE-1 12
#0000#c0a70209#:0 MIT-MAGIC-COOKIE-1 23
```

Adding and Removing File Entries

Use the `XAUTH ADD` and `XAUTH REMOVE` commands to add entries to or delete entries from an X authority file.

If you have created a display device (using the `SET DISPLAY` command), you can specify the device name on the `xauth` command line to insert or remove entries related to the display device. Typically, the X authority file entry for a display device corresponds to the display server specified by the `SET DISPLAY` command. However, if the `SET DISPLAY` command specifies that a proxy server be used, the file entry pertains to that proxy server.

For example, the following X authority file has a single entry for the LOCAL transport on node ZEPHYR. To use the same authorization key for the DECnet transport and to specify that Kerberos be used when connecting to remote node HUBBUB, you could add the following entries to the X authority file `UNTRUSTED.DECW$XAUTH`:

```
$ XAUTH -f UNTRUSTED.DECW$XAUTH
Using authority file untrusted.decw$xauth

xauth> LIST
local/ZEPHYR:0 MIT-MAGIC-COOKIE-1 cfcc5ef98f9718f90154f355c0ae9f62

xauth> ADD ::0 . cfcc5ef98f9718f90154f355c0ae9f62
xauth> ADD HUBBUB::0 MIT-KERBEROS-5 ""
```

General User Features

2.5 Tools and Utilities

```
xauth> LIST
local/ZEPHYR:0 MIT-MAGIC-COOKIE-1 cfc5ef98f9718f90154f355c0ae9f62
decnet/ZEPHYR::0 MIT-MAGIC-COOKIE-1 cfc5ef98f9718f90154f355c0ae9f62
decnet/HUBBUB::0 MIT-KERBEROS-5
```

```
xauth> EXIT
Writing X authority file untrusted.decw$xauth
```

Client applications running on systems in the same cluster share a single X authority file. As a result, in cluster configurations, adding an entry for the DECnet transport to the local system grants client applications running on other nodes in the cluster access to that system.

To discontinue remote access to HUBBUB, you could use the XAUTH REMOVE command to remove the entry, as follows:

```
$ XAUTH -f UNTRUSTED.DECW$XAUTH
Using authority file untrusted.decw$xauth

xauth> REMOVE HUBBUB::0
1 entries removed

xauth> LIST
local/ZEPHYR:0 MIT-MAGIC-COOKIE-1 cfc5ef98f9718f90154f355c0ae9f62
decnet/ZEPHYR::0 MIT-MAGIC-COOKIE-1 cfc5ef98f9718f90154f355c0ae9f62

xauth> EXIT
Writing X authority file untrusted.decw$xauth
```

Copying Entries Between Files

Use one or more of the following XAUTH commands to copy entries for a particular display from one X authority file to another.

This enables you to use an existing entry to grant another user access to a particular display or to obtain access to a remote host from the current display device.

- **EXTRACT** – Creates a new X authority file whose entries match those in the original file.
- **MERGE** – Appends the contents of one file to another, replacing entries for the same display name or adding entries for different names.
- **NEXTRACT** and **NMERGE** – These commands are designed to be used with the PIPE command. NEXTRACT extracts file entries in a text format that can then be used as input for the NMERGE command.

For example, the following command extracts the X authority file entry for the local transport from the file UNTRUSTED.DECW\$XAUTH and adds it to a new X authority file NEW_XAUTHORITY.DECW\$XAUTH:

```
$ PIPE XAUTH -f UNTRUSTED.DECW$XAUTH NEXTRACT SYS$OUTPUT :0 | -
_ $ XAUTH -f NEW_XAUTHORITY.DECW$XAUTH NMERGE SYS$INPUT
```

These commands can also be used with the rsh command to copy entries from an X authority file on an OpenVMS host to an X authority file on a remote UNIX system. For example, the following command extracts the entry for TCP/IP access (TCPIP/0:0) and adds it to the current file for user SMITH on the remote UNIX system FLOPSY:

```
$ PIPE XAUTH -f UNTRUSTED.DECW$XAUTH NEXTRACT - TCPIP/0:0 | -
_ $ rsh/user=smith/password=secret flopsy "xauth nmerge -"
```

Note

When using the PIPE and XAUTH commands to pass information to a UNIX host, you must press Ctrl/C to terminate the connection to the UNIX host and return control to OpenVMS.

2.5.2.8 Generating Authorization Keys

When the SECURITY extension is enabled on an X display server, you can generate additional authorization keys. Generated keys enable you to further manage server access and control the type of operations performed over the connection. For example, you can revoke a generated key at will, set it to expire after a certain time period, or use it to further grant or restrict the operations (at the X atom level) that can be performed over the connection.

Use the XAUTH GENERATE command to produce a new authorization key. Note that the generated key overwrites any existing key for the current session. To preserve the existing key, specify an alternate X authority file on the XAUTH command line.

For example, the following commands specify the alternate X authority file UNTRUSTED.DECW\$XAUTH, generate and display a new key for the local display, and write the generated key to the alternate file:

```
$ XAUTH -f UNTRUSTED.DECW$XAUTH
Using authority file untrusted.decw$xauth

xauth> LIST :0
local/ZEPHYR:0 MIT-MAGIC-COOKIE-1 cfcc5ef98f9718f90154f355c0ae9f62

xauth> GENERATE :0

xauth> LIST :0
local/ZEPHYR:0 MIT-MAGIC-COOKIE-1 cfcc4ff77f3709c46222c355f0ea1c93

xauth> EXIT
Writing X authority file untrusted.decw$xauth
```

2.5.3 X Keyboard Compiler Utility (xkbcomp)

V1.3

The Keyboard Compiler utility (xkbcomp) compiles X Keyboard source files into loadable X Keyboard layout (.XKM) files. Using xkbcomp, you can customize the standard layouts provided with the X Window System by creating or modifying the component source files.

For information about enabling and using XKB and X Keyboard keymaps, see the *HP DECwindows Motif for OpenVMS Management Guide*.

To run this utility, define xkbcomp as a foreign command:

```
$ xkbcomp == "$SYS$SYSTEM:DECW$XKBCOMP"
```

General User Features

2.5 Tools and Utilities

The command format for `xkbcomp` is as follows:

```
$ xkbcomp [-options...] input-file [output-file]
```

Table 2–9 Keyboard Compiler Options

Option	Description
-a	Specifies that all user actions be displayed.
-C	Creates a C header file during compilation.
-em1 <i>message</i>	Prints the specified message before printing any informational or error messages.
-emp <i>message</i>	Prints the specified message at the start of each line of messages.
-eml <i>message</i>	Prints the specified message after any informational or error messages.
-dflts	Specifies that the compiler generate default values for any missing parameters.
-I [<i>directory</i>]	Specifies the top level directory for include statements. A comma-separated list of multiple directories is allowed.
-l [<i>flags</i>]	Specifies that a list of matching keymap files be displayed, where <i>flags</i> can be one or more of the following options: f: lists fully-specified filenames h: lists hidden keymap files l: generates a long list p: lists partial keymap files r: lists recursive subdirectories The default of -l with no flags turns all options off.
-map <i>keymap file</i>	Specifies the keymap file to compile.
-merge	Merges the keymap file with the keymap currently residing on the server.
-o <i>file</i>	Specifies the fully-qualified name of the compiled keymap (.XKM) file.
-optional <i>parts</i>	Specifies optional components of a keymap file, where <i>parts</i> can be any combination of: c: compatibility map g: geometry k: keycodes s: symbols t: types Note that errors in specifying optional components are not fatal.
-R [<i>DIRECTORY</i>]	Specifies the directory in which the component source files are located.
-synch	Forces keymap synchronization.
-w [<i>level</i>]	Sets the warning level for compiler errors, ranging from 0 (none) to all (10).
-xkb	Generates an X keyboard (.XKB) source file.
-xkm	Generates a compiled keymap (.XKM) file.

For information about the keyboard components database, see the *HP DECwindows Motif for OpenVMS Management Guide*.

2.5.4 Window Dump to Print File (xpr) Utility

V1.2

The Window Dump to Print File utility prints an X Window dump using the xpr program.

The xpr program receives as input a window dump file produced by the Window Dump utility (xwd) and formats it for output on the following printers:

- PostScript
- DIGITAL LN03 or LA100
- IBM PP3812 page printer
- HP LaserJet (or other PCL printers)
- HP PaintJet

To use the xpr program, define xpr as a user-defined command:

```
$ xpr == "$DECW$UTILS:XPR"
```

You must specify an input file. The xpr program prints the largest possible representation of the window on the output page. Options allow the user to add headers and trailers, specify margins, adjust the scale and orientation, and append multiple window dumps to a single output file.

Use the following command format:

```
$ xpr input_file [options...]
```

Options include:

```
-append filename -noff -output filename
-compact
-device {ln03 | la100 | ps | lw | pp | ljet | pjet | pjetxl}
-dump
-gamma correction
-gray {2 | 3 | 4}
-height inches -width inches
-header string -trailer string
-landscape -portrait
-left inches -top inches
-noposition
-nosixopt
-plane n
-psfig
-render type
-report
-rv
-scale scale
-slide
-split n-pages
```

Table 2–10 defines the available options.

General User Features

2.5 Tools and Utilities

Table 2–10 Window Dump to Print File Options

Option	Description
-device <i>devtype</i>	Specifies the device on which the file is printed. Currently supported devices: la100 DIGITAL LA100. ln03 DIGITAL LN03. ljet HP LaserJet series and other monochrome PCL devices such as ThinkJet, QuietJet, RuggedWriter, HP series, and HP-series printers. pjet HP PaintJet (color mode). pjetxl HP PaintJet XL Color Graphics Printer (color mode). pp IBM PP3812. ps PostScript printer. lw LaserWriter is equivalent to -device ps and is provided only for backwards compatibility. The default is PostScript.
-scale <i>scale</i>	Affects the size of the window on the page. The PostScript, LN03, and HP printers can translate each bit in a window pixel map into a grid of a specified size. For example, each bit might translate into a 3x3 grid. This would be specified by -scale 3. By default, a window is printed with the largest scale that will fit onto the page for the specified orientation.
-height <i>inches</i>	Specifies the maximum height of the page.
-width <i>inches</i>	Specifies the maximum width of the page.
-left <i>inches</i>	Specifies the left margin in inches. Fractions are allowed. By default the window is centered in the page.
-top <i>inches</i>	Specifies the top margin for the picture in inches. Fractions are allowed.
-header <i>string</i>	Specifies a header string to be printed above the window.
-trailer <i>string</i>	Specifies a trailer string to be printed below the window.
-landscape	Forces the window to be printed in landscape mode. By default, a window is printed so that its longest side follows the long side of the paper.
-portrait	Forces the window to be printed in portrait mode. By default a window is printed so that its longest side follows the long side of the paper.
-plane <i>number</i>	Specifies which bit plane to use in an image. The default is to use the entire image and map values into black and white based on color intensities.
-gray	Uses a 2x2, 3x3, or 4x4 gray scale conversion on a color image, rather than mapping to strictly black and white. This doubles, triples, or quadruples the effective width and height of the image.
-rv	Forces the window to print in reverse video.
-compact	Uses run-length encoding for compact representation of windows with white pixels.

(continued on next page)

Table 2–10 (Cont.) Window Dump to Print File Options

Option	Description
-output <i>filename</i>	Specifies an output file name.
-append <i>filename</i>	Specifies a file name previously produced by xpr to which the window is to be appended.
-noff	When specified in conjunction with -append, the window appears on the same page as the previous window.
-split <i>n-pages</i>	Allows the user to split a window onto several pages. This might be necessary for very large windows that would otherwise cause the printer to overload and print the page in an obscure manner.
-psfig	Suppresses translation of the PostScript picture to the center of the page.
-density <i>dpi</i>	Indicates dot-per-inch density to be used by the HP printer.
-cutoff <i>level</i>	Changes the intensity level where colors are mapped to either black or white for monochrome output on a LaserJet printer. The level is expressed as percentage of full brightness. Fractions are allowed.
-noposition	Causes header, trailer, and image positioning command generation to be bypassed for LaserJet, PaintJet and PaintJet XL printers.
-gamma <i>correction</i>	Changes the intensity of the colors printed by the PaintJet XL printer. The correction is a floating-point value in the range 0.00 to 3.00. Consult the operator's manual to determine the correct value for the specific printer.
-render <i>algorithm</i>	Allows the PaintJet XL printer to render the image with the best quality versus performance tradeoff. Consult the operator's manual to determine the available algorithms.
-slide <i>filename</i>	Allows overhead transparencies to be printed using the PaintJet and PaintJet XL printers.

The program contains the following limitations:

- Support for PostScript output currently cannot use the -append, -noff, or -split options.
- The -compact option is only supported for PostScript output. It compresses white space but not black space, so it is not useful for reverse-video windows.
- For color images, map directly to PostScript image support.

Program limitations with an LN03 printer:

- The current version of xpr can print most X Windows that are not larger than two-thirds of the screen.

For example, the LN03 prints a large Emacs window, but fails when trying to print the entire screen.

General User Features

2.5 Tools and Utilities

- The LN03 has memory limitations that cause it to incorrectly print large or complex windows. The two most common errors encountered are “band too complex” and “page memory exceeded” and are described as follows:
 - “band too complex”

A window may have a particular six pixel row that contains too many changes (from black to white to black). This causes the printer to drop part of the line and possibly drop parts of the page. The printer flashes the number “1” on its front panel when this problem occurs. A possible solution to this problem is to increase the scale of the picture or to split the picture onto two or more pages.
 - “page memory exceeded”

This occurs if the picture contains too much black space, or if the picture contains complex half-tones, such as the background color of a display. When this problem occurs, the printer automatically splits the picture onto two or more pages. The number “5” may flash on its front panel. As a possible solution to the problem, it might be necessary to either cut and paste or to rework the application to produce a less complex picture.

Program limitations with a LA100 printer:

- The picture is always printed in portrait mode.
- The scale is ignored.
- The scale factor will be different in the horizontal and vertical directions.

Program limitations with an HP printer:

- If the `-density` option is not specified, 300 dots-per-inch (dpi) is assumed for the `ljet` device and 90-dpi for the `pjet` device. The LaserJet printer supports 300-, 150-, 100-, and 75-dpi. Consult the operator’s manual to determine the densities supported by other printers.
- If the `-scale` option is not specified, the image is expanded to fit the printable page area.
- The default printable page area is 8x10.5 inches. Other paper sizes can be accommodated using the `-height` and `-width` options.
- Note that a 1024x768 image fits the default printable area when processed at 100-dpi with `scale=1`; the same image can also be printed using 300-dpi with `scale=3`, but it requires more data to be transferred to the printer.
- The `xpr` program may be tailored for use with monochrome PCL printers other than the LaserJet. To print on a ThinkJet (HP 2225A) printer, invoke `xpr` as follows:

```
xpr -density 96 -width 6.667 filename
```

To print black-and-white output on a PaintJet printer, invoke `xpr` as follows:

```
xpr -density 180 filename
```

- The monochrome intensity of a pixel is computed as $0.30 \cdot R + 0.59 \cdot G + 0.11 \cdot B$. If the computed intensity of a pixel is less than the `-cutoff level`, it prints white. This maps light-on-dark display images to black-on-white hard copy. The default cutoff intensity is 50% of full brightness. For example, specifying `-cutoff 87.5` means that a pixel will be displayed as black if the computed intensity is less than 85% of full brightness.

- A LaserJet printer must be configured with sufficient memory to print the image. To print a full page at 300-dpi, approximately 2 MB of printer memory is required.
- Color images are produced on the PaintJet printer at 90-dpi. The PaintJet is limited to 16 colors from its 330 color palette on each horizontal print line. The xpr program issues a warning message if more than 16 colors are encountered on a line. Xpr programs the PaintJet for the first 16 colors encountered on each line and uses the nearest matching programmed value for other colors on the line.
- Specifying the -rv option on the PaintJet printer causes black and white to be interchanged on the output image. No other colors are changed.
- Multiplane images must be recorded by xwd in ZPixmap format. Single-plane (monochrome) images may be in either XYPixmap or ZPixmap format.
- Some PCL printers do not recognize image positioning commands. Output for these printers is not centered on the page, and header and trailer strings may not appear where expected.
- The -gamma and -render options are supported only on the PaintJet XL printers.
- The -slide option is not supported on LaserJet printers.
- The -split option is not supported on HP printers.
- The -gray option is not supported on HP or IBM printers.

System Management Features

This chapter provides information about new features and enhancements related to DECwindows Motif system management.

3.1 Installation and Upgrade Information

The following sections describe features that pertain to installing and upgrading DECwindows Motif systems.

3.1.1 DECW\$STARTUP Log File

V1.5

When you start DECwindows Motif using the command `@SYS$MANAGER:DECW$STARTUP RESTART`, a startup log file is now created. This log file records activity from the detached process that invokes DECwindows Motif, which can be useful in diagnosing startup problems.

The name of the log file is `SYS$MANAGER:DECW$STARTUP_n.LOG`, where *n* represents the server number. The default server number value is 0; hence, the default filename is `DECW$STARTUP_0.LOG`. If you have set the server number to a value other than 0, that value is then used in the name of the corresponding log file.

To minimize disk space usage, the log files are purged periodically. A maximum of three versions of a startup log file are kept on disk.

For more information on the DECwindows Motif startup procedure, see the *HP DECwindows Motif for OpenVMS Management Guide*.

3.2 Security and Authorization

The following sections describe features that pertain to maintaining system and network security of DECwindows Motif systems.

3.2.1 Support for ACME Subsystem (Alpha Only)

V1.5

The Authentication and Credentials Management Extensions (ACME) subsystem features of OpenVMS Alpha are now supported by DECwindows. The following dialog boxes and screens support access checks using ACME:

- Login dialog box (New Desktop)
- Lock Screen dialog box (New Desktop)
- Start Session dialog box (Traditional DECwindows Desktop)
- Pause Session screen (Traditional DECwindows Desktop)

System Management Features

3.2 Security and Authorization

All ACME features are supported, including intrusion detection and remote authentication. See the *HP OpenVMS Guide to System Security* for a detailed description of the ACME subsystem and its components.

3.3 X Display Server Management

The following sections describe features that pertain to managing the DECwindows X11 Display Server.

3.3.1 New Parameter for Enabling IPv6 Support in the Font Server

V1.5

A new server parameter, `DECW$IPV6_FONT_SUPPORT`, has been added that controls how the TCP transport is interpreted when specifying font servers either through use of the `DECW$FONT_SERVER` parameter or through the use of an explicit font server path. In addition, the `DECW$FONT_SERVER` parameter now accepts an expanded set of transport values.

To set these parameters for the server process, do the following:

1. Edit the `SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM` file.
2. Define both the `DECW$FONT_SERVER` and `DECW$IPV6_FONT_SUPPORT` parameters as required.
3. Save the file and restart the server.

DECW\$FONT_SERVER

The transport options available through this parameter have been expanded to include all valid transports, as described in the *HP DECwindows Motif for OpenVMS Management Guide*.

DECW\$IPV6_FONT_SUPPORT

This parameter controls the interpretation of the TCP or TCPIP transport name in the `DECW$FONT_SERVER` definition and whenever a client explicitly specifies a font server path. Set the parameter to one of the following values:

- `TCP_IS_INET6`

The TCP and TCPIP transport names are interpreted as synonyms for the INET6 transport. Using this value enables IPv6 as the default transport between the display server and its font servers when the TCP or TCPIP transport name is specified. Note that this setting requests the IPv6 transport be used as a default. If the IPv6 transport is not available for the specified node, the IPv4 transport is used.

- `DISABLED`

The TCP and TCPIP transport names are interpreted as synonyms for the INET transport. This is the default value if the parameter is undefined. Using this value sets IPv4 as the transport between the display server and its font server when a TCP or TCPIP transport name is specified. A client can still request the IPv6 transport by explicitly specifying the INET6 transport in its font server path.

Programming Features

This chapter describes new features relating to application and system programming in the DECwindows Motif environment. This includes extensions, libraries, and functions made available as part of the X11R6.6 and X11R6.7 implementation as well as those that are specific to the DECwindows Motif environment.

For information on how to program X Window System applications, see the *X Window System* and *X Window System Toolkit* (Scheifler and Gettys) series of manuals described in the *HP DECwindows Motif for OpenVMS Documentation Overview*.

4.1 General Run-Time and Programming Environment

The following sections describe features related to general DECwindows Motif programming environment.

4.1.1 Multithreading Support

V1.3

DECwindows Motif now supports multithreaded client applications. Client applications that use the HP POSIX Threads Library or HP Ada tasks are now fully supported in the DECwindows Motif Version 1.3 or higher environment.

Additionally, each of the following libraries are now fully thread-safe, supporting simultaneous calls from multiple threads:

- X Library (Xlib)
- X Toolkit Intrinsics (Xt)
- X Window System Extensions library
- Inter-Client Exchange (ICE) library
- X Session Management Protocol (XSMP) library

4.1.2 Binary Compatibility

V1.3

Client applications linked against previous versions of DECwindows Motif are binary compatible, with the exception of those applications that use any changed or retired Xlib entry points documented in this manual and the *HP DECwindows Motif for OpenVMS Release Notes*.

Programming Features

4.1 General Run-Time and Programming Environment

Existing applications require recompilation against the X11R6.6-compatible X Window System libraries if they produce a shared image potentially used by other multithreaded applications and that image uses:

- Any of the following macros defined in DECW\$INCLUDE:XLIBINT.H and is intended for use in a multithreaded environment:

LockDisplay
UnlockDisplay
LockMutex
UnlockMutex

- The ConnectionNumber macro or XConnectionNumber function and assumes the return value is an event flag.

Note that recompiling unmodified applications (those that use the previous version of the CompositeClassExtensionRec structure) against the updated library functions may generate Xt warning messages.

4.1.2.1 Use of Asynchronous System Traps (ASTs)

In the past, DECwindows Motif supported application calls from user-mode and normal-mode AST handlers. With DECwindows Motif Version 1.3 and higher, applications calls from AST handlers are only supported for existing, unmodified applications (compiled against the X11R5 libraries).

New applications compiled against the X11R6.6 libraries should not call functions (other than XtNoticeSignal) from AST handlers even in single-threaded environments.

Applications compiled against the updated libraries should either use multiple threads if a higher level of concurrency is required or call XtNoticeSignal so that AST events are processed in the Xt main loop.

4.1.2.2 Levels of Thread Safety and Concurrency

New and existing DECwindows Motif shared images can be grouped according to the following levels of thread safety:

- **Thread-safe**—The image can be called concurrently from multiple threads.
- **Thread-aware**—The image can be used in an application that includes multiple threads. However, the application code must avoid making concurrent calls to the image, typically by using a global lock.
- **Thread-unsafe**—The image cannot be used in any application that has multiple thread support enabled.

Table 4–1 shows the current thread safety level for each of the DECwindows Motif shared images. Note that all images in the Translated Image Environment are thread-unsafe.

Table 4–1 Level of Thread Safety for DECwindows Motif Images

Image	Level of Thread Safety
DECW\$AILSHR.EXE	unsafe
DECW\$AILSHRR5.EXE	unsafe

(continued on next page)

Table 4–1 (Cont.) Level of Thread Safety for DECwindows Motif Images

Image	Level of Thread Safety
DECW\$BKRSRSHR.EXE	aware
DECW\$BKRSRSHR12.EXE	aware
DECW\$D2DXLIBSHR.EXE	aware
DECW\$DWTLIBSHR.EXE	aware
DECW\$DXMLIBSHR.EXE	aware
DECW\$DXMLIBSHR12.EXE	aware
DECW\$ICELIB.EXE	safe
DECW\$LCNLIBSHR.EXE	safe
DECW\$MAILSHR.EXE	unsafe
DECW\$MAILSHR12.EXE	unsafe
DECW\$MRMLIBSHR12.EXE	aware
DECW\$PRINTWGTSRSHR.EXE	aware
DECW\$SMSRSHR.EXE	safe
DECW\$TERMINALSHR.EXE	aware
DECW\$TERMINALSHR12.EXE	aware
DECW\$XEXTLIBSHR.EXE	safe
DECW\$XLIBSHR.EXE	safe
DECW\$XMLIBSHR.EXE	aware
DECW\$XMLIBSHR12.EXE	aware
DECW\$XMULIBSHR.EXE	aware
DECW\$XMULIBSHRR5.EXE	aware
DECW\$XTLIBSHRR5.EXE	safe
DECW\$XTRAPLIBSHR.EXE	aware
DECW\$XTRAPLIBSHRR5.EXE	aware
DECW\$XTSHR.EXE	aware

In addition to thread safety, the updated libraries offer varying levels of concurrency when called from multiple threads:

- **Xlib and X Window System Extension libraries**

For most operations, Xlib and X Window System Extension libraries allow a single concurrent operation on each display connection. If XOpenDisplay is called twice to open two separate server connections, both connections can be operated upon at the same time. However, there are a few operations for which a global lock is needed to prevent corruption of global data.

Note that XSelectAsyncInput and XSelectAsyncEvent are not supported if multithreading has been enabled by a call to XInitThreads. In general, use of XSelectAsyncEvent and XSelectAsyncInput is discouraged since they are non-standard functions. The equivalent functionality can be obtained by using threaded Xlib functions.

- **Xt library**

For most operations, Xt allows a single concurrent operation on each application context. There are a few operations for which a global lock is needed to prevent corruption of global data.

Programming Features

4.1 General Run-Time and Programming Environment

- **ICE and XSMP libraries**

ICE and XSMP operations allow a single concurrent operation on each ICE connection.

- **Transport library**

The transport library is used to communicate between the client application and the X server and between multiple client applications that use the ICE library. The final level of the transport code (which can communicate via global sections, DECnet, or TCP/IP) executes in OpenVMS inner mode, and as a result is serialized by the operating system.

Running on only one kernel thread at a time, this code temporarily blocks all other kernel threads making OpenVMS system calls that execute in inner mode. However, whenever the transport code is blocked, such as when waiting for a connection to open or for a reply to arrive, it is blocked at user mode allowing other threads to execute and use the transport.

4.1.2.3 Enabling Support for Multithreading

To enable multithreading, a client application must include initial calls to the multithreading functions in Table 4–2. The specific functions called by the application depend on the shared image(s) in use.

Table 4–2 Multithreading Functions

Function	In Image	Enables Multithreading For
DECW\$LCN_THREAD_INIT	DECW\$LCNLIBSHR	Transport Interface
IceInitThreads	DECW\$ICELIB	ICE and XSMP
XInitThreads	DECW\$XLIBSHR	Xlib and X Extensions
XtToolkitThreadInitialize	DECW\$XTLIBSHRR5	Xt

Note that IceInitThreads and XInitThreads implicitly call DECW\$LCN_THREAD_INIT. The interface to DECW\$LCN_THREAD_INIT is described in Section 4.8.2.

These functions have no arguments and return a success status upon successful initialization. To ensure successful initialization, be sure to:

- Link the image with threads.
- Verify there is adequate process memory.
- Issue the initialization call prior to making any other call.

In addition to calling one of the multithreading functions, a client application must also be linked against the POSIX Threads Library. For example:

```
$ LINK THREAD_ICO/THREADS SYS$INPUT/OPT
SYS$LIBRARY:DECW$XLIBSHR/SHARE
SYS$LIBRARY:PTHREAD$RTL/SHARE
```

Explicit links against the threads library are not required if the application calls POSIX thread functions (such as, pthread_create).

4.1.2.4 Developing Applications with Thread-Aware Images

If a thread-aware image is used in a multithreaded application, the image must not accept concurrent calls nor make calls to other images that could change the state of the thread-aware image.

Note that if a thread-aware application uses the `XtAppMainLoop` or `XtMainLoop` function for dispatch handling, calls to the image will be made from callback functions. Xt makes these callbacks with an exclusive lock held on the application context. To avoid conflicts and deadlocks, applications that use a thread-aware image should include calls to `XtAppLock` before and `XtAppUnlock` after each call, or sequence of calls, to the image. The application must also call `XtProcessLock` and `XtProcessUnlock` to protect the thread-aware image against changes made by Xt to process global data.

In the following example, an application contains a background thread that constantly checks for error situations and displays an error message when a problem occurs. The main program thread first initializes thread support, creates the application context, creates the background thread, and then enters the `Xtmain` loop:

```
static XtAppContext app_context;

int main ()
{
    .
    .
    .
    MrmInitialize ();
    XInitThreads ();
    XtToolkitThreadInitialize();
    XtToolkitInitialize();
    app_context = XtCreateApplicationContext();
    .
    .
    .
    pthread_create (&thread, 0, backgroundCode, 0);
    .
    .
    .
    XtAppMainLoop(app_context)
}
```

Code for the background thread is as follows:

```
void* backgroundCode (void* data)
{
    .
    .
    .
    if (problem_detected)
    {
        XtAppLock (app_context);
        XtProcessLock();
        if (! dlog ) dlog = XmcreateWarningDialog (...);
        XtManageChild(dlog);
        XSync(display, 0);
        XtProcessUnlock();
        XtAppUnlock(app_context);
    }
    .
    .
    .
}
```

Programming Features

4.1 General Run-Time and Programming Environment

Callbacks for handling the main events of the application do not require changes for multithreading, since they are called with the application context already locked.

Worker Threads

Each of the DECwindows Motif libraries can create worker threads to support multithreading. These threads are identified by their name, which begins with the string DECW\$.

Worker threads typically operate at an elevated priority to prevent task inversion, where a high-priority application thread is waiting for the worker thread to complete its operation. Note that worker threads are typically used for short duration tasks, such as responding to an internal AST or sending a status broadcast to all threads waiting for a particular activity.

Upcalls and Kernel Threads

In general, DECwindows Motif supports client applications either with or without upcalls or multiple kernel threads enabled. However, to avoid problems with priority inversion, HP recommends that upcalls be enabled for all applications that use XtAppAddInput. If upcalls cannot be enabled, then HP recommends assigning the same priority to all threads that use DECwindows Motif.

For example, an application calls XtAppAddInput to request a response to an OpenVMS event flag. The worker thread executes a SYS\$WFLOR system call to wait for the event flag. Without upcalls enabled, this thread remains available even though there is no event flag set. And as a result, lower priority threads would not be scheduled.

Cancellation Points

Although some calls in the thread-safe libraries include cancellation points, the action of canceling threads that are executing DECwindows Motif functions is not supported. Canceled threads may hold locks, which can block other threads.

Multiple Application Contexts

Note that multiple application contexts should not be used with multiple threads and thread-aware images. Thread-aware images may contain process global data that requires a single lock to control the data. However, multiple calls to thread-aware images may be made from Xt event handling functions prior to acquiring the lock.

4.2 Application Programming

The following sections describe features related to application programming.

4.2.1 CDA Programming

This section describes features and changes related to CDA programming.

4.2.1.1 Changes to the CDA Programming Interface

V1.2

This section describes the changes to the programming interface for this version of CDA Run-Time Services.

This version provides a new set of header files that define CDA constants, types, and routines using portable naming conventions. By using these new naming conventions, you can use a wider variety of C compilers to minimize the amount of system-specific code in your CDA applications.

The names of the new set of header files are the same as the names of the previous set of header files, except that the dollar sign (\$) has been removed. For example, the `cda$msg.h` include file is now called `cdamsg.h`. Other examples include the following: The `DDIF$K_DSC_MAJOR_VERSION` symbol is now declared as `DDIF_K_DSC_MAJOR_VERSION`, and the `CDA$_NORMAL` status value is now defined as `CDA_NORMAL`.

The previous set of header files is also included in this version, but these files will no longer be updated. Changes introduced since the release of DECwindows Motif Version 1.1 (for example, the new definitions for audio support), are available only in the new set of header files. To use the new CDA features, change the file names in your source code.

The new set of header files supplements the previous set of header files. If you want to write ANSI-compliant applications using CDA definitions and CDA Toolkit calls, use the new set of header files. However, you can continue to use the header files that define symbols containing the dollar sign (\$) provided you choose a non-ANSI compilation mode.

By using the previous set of header files, you can successfully build existing source code that uses the previous naming conventions.

See Table 4–3 for a list of new header file names.

Table 4–3 New Header File Names

Previous Name	New Name
<code>cda\$def.h</code>	<code>cdadef.h</code>
<code>cda\$msg.h</code>	<code>cdamsg.h</code>
<code>ddif\$def.h</code>	<code>ddifdef.h</code>
<code>dtif\$def.h</code>	<code>dtifdef.h</code>
<code>cda\$ptp.h</code>	<code>cdaptp.h</code>
<code>cda\$typ.h</code>	<code>cdatyp.h</code>
<code>dvr\$msg.h</code>	<code>dvrmsg.h</code>
<code>dvr\$cc_def.h</code>	<code>dvrccdef.h</code>
<code>dvr\$cc_ptp.h</code>	<code>dvrccptp.h</code>
<code>dvr\$decw_def.h</code>	<code>dvrwdef.h</code>
<code>dvr\$decw_ptp.h</code>	<code>dvrwptp.h</code>

4.2.1.2 Changes to CDA External Reference Processing

V1.2

CDA Run-Time Services supports relative file specifications for external references. Relative references are also supported: that is, a reference where the directory path is not fully specified but is relative to the directory path of the parent document.

Programming Features

4.2 Application Programming

4.2.1.3 Restructuring CDA Shareable Images

V1.2

The CDA Viewer includes two shareable images to allow installation on systems where DECwindows is not installed.

In DECwindows Motif for OpenVMS Version 1.2 software, shareable images that use the X services were renamed. Table 4–4 lists the shareable images in the various versions of the CDA Viewer.

Table 4–4 Names of Shareable Images

CDA Version	Image Name	Description
Version 1.6	CDA\$ACCESS	CDA Run-Time Services shareable image.
	DDIF\$VIEWSHR	Callable viewer widget.
Version 1.7	CDA\$ACCESS	CDA Run-Time Services shareable image.
	DDIF\$VIEWSHR	The DDIF\$DECW_VIEWSHR widget is a callable viewer widget that uses the LIB\$FIND_IMAGE_SYMBOL routine to invoke the DDIF\$DECW_VIEWSHR (DECwindows interface) and DDIF\$CC_VIEWSHR (character-cell interface) widgets.
	DDIF\$DECW_VIEWSHR	
	DDIF\$CC_VIEWSHR	
Version 1.8A or later	CDA\$ACCESS	CDA Run-Time Services shareable image.
	DDIF\$VIEWSHR12	The DDIF\$VIEWSHR12 widget is a callable viewer widget that uses LIB\$FIND_IMAGE_SYMBOL to invoke the DDIF\$DECW_VIEWSHR12 (DECwindows interface) and DDIF\$CC_VIEWSHR (character-cell interface) widgets.

By using the LIB\$FIND_IMAGE_SYMBOL routine to reference the entry points to the DDIF\$DECW_VIEWSHR, DDIF\$DECW_VIEWSHR12, and DDIF\$CC_VIEWSHR images, an application can dynamically determine whether it can execute in a given environment. The DDIF\$VIEW.EXE application now replaces this routine.

The previous DDIF\$VIEWSHR.EXE shareable image is still included to maintain compatibility with applications linked against it. However, new applications (and previous applications that take advantage of new features) should use the new shareable images.

4.2.2 DECterm Programming

This section describes features and changes related to DECterm programming.

4.2.2.1 ReGIS Input Cursors

V1.2–3

DECterm supports the following input cursors: cross-hair, rubber-band line, diamond, and rubber-band rectangle. To select input cursors use the S(C(In)) command. Table 4–5 shows the values of *n*.

Table 4–5 ReGIS Input Cursors—Cursor styles and Values

Cursor Style	Variable <i>n</i>
Cross-hair	Omitted
Cross-hair (default)	0
Diamond	1
Cross-hair	2
Rubber-band line	3
Rubber-band rectangle	4

Note

If a shape other than the diamond cursor is desired when *n* is equal to 1, define the logical name DECW\$DECTERM_REGIS_CURSOR to be one of the numbers defined in the SYS\$LIBRARY:DECW\$CURSOR.H file.

4.2.2.2 Page-Movement Escape Sequences

V1.2

The following page-movement escape sequences are implemented in DECterm:

NP	<CSI> Pn U	Next Page
PP	<CSI> Pn V	Previous Page
PPA	<CSI> Pn P	Page Position Absolute
PPB	<CSI> Pn R	Page Position Backward
PPR	<CSI> Pn Q	Page Position Relative

Note that “Pn” is the number of pages to move; the exception is PPA, where “Pn” is the actual page number.

Note

DECterm does not support cursor coupling; the cursor is always bound to the current (displayed) page.

4.3 OSF/Motif Toolkit (Xm) Programming

The following sections contain features related to OSF/Motif Toolkit (Xm) programming.

4.3.1 Drag-and-Drop Enabled Widgets

V1.2

The drag-and-drop feature lets you move or copy information between widgets. This feature is provided primarily for programmers to incorporate the feature into their applications.

All DECwindows Motif for OpenVMS Version 1.2 and higher applications support the drag-and-drop feature, with the exception of Notepad. DECwindows Mail supports drag-and-drop in all windows except the main message area, where DECwindows Mail has its own drag-and-drop feature; you can use MB2 to move messages around with the SVN interface.

Programming Features

4.3 OSF/Motif Toolkit (Xm) Programming

Drag-and-drop functionality has been implemented in the widgets listed in Table 4–6.

Table 4–6 Drag-and-Drop Widgets

Widget	Drag Operation	Drop Operation
XmText	copy and move	copy and move
XmTextField	copy and move	copy and move
XmLabel	copy	
XmPushButton	copy	
XmToggleButton	copy	
XmList	copy	

For information about how to include additional drag-and-drop functionality in applications and for an example of a drag-and-drop program, see the *Open Software Foundation: OSF/Motif Programmer's Guide, Revision 1.2* manual.

4.4 HP Extensions to Motif (DXm) Programming

The following sections describe features related to HP (Digital) Extensions to Motif (DXm) programming.

4.4.1 SVN Widget Supports Extended Selection

V1.2–6

The Structured Visual Navigation (SVN) widget now allows users to extend a range of selection using the Shift+Down-Arrow key sequence. Note that this change has also been applied to the sample program SVNMSAMPLE.C.

4.4.2 DXmCSText Input Method Support

V1.2

X11R5 input method support was added to the DXmCSText widget. Specify input methods using the vendor shell XmNinputMethod resource. However, to maintain backward compatibility, the existing input method resources DXmNinputMethod and DXmNinputMethodType are still available.

4.5 X Toolkit Intrinsics (Xt) Programming

The following sections describe features related to the X Toolkit Intrinsics (Xt).

4.5.1 New Functions Available with X11R6.6 Upgrade

V1.3

The following functions from X11R6.6 have been added to Xt.

Table 4–7 New Xt Functions Supported for X11R6.6

Function Name	Description
XtAppAddBlockHook	Registers a block hook procedure.
XtAppAddSignal	Registers a signal callback.
XtAppGetExitFlag	Supports controlled exit from main loop in a multithreaded application by returning the flag set by XtAppSetExitFlag.
XtAppLock	Locks the application context in a multithreaded application.
XtAppSetExitFlag	Supports controlled exit from main loop in a multithreaded application by setting a flag in the application context.
XtAppUnlock	Releases an application context lock.
XtCancelSelectionRequest	Cancels a multiple selection request.
XtChangeManagedSet	Simultaneously removes from and adds to the geometry managed set of a composite widget.
XtCreateSelectionRequest	Adds to a multiple selection request.
XtDispatchEventToWidget	Dispatches an event to a specified widget.
XtGetClassExtension	Locates a class extension record of an object class.
XtGetDisplays	Lists the open displays associated with an application context.
XtGetKeyboardFocusWidget	Determines which widget would be the end result of keyboard event forwarding for a keyboard event on a specified widget.
XtGetSelectionParameters	Gets target parameters needed to perform a selection conversion.
XtHooksOfDisplay	Retrieves the hook registration object for the specified display.
XtInsertEventTypeHandler	Registers an event handler procedure by event type.
XtIsSessionShell	Widget subclass verification function for the session shell widget.
XtLastEventProcessed	Retrieves the last event processed by XtDispatchEvent.
XtNoticeSignal	Notifies the X Toolkit that a signal has occurred.
XtOpenApplication	Convenience function to initialize intrinsics, create an application context, open a display connection, and create an application shell.
XtProcessLock	Locks the X Toolkit process lock.
XtProcessUnlock	Releases the X Toolkit process lock.

(continued on next page)

Programming Features

4.5 X Toolkit Intrinsic (Xt) Programming

Table 4–7 (Cont.) New Xt Functions Supported for X11R6.6

Function Name	Description
XtRegisterDrawable	Associates a drawable with a widget so that the drawable receives events as if part of the widget.
XtRegisterExtensionSelector	Registers a procedure to receive extension events for a widget.
XtRemoveBlockHook	Discontinues use of a block hook procedure.
XtRemoveEventTypeHandler	Removes a registration created by XtInsertEventTypeHandler.
XtRemoveSignal	Removes a registered signal callback.
XtReleasePropertyAtom	Releases a reservation made by XtReservePropertyAtom.
XtReservePropertyAtom	Reserves a unique atom for selection requests on a widget.
XtSendSelectionRequest	Sends a multiple selection request.
XtSessionGetToken	With the new session shell widget, gets an additional token for a save callback response with a deferred outcome.
XtSessionReturnToken	Returns a token obtained using XtSessionGetToken when checkpoint processing is complete.
XtSetEventDispatcher	Registers the event dispatcher procedure for events of the specified type.
XtSetSelectionParameters	Associates target parameters with a selection.
XtToolkitThreadInitialize	Initializes multithreaded support.
XtUnregisterDrawable	Removes an association set by XtRegisterDrawable.
XtVaOpenApplication	Convenience function to initialize intrinsics, create an application context, open a display connection, and create an application shell.

In addition, two new variables are provided in support of the new session shell widget:

```
sessionShellClassRec
sessionShellWidgetClass
```

See the *X Toolkit Intrinsics - C Language Interface, X Window System, X Version 11 Release 6.4* specification available from X.Org for detailed information about each of these functions and variables.

4.5.2 Support for Easy Resource Configuration

V1.3

Setting and changing resources in X Window System applications can be difficult for both the application developer and the end user. Resource Configuration Management (RCM) addresses this problem by changing the X Intrinsics to immediately modify a resource for the specified widget and each child widget in the hierarchy. As a result:

- No sourcing of resource files is required.
- The application does not need to be restarted for the new resource values to take effect.

- The change occurs immediately.

RCM was made available as part of the X11R6.4 release and is now available with DECwindows Motif Version 1.3. However, note that RCM is not a standard part of the X Toolkit Intrinsics. It is neither an X Consortium standard nor part of the X Window System specifications. As a result, there are currently no public customization tools that take advantage of this feature.

If you are interested in learning more about RCM, see the *X Toolkit Intrinsics - C Language Interface, X Window System, X Version 11 Release 6.4 Release Notes* available from X.Org.

4.5.3 New Option for CompositeClassExtensionRec

V1.3

With X11R6, some modifications were made to the widget internals, as described in the *X Window System Toolkit* manual.

In particular, a new option in the Composite Class extension record enables you to make bundled changes to the managed set of a Composite widget. Widgets that define a change-managed procedure that performs additions and deletions to the managed set of children in a single invocation should set `allows_change_managed_set` option to TRUE in the extension record.

For more information about the impact this new option may have on existing applications, see the *HP DECwindows Motif for OpenVMS Release Notes*.

4.5.4 New Default Format for XtResolvePathname

V1.2–6

In `XtResolvePathname`, the default pathname is required to have certain properties when either no other path information is present in the call, or when it is referenced by the environment variable `XFILESEARCHPATH`. The former default OpenVMS format of the pathname consisted of a type-name-suffix substitution. The modified pathname now reflects the 6-part fallback, as specified by X11 Release 6.

The new pathname behavior is enabled by setting the `DECW$VSW_COMPLIANT` variable, as follows:

```
$ DEFINE DECW$VSW_COMPLIANT 1
```

4.5.5 XtAppMainLoop Routine

V1.2–5

Previously, if a program entered its event loop, (for example, by calling `XtAppMainLoop`) without having opened a display or specified a timer or event flag for the program to wait for (by calling `XtAppAddTimeout` or `XtAppAddInput`), Xlib terminated the program with the following error message:

```
X Toolkit Error: Error in XMultiplexInput
```

Starting with DECwindows Motif for OpenVMS Version 1.2–5, if there is nothing to wait for, Xlib stalls waiting for input instead of terminating with an error status.

Programming Features

4.5 X Toolkit Intrinsic (Xt) Programming

To allow Xlib to process events at a later time, applications should provide some means of regaining control, such as specifying an event flag (on DECwindows Motif for OpenVMS Version 1.2–6 and previous systems) or a logical connection number (on DECwindows Motif Version 1.3 and higher systems) by calling `XtAppAddInput`.

4.6 X Window System Library (Xlib)

The following sections describe features related to X Window System library (Xlib).

4.6.1 IPv6 Considerations

V1.5

The Xlib library now supports both IPv4- and IPv6-compliant transport names discussed in *HP DECwindows Motif for OpenVMS Management Guide*.

The `XOpenDisplay` function cannot be called from a user-mode AST when IPv6 support is enabled. This function uses the TCP/IP Services `getaddrinfo` function which is not supported in user-mode ASTs. If your program must use the `XOpenDisplay` function from within a user-mode AST, you must ensure that the program is run in an environment where the `DECW$IPV6_SUPPORT` logical is defined as “DISABLED”.

4.6.2 New Functions Available with X11R6.6 Upgrade

V1.3

As part of the core system upgrade to X11R6.6, the following functions listed in Table 4–8 have been added to the X Window System Library (Xlib). Note that all functions are included in the current version of the `DECW$XLIBSHR` image.

Table 4–8 New Xlib Functions Supported for X11R6.6

Function Name	Description
<code>_XAllocTemp</code>	Thread-safe allocation of scratch data space for use by extension writers.
<code>_XFreeTemp</code>	Frees the scratch data space allocated by <code>_XAllocTemp</code> .
<code>XCcloseOM</code>	Closes the specified output method.
<code>XcmsSetCCOfColormap</code>	Sets the color conversion context for the specified colormap.
<code>XAddConnectionWatch</code>	Establishes a watch procedure callback for when internal connections are opened or closed.
<code>XConvertCase</code>	Obtains the uppercase and lowercase forms of a <code>KeySym</code> .
<code>XContextualDrawing</code>	Indicates whether text drawn with the current font set includes context-dependent drawing.
<code>XCreateOC</code>	Creates an output context within the specified output method.
<code>XDestroyOC</code>	Destroys an output context.
<code>XDirectionalDependentDrawing</code>	Indicates whether the drawing functions implement text directionality.

(continued on next page)

Table 4–8 (Cont.) New Xlib Functions Supported for X11R6.6

Function Name	Description
XDisplayOfOM	Returns the display associated with the specified output method.
XESetBeforeFlush	Defines a procedure that will be called just before data is sent to the X server.
XExtendedMaxRequestSize	Returns the maximum request size using extended length encoding (the BIG-REQUESTS extension).
XGetAtomNames	Returns the names associated with the specified X atoms.
XGetOCValues	Obtains the current output context values.
XGetOMValues	Obtains the current output method values.
XInitThreads	Initializes support for multiple threads.
XInternalConnectionNumbers	Returns a list of the internal connections open for a specified display.
XInternAtoms	Returns atoms for an array of names.
XLocaleOfOM	Returns the locale associated with the specified output method.
XLockDisplay	Locks a display to protect against concurrent access from multiple threads.
XOMOfOC	Returns the output method associated with the specified output context.
XOpenOM	Opens an X output method for the specified locale and modifiers.
XProcessInternalConnection	Processes input available on an internal connection.
XReadBitmapFileData	Reads a bitmap from a file and returns it as data.
XRegisterIMInstantiateCallback	Registers an input method callback.
XRemoveConnectionWatch	Removes a watch procedure established by XAddConnectionWatch.
XSetOCValues	Specifies one or more output context values.
XSetOMValues	Specifies one or more output method values.
XUnlockDisplay	Removes a lock established by XLockDisplay.
XUnregisterIMInstantiateCallback	Unregisters an input method instantiation callback.

See the *Xlib - C Language X Interface, X Consortium Standard, X Version 11, Release 6.4* specification available from X.Org for detailed information about each of these functions.

4.6.3 Updated Client-Side Extension Library

V1.3

The client-side extension library (DECW\$XEXTLIBSHR) has been updated to support multithreading and new header files for the following extensions available as part of the upgrade to X11R6.6:

- Application Group (XC-APPGROUP)
- Colormap Utilization Policy (TOG-CUP)
- Extended Visual Information (EVI)

Programming Features

4.6 X Window System Library (Xlib)

- Low-Bandwidth X (LBX)
- Security (SECURITY)
- Synchronization (SYNC)
- X Double Buffer (DBE)
- XINERAMA (formerly Panoramix)
- X Print (Xp)

All extensions in the library (new and existing) have been made thread-safe (as described in Section 4.1.1). In addition, the minor version of the library has been updated from 2,2 to 2,3 to prevent images linked against the updated DECW\$XEXTLIBSHR from loading the incorrect version of the library.

Also, function names longer than 31 characters have been replaced by macro definitions compatible with the current version of the OpenVMS Linker.

See Section 4.7.1 for an overview of each of these extensions. For instructions on how to link to this library, see Section 4.6.9.

4.6.4 Support for LCNs

V1.3

Xlib now provides an alternate means of obtaining connection numbers for connections to DECwindows Motif Version 1.3 or higher servers. The logical connection number (LCN) interface was specifically designed to support the communication needs of X11R6 systems and is intended as a replacement for the EFN mechanism.

The following functions and macros are designed for use with the new LCN interface:

- XAddConnectionWatch function (registers watch procedure)
- XInternalConnectionNumbers function
- XConnectionNumber function
- ConnectionNumber macro

These Xlib functions and macros are described in the *Xlib - C Language X Interface, X Consortium Standard, X Version 11, Release 6.4* specification available from X.Org.

Note for compatibility with DECwindows Motif for OpenVMS Version 1.2–6 and earlier clients, the existing event flag mechanism remains unchanged, and the XtAppAddInput function accepts both EFNs and LCNs. However, HP recommends that new applications, in particular ones that use multithreading, use LCNs. When Xlib has multithreading enabled, EFNs are not available. If a multithreaded application uses EFNs without multithreading enabled in Xlib, the EFN should be restricted to a single thread—the same thread used for all X calls.

For more information about the LCN interface and its available routines, see Section 4.8.2.

4.6.5 Updated X11 Environment Variable Parsing

V1.3

Xlib now accepts the equivalent X11 Release 6 (X11R6) POSIX-compliant forms of the following environment variables:

OpenVMS Form	X11R6 Form
DECW\$DISPLAY	DISPLAY
DECW\$RESOURCE_NAME	RESOURCE_NAME ¹

¹Also requires the symbol DECW\$VSW_COMPLIANT.

On connection to the X display server, Xlib checks the variable name. If the OpenVMS variable is not defined, Xlib checks for the X11R6 equivalent before returning a status value.

4.6.6 Additional Non-C Language Bindings Available with X11R6.6

Non-C language bindings (such as Fortran and Pascal) for the following Xlib functions have been added to DECwindows Motif Version 1.3. These bindings are in addition to those documented in the *DECwindows Motif for OpenVMS Guide to Non-C Bindings*.

```
X$CLOSE_OM
X$CONTEXTUAL_DRAWING
X$CONVERT_CASE
X$DESTROY_OC
X$DIRECTIONAL_DEPENDENT_DRAWING
X$DISPLAY_OF_OM
X$EXTENDED_MAX_REQUEST_SIZE
X$INIT_IMAGE
X$INIT_THREADS
X$INTERNAL_CONNECTION_NUMBERS
X$LOCALE_OF_OM
X$LOCK_DISPLAY
X$OPEN_OM
X$PROCESS_INTERNAL_CONNECTION
X$REGISTER_IM_INSTANTIATE_CB
X$SET_AUTHORIZATION
X$UNLOCK_DISPLAY
X$UNREGISTER_IM_INSTANTIATE_CB
```

4.6.6.1 CLOSE OM

OpenVMS Format

```
status_return = X$CLOSE_OM
```

```
(om)
```

Argument Information

Argument	Usage	Data Type	Access	Mechanism
status_return	longword	longword	write	value
om	identifier	uns longword	read	reference

Programming Features

4.6 X Window System Library (Xlib)

4.6.6.2 CONTEXTUAL DRAWING

OpenVMS Format

status_return = X\$CONTEXTUAL_DRAWING

(font_set)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
status_return	cond_value	longword	write	value
font_set	identifier	uns longword	read	reference

4.6.6.3 CONVERT CASE

OpenVMS Format

X\$CONVERT_CASE

(sym, lower, upper)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
sym	uns longword	uns longword	read	reference
lower	uns longword	uns longword	write	reference
upper	uns longword	uns longword	write	reference

4.6.6.4 DESTROY OC

OpenVMS Format

X\$DESTROY_OC

(oc)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
oc	uns longword	uns longword	write	reference

4.6.6.5 DIRECTIONAL DEPENDENT DRAWING

OpenVMS Format

status_return = X\$DIRECTIONAL_DEPENDENT_DRAWING

(font_set)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
status_return	cond_value	longword	write	value
font_set	identifier	uns longword	read	reference

4.6.6.6 DISPLAY OF OM

OpenVMS Format

display_return = X\$DISPLAY_OF_OM

(om)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
display_return	identifier	uns longword	write	value
om	uns longword	uns longword	read	reference

4.6.6.7 EXTENDED MAX REQUEST SIZE

OpenVMS Format

req_size_return = X\$EXTENDED_MAX_REQUEST_SIZE

(display)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
req_size_return	longword	longword	write	value
display	identifier	uns longword	read	reference

4.6.6.8 INIT IMAGE

OpenVMS Format

status_return = X\$INIT_IMAGE

(ximage)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
status_return	cond_value	longword	write	value
ximage	record	x\$image	read	reference

4.6.6.9 INIT THREADS

OpenVMS Format

status_return = X\$INIT_THREADS

Argument Information

Argument	Usage	Data Type	Access	Mechanism
status_return	cond_value	longword	write	value

Programming Features

4.6 X Window System Library (Xlib)

4.6.6.10 INTERNAL CONNECTION NUMBERS

OpenVMS Format

status_return = X\$INTERNAL_CONNECTION_NUMBERS

(display, fdptr, nptr)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
status_return	cond_value	longword	write	value
display	identifier	uns longword	read	reference
fdptr	longword	longword	write	reference
nptr	longword	longword	write	reference

4.6.6.11 LOCALE OF OM

OpenVMS Format

return_value = X\$LOCALE_OF_OM

(om, return_string)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
return_value	longword	longword	write	value
om	uns longword	uns longword	read	reference
return_string	char_string	character string	write	descriptor

4.6.6.12 LOCK DISPLAY

OpenVMS Format

X\$LOCK_DISPLAY

(display)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
display	identifier	uns longword	read	reference

4.6.6.13 OPEN OM

OpenVMS Format

om_return = X\$OPEN_OM

(display, db, str1, str2)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
om_return	uns longword	uns longword	write	reference
display	identifier	uns longword	read	reference
db	identifier	uns longword	read	reference
str1	char_string	character string	read	descriptor
str2	char_string	character string	read	descriptor

4.6.6.14 PROCESS INTERNAL CONNECTION

OpenVMS Format

X\$PROCESS_INTERNAL_CONNECTION

(display, fdptr)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
display	identifier	uns longword	read	reference
fdptr	longword	longword	read	reference

4.6.6.15 REGISTER IM INSTANTIATE Callback

OpenVMS Format

return_value = X\$REGISTER_IM_INSTANTIATE_CB

(display, database, name_str, class_str, callback, client_data)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
return_value	longword	longword	write	value
display	identifier	uns longword	read	reference
database	identifier	uns longword	read	reference
name_str	char_string	character string	read	descriptor
class_str	char_string	character string	read	descriptor
callback	procedure	proc entry mask	read	value
client_data	char_string	character string	read	descriptor

4.6.6.16 SET AUTHORIZATION

OpenVMS Format

X\$SET_AUTHORIZATION

(name, data)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
name	char_string	character string	read	descriptor
data	char_string	character string	read	descriptor

4.6.6.17 UNLOCK DISPLAY

OpenVMS Format

X\$UNLOCK_DISPLAY

(display)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
display	identifier	uns longword	read	reference

Programming Features

4.6 X Window System Library (Xlib)

4.6.6.18 UNREGISTER IM INSTANTIATE Callback

OpenVMS Format

return_value = X\$UNREGISTER_IM_INSTANTIATE_CB

(display, database, name_str, class_str, callback, client_data)

Argument Information

Argument	Usage	Data Type	Access	Mechanism
return_value	longword	longword	write	value
display	identifier	uns longword	read	reference
database	identifier	uns longword	read	reference
name_str	char_string	character string	read	descriptor
class_str	char_string	character string	read	descriptor
callback	procedure	proc entry mask	read	value
client_data	char_string	character string	read	descriptor

4.6.7 Support for Additional Fonts

V1.3

DECwindows Motif Version 1.3 offers support for the following additional fonts and font technologies:

- **Agfa Monotype Windows-Compatible TrueType fonts** – To ensure fast, high-quality text rendering capabilities, DECwindows Motif includes the iType font rendering technology from Agfa Monotype Corporation along with a number of scalable fonts, including the Albany, Cumberland, Screen, and Thorndale type families. These fonts are identical in screen and printer metrics to the Windows core fonts Arial, Courier, and Times New Roman. Agfa Monotype's Windows-compatible fonts are part of the Enhanced Screen Quality (ESQ) line of TrueType fonts optimized for viewing at any resolution. For more information about the iType technology, visit the Agfa Monotype web site (<http://www.agfamonotype.com>).
- **X11R6.6 fonts** – As part of the standard X11R6.6 implementation, DECwindows Motif now includes the 75- and 100-dpi versions of the Bitstream Charter and Adobe Utopia font families.
- **Previously undocumented fonts** – These fonts include the 75-dpi, 100-dpi, and common versions of the Lucida, Present Bullets, Fixed Width, Sun Open Look Glyph, and VT330 font families, and well as a set of language-specific and miscellaneous fonts. Also included are the Bitstream Speedo, Adobe Type1 Courier, and Utopia scalable fonts.

The following sections further describe the new font families, which can be loaded as described in Chapter 8 of the *VMS DECwindows Guide to Xlib (Release 4) Programming: MIT C Binding* manual. Each section provides the following information:

- Location and format of the font files
- List of the individual font file and font names in X Logical Font Description (XLFD) format

The file and font names are provided in a series of tables that are intended to supplement the existing font tables found in Appendix C of the *VMS DECwindows Guide to Xlib (Release 4) Programming: MIT C Binding* manual.

Note

File names containing consecutive underscore characters (_) or hyphens (-) may appear to contain a space between the consecutive characters. In all cases, the space is not present in the font name.

4.6.7.1 Additional 75-dpi Fonts

Table 4–9 lists the new and previously undocumented 75-dpi fonts and their file names. The files for these fonts are located in the following directory:

DECW\$SYSCOMMON:[SYSFONT.DECW.75DPI]

Table 4–9 Additional 75-dpi Fonts (.PCF File Extension)

File Name	Font Name
Charter	
CHARTER08	-Bitstream-Charter-Medium-R-Normal- -8-80-75-75-P-45-ISO8859-1
CHARTER10	-Bitstream-Charter-Medium-R-Normal- -10-100-75-75-P-56-ISO8859-1
CHARTER12	-Bitstream-Charter-Medium-R-Normal- -12-120-75-75-P-67-ISO8859-1
CHARTER14	-Bitstream-Charter-Medium-R-Normal- -15-140-75-75-P-84-ISO8859-1
CHARTER18	-Bitstream-Charter-Medium-R-Normal- -19-180-75-75-P-106-ISO8859-1
CHARTER24	-Bitstream-Charter-Medium-R-Normal- -25-240-75-75-P-139-ISO8859-1
CHARTER_BOLD_ITALIC08	-Bitstream-Charter-Bold-I-Normal- -8-80-75-75-P-50-ISO8859-1
CHARTER_BOLD_ITALIC10	-Bitstream-Charter-Bold-I-Normal- -10-100-75-75-P-62-ISO8859-1
CHARTER_BOLD_ITALIC14	-Bitstream-Charter-Bold-I-Normal- -15-140-75-75-P-93-ISO8859-1
CHARTER_BOLD_ITALIC12	-Bitstream-Charter-Bold-I-Normal- -12-120-75-75-P-74-ISO8859-1
CHARTER_BOLD_ITALIC18	-Bitstream-Charter-Bold-I-Normal- -19-180-75-75-P-117-ISO8859-1
CHARTER_BOLD_ITALIC24	-Bitstream-Charter-Bold-I-Normal- -25-240-75-75-P-154-ISO8859-1
CHARTER_ITALIC08	-Bitstream-Charter-Medium-I-Normal- -8-80-75-75-P-44-ISO8859-1
CHARTER_ITALIC10	-Bitstream-Charter-Medium-I-Normal- -10-100-75-75-P-55-ISO8859-1
CHARTER_ITALIC12	-Bitstream-Charter-Medium-I-Normal- -12-120-75-75-P-65-ISO8859-1
CHARTER_ITALIC14	-Bitstream-Charter-Medium-I-Normal- -15-140-75-75-P-82-ISO8859-1
CHARTER_ITALIC18	-Bitstream-Charter-Medium-I-Normal- -19-180-75-75-P-103-ISO8859-1
CHARTER_ITALIC24	-Bitstream-Charter-Medium-I-Normal- -25-240-75-75-P-136-ISO8859-1
CHARTER_BOLD08	-Bitstream-Charter-Bold-R-Normal- -8-80-75-75-P-50-ISO8859-1
CHARTER_BOLD10	-Bitstream-Charter-Bold-R-Normal- -10-100-75-75-P-63-ISO8859-1
CHARTER_BOLD12	-Bitstream-Charter-Bold-R-Normal- -12-120-75-75-P-75-ISO8859-1

(continued on next page)

Programming Features

4.6 X Window System Library (Xlib)

Table 4–9 (Cont.) Additional 75-dpi Fonts (.PCF File Extension)

File Name	Font Name
Charter	
CHARTER_BOLD14	-Bitstream-Charter-Bold-R-Normal- -15-140-75-75-P-94-ISO8859-1
CHARTER_BOLD18	-Bitstream-Charter-Bold-R-Normal- -19-180-75-75-P-119-ISO8859-1
CHARTER_BOLD24	-Bitstream-Charter-Bold-R-Normal- -25-240-75-75-P-157-ISO8859-1
Lucida	
LUCIDABRIGHT08	-B&H-LucidaBright-Medium-R-Normal- -8-80-75-75-P-45-ISO8859-1
LUCIDABRIGHT10	-B&H-LucidaBright-Medium-R-Normal- -10-100-75-75-P-56-ISO8859-1
LUCIDABRIGHT12	-B&H-LucidaBright-Medium-R-Normal- -12-120-75-75-P-68-ISO8859-1
LUCIDABRIGHT14	-B&H-LucidaBright-Medium-R-Normal- -14-140-75-75-P-80-ISO8859-1
LUCIDABRIGHT18	-B&H-LucidaBright-Medium-R-Normal- -18-180-75-75-P-103-ISO8859-1
LUCIDABRIGHT19	-B&H-LucidaBright-Medium-R-Normal- -19-190-75-75-P-109-ISO8859-1
LUCIDABRIGHT24	-B&H-LucidaBright-Medium-R-Normal- -24-240-75-75-P-137-ISO8859-1
LUCIDABRIGHT_DEMIITALIC08	-B&H-LucidaBright-Demibold-I-Normal- -8-80-75-75-P-48-ISO8859-1
LUCIDABRIGHT_DEMIITALIC10	-B&H-LucidaBright-Demibold-I-Normal- -10-100-75-75-P-59-ISO8859-1
LUCIDABRIGHT_DEMIITALIC12	-B&H-LucidaBright-Demibold-I-Normal- -12-120-75-75-P-72-ISO8859-1
LUCIDABRIGHT_DEMIITALIC14	-B&H-LucidaBright-Demibold-I-Normal- -14-140-75-75-P-84-ISO8859-1
LUCIDABRIGHT_DEMIITALIC18	-B&H-LucidaBright-Demibold-I-Normal- -18-180-75-75-P-107-ISO8859-1
LUCIDABRIGHT_DEMIITALIC19	-B&H-LucidaBright-Demibold-I-Normal- -19-190-75-75-P-114-ISO8859-1
LUCIDABRIGHT_DEMIITALIC24	-B&H-LucidaBright-Demibold-I-Normal- -24-240-75-75-P-143-ISO8859-1
LUCIDABRIGHT_ITALIC08	-B&H-LucidaBright-Medium-I-Normal- -8-80-75-75-P-45-ISO8859-1
LUCIDABRIGHT_ITALIC10	-B&H-LucidaBright-Medium-I-Normal- -10-100-75-75-P-57-ISO8859-1
LUCIDABRIGHT_ITALIC12	-B&H-LucidaBright-Medium-I-Normal- -12-120-75-75-P-67-ISO8859-1
LUCIDABRIGHT_ITALIC14	-B&H-LucidaBright-Medium-I-Normal- -14-140-75-75-P-80-ISO8859-1
LUCIDABRIGHT_ITALIC18	-B&H-LucidaBright-Medium-I-Normal- -18-180-75-75-P-102-ISO8859-1
LUCIDABRIGHT_ITALIC19	-B&H-LucidaBright-Medium-I-Normal- -19-190-75-75-P-109-ISO8859-1
LUCIDABRIGHT_ITALIC24	-B&H-LucidaBright-Medium-I-Normal- -24-240-75-75-P-136-ISO8859-1
LUCIDABRIGHT_DEMI08	-B&H-LucidaBright-Demibold-R-Normal- -8-80-75-75-P-47-ISO8859-1
LUCIDABRIGHT_DEMI10	-B&H-LucidaBright-Demibold-R-Normal- -10-100-75-75-P-59-ISO8859-1
LUCIDABRIGHT_DEMI12	-B&H-LucidaBright-Demibold-R-Normal- -12-120-75-75-P-71-ISO8859-1
LUCIDABRIGHT_DEMI14	-B&H-LucidaBright-Demibold-R-Normal- -14-140-75-75-P-84-ISO8859-1
LUCIDABRIGHT_DEMI18	-B&H-LucidaBright-Demibold-R-Normal- -18-180-75-75-P-107-ISO8859-1
LUCIDABRIGHT_DEMI19	-B&H-LucidaBright-Demibold-R-Normal- -19-190-75-75-P-114-ISO8859-1

(continued on next page)

Table 4–9 (Cont.) Additional 75-dpi Fonts (.PCF File Extension)

File Name	Font Name
Lucida	
LUCIDABRIGHT_DEMI24	-B&H-LucidaBright-Demibold-R-Normal- -24-240-75-75-P-143-ISO8859-1
LUCIDATYPEWRITER_BOLD SANS08	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-8-80-75-75-m-50-ISO8859-1
LUCIDATYPEWRITER_BOLD SANS10	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-10-100-75-75-m-60-ISO8859-1
LUCIDATYPEWRITER_BOLD SANS12	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-12-120-75-75-m-70-ISO8859-1
LUCIDATYPEWRITER_BOLD SANS14	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-14-140-75-75-m-90-ISO8859-1
LUCIDATYPEWRITER_BOLD SANS18	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-18-180-75-75-m-110-ISO8859-1
LUCIDATYPEWRITER_BOLD SANS19	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-19-190-75-75-m-110-ISO8859-1
LUCIDATYPEWRITER_BOLD SANS24	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-24-240-75-75-m-140-ISO8859-1
LUCIDATYPEWRITER_SANS08	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-8-80-75-75-m-50-ISO8859-1
LUCIDATYPEWRITER_SANS10	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-10-100-75-75-m-60-ISO8859-1
LUCIDATYPEWRITER_SANS12	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-12-120-75-75-m-70-ISO8859-1
LUCIDATYPEWRITER_SANS14	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-14-140-75-75-m-90-ISO8859-1
LUCIDATYPEWRITER_SANS18	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-18-180-75-75-m-110-ISO8859-1
LUCIDATYPEWRITER_SANS19	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-19-190-75-75-m-110-ISO8859-1
LUCIDATYPEWRITER_SANS24	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-24-240-75-75-m-140-ISO8859-1
LUCIDA_BOLDITALICSANS08	-B&H-Lucida-Bold-I-Normal-Sans-8-80-75-75-P-49-ISO8859-1
LUCIDA_BOLDITALICSANS10	-B&H-Lucida-Bold-I-Normal-Sans-10-100-75-75-P-67-ISO8859-1
LUCIDA_BOLDITALICSANS12	-B&H-Lucida-Bold-I-Normal-Sans-12-120-75-75-P-79-ISO8859-1
LUCIDA_BOLDITALICSANS14	-B&H-Lucida-Bold-I-Normal-Sans-14-140-75-75-P-92-ISO8859-1
LUCIDA_BOLDITALICSANS18	-B&H-Lucida-Bold-I-Normal-Sans-18-180-75-75-P-119-ISO8859-1
LUCIDA_BOLDITALICSANS19	-B&H-Lucida-Bold-I-Normal-Sans-19-190-75-75-P-122-ISO8859-1
LUCIDA_BOLDITALICSANS24	-B&H-Lucida-Bold-I-Normal-Sans-24-240-75-75-P-151-ISO8859-1
LUCIDA_BOLDANS08	-B&H-Lucida-Bold-R-Normal-Sans-8-80-75-75-P-50-ISO8859-1
LUCIDA_BOLDANS10	-B&H-Lucida-Bold-R-Normal-Sans-10-100-75-75-P-66-ISO8859-1
LUCIDA_BOLDANS12	-B&H-Lucida-Bold-R-Normal-Sans-12-120-75-75-P-79-ISO8859-1
LUCIDA_BOLDANS14	-B&H-Lucida-Bold-R-Normal-Sans-14-140-75-75-P-92-ISO8859-1
LUCIDA_BOLDANS18	-B&H-Lucida-Bold-R-Normal-Sans-18-180-75-75-P-120-ISO8859-1
LUCIDA_BOLDANS19	-B&H-Lucida-Bold-R-Normal-Sans-19-190-75-75-P-122-ISO8859-1
LUCIDA_BOLDANS24	-B&H-Lucida-Bold-R-Normal-Sans-24-240-75-75-P-152-ISO8859-1
LUCIDA_ITALICSANS08	-B&H-Lucida-Medium-I-Normal-Sans-8-80-75-75-P-45-ISO8859-1
LUCIDA_ITALICSANS10	-B&H-Lucida-Medium-I-Normal-Sans-10-100-75-75-P-59-ISO8859-1
LUCIDA_ITALICSANS12	-B&H-Lucida-Medium-I-Normal-Sans-12-120-75-75-P-71-ISO8859-1

(continued on next page)

Programming Features

4.6 X Window System Library (Xlib)

Table 4–9 (Cont.) Additional 75-dpi Fonts (.PCF File Extension)

File Name	Font Name
Lucida	
LUCIDA_ITALICSANS14	-B&H-Lucida-Medium-I-Normal-Sans-14-140-75-75-P-82-ISO8859-1
LUCIDA_ITALICSANS18	-B&H-Lucida-Medium-I-Normal-Sans-18-180-75-75-P-105-ISO8859-1
LUCIDA_ITALICSANS19	-B&H-Lucida-Medium-I-Normal-Sans-19-190-75-75-P-108-ISO8859-1
LUCIDA_ITALICSANS24	-B&H-Lucida-Medium-I-Normal-Sans-24-240-75-75-P-136-ISO8859-1
LUCIDA_SANS08	-B&H-Lucida-Medium-R-Normal-Sans-8-80-75-75-P-45-ISO8859-1
LUCIDA_SANS10	-B&H-Lucida-Medium-R-Normal-Sans-10-100-75-75-P-58-ISO8859-1
LUCIDA_SANS12	-B&H-Lucida-Medium-R-Normal-Sans-12-120-75-75-P-71-ISO8859-1
LUCIDA_SANS14	-B&H-Lucida-Medium-R-Normal-Sans-14-140-75-75-P-81-ISO8859-1
LUCIDA_SANS18	-B&H-Lucida-Medium-R-Normal-Sans-18-180-75-75-P-106-ISO8859-1
LUCIDA_SANS19	-B&H-Lucida-Medium-R-Normal-Sans-19-190-75-75-P-108-ISO8859-1
LUCIDA_SANS24	-B&H-Lucida-Medium-R-Normal-Sans-24-240-75-75-P-136-ISO8859-1
Present Bullets	
PRESENT_BULLETS8_75	-DEC-PresentBullets-Medium-R-Normal- -8-80-75-75-P-76-DEC-FontSpecific
PRESENT_BULLETS10_75	-DEC-PresentBullets-Medium-R-Normal- -10-100-75-75-P-96-DEC-FontSpecific
PRESENT_BULLETS12_75	-DEC-PresentBullets-Medium-R-Normal- -12-120-75-75-P-114-DEC-FontSpecific
PRESENT_BULLETS14_75	-DEC-PresentBullets-Medium-R-Normal- -14-140-75-75-P-134-DEC-FontSpecific
PRESENT_BULLETS18_75	-DEC-PresentBullets-Medium-R-Normal- -18-180-75-75-P-172-DEC-FontSpecific
PRESENT_BULLETS24_75	-DEC-PresentBullets-Medium-R-Normal- -24-240-75-75-P-229-DEC-FontSpecific
PRESENT_BULLETS36_75	-DEC-PresentBullets-Medium-R-Normal- -36-360-75-75-P-343-DEC-FontSpecific
PRESENT_BULLETS48_75	-DEC-PresentBullets-Medium-R-Normal- -48-480-75-75-P-458-DEC-FontSpecific
PRESENT_BULLETS72_75	-DEC-PresentBullets-Medium-R-Normal- -72-720-75-75-P-686-DEC-FontSpecific
Utopia	
UTOPIA10	-Adobe-Utopia-Regular-R-Normal- -10-100-75-75-P-56-ISO8859-1
UTOPIA12	-Adobe-Utopia-Regular-R-Normal- -12-120-75-75-P-67-ISO8859-1
UTOPIA14	-Adobe-Utopia-Regular-R-Normal- -15-140-75-75-P-79-ISO8859-1
UTOPIA18	-Adobe-Utopia-Regular-R-Normal- -19-180-75-75-P-101-ISO8859-1
UTOPIA24	-Adobe-Utopia-Regular-R-Normal- -25-240-75-75-P-135-ISO8859-1
UTOPIA_BOLD10	-Adobe-Utopia-Bold-R-Normal- -10-100-75-75-P-59-ISO8859-1
UTOPIA_BOLD12	-Adobe-Utopia-Bold-R-Normal- -12-120-75-75-P-70-ISO8859-1
UTOPIA_BOLD14	-Adobe-Utopia-Bold-R-Normal- -15-140-75-75-P-82-ISO8859-1
UTOPIA_BOLD18	-Adobe-Utopia-Bold-R-Normal- -19-180-75-75-P-105-ISO8859-1
UTOPIA_BOLD24	-Adobe-Utopia-Bold-R-Normal- -25-240-75-75-P-140-ISO8859-1
UTOPIA_BOLD_ITALIC10	-Adobe-Utopia-Bold-I-Normal- -10-100-75-75-P-58-ISO8859-1
UTOPIA_BOLD_ITALIC12	-Adobe-Utopia-Bold-I-Normal- -12-120-75-75-P-70-ISO8859-1
UTOPIA_BOLD_ITALIC14	-Adobe-Utopia-Bold-I-Normal- -15-120-75-75-P-70-ISO8859-1

(continued on next page)

Table 4–9 (Cont.) Additional 75-dpi Fonts (.PCF File Extension)

File Name	Font Name
Utopia	
UTOPIA_BOLD_ITALIC18	-Adobe-Utopia-Bold-I-Normal- -19-180-75-75-P-105-ISO8859-1
UTOPIA_BOLD_ITALIC24	-Adobe-Utopia-Bold-I-Normal- -25-240-75-75-P-140-ISO8859-1
UTOPIA_ITALIC10	-Adobe-Utopia-Regular-I-Normal- -10-100-75-75-P-55-ISO8859-1
UTOPIA_ITALIC12	-Adobe-Utopia-Regular-I-Normal- -12-120-75-75-P-67-ISO8859-1
UTOPIA_ITALIC14	-Adobe-Utopia-Regular-I-Normal- -15-140-75-75-P-79-ISO8859-1
UTOPIA_ITALIC18	-Adobe-Utopia-Regular-I-Normal- -19-180-75-75-P-100-ISO8859-1
UTOPIA_ITALIC24	-Adobe-Utopia-Regular-I-Normal- -25-240-75-75-P-133-ISO8859-1

4.6.7.2 Additional 100-dpi Fonts

Table 4–10 lists the new and previously undocumented 100-dpi fonts and their file names. The files for these fonts are located in the following directory:

DECW\$SYSCOMMON:[SYSFONT.DECW.100DPI]

Table 4–10 Additional 100-dpi Fonts (.PCF File Extension)

File Name	Font Name
Charter	
CHARTER08_100DPI	-Bitstream-Charter-Medium-R-Normal- -11-80-100-100-P-61-ISO8859-1
CHARTER10_100DPI	-Bitstream-Charter-Medium-R-Normal- -14-100-100-100-P-78-ISO8859-1
CHARTER12_100DPI	-Bitstream-Charter-Medium-R-Normal- -17-120-100-100-P-95-ISO8859-1
CHARTER14_100DPI	-Bitstream-Charter-Medium-R-Normal- -19-140-100-100-P-106-ISO8859-1
CHARTER18_100DPI	-Bitstream-Charter-Medium-R-Normal- -25-180-100-100-P-139-ISO8859-1
CHARTER24_100DPI	-Bitstream-Charter-Medium-R-Normal- -33-240-100-100-P-183-ISO8859-1
CHARTER_BOLD08_100DPI	-Bitstream-Charter-Bold-R-Normal- -11-80-100-100-P-69-ISO8859-1
CHARTER_BOLD10_100DPI	-Bitstream-Charter-Bold-R-Normal- -14-100-100-100-P-88-ISO8859-1
CHARTER_BOLD12_100DPI	-Bitstream-Charter-Bold-R-Normal- -17-120-100-100-P-107-ISO8859-1
CHARTER_BOLD14_100DPI	-Bitstream-Charter-Bold-R-Normal- -19-140-100-100-P-119-ISO8859-1
CHARTER_BOLD18_100DPI	-Bitstream-Charter-Bold-R-Normal- -25-180-100-100-P-157-ISO8859-1
CHARTER_BOLD24_100DPI	-Bitstream-Charter-Bold-R-Normal- -33-240-100-100-P-206-ISO8859-1
CHARTER_BOLD_ITALIC08_100DPI	-Bitstream-Charter-Bold-I-Normal- -11-80-100-100-P-68-ISO8859-1
CHARTER_BOLD_ITALIC10_100DPI	-Bitstream-Charter-Bold-I-Normal- -14-100-100-100-P-86-ISO8859-1
CHARTER_BOLD_ITALIC12_100DPI	-Bitstream-Charter-Bold-I-Normal- -17-120-100-100-P-105-ISO8859-1
CHARTER_BOLD_ITALIC14_100DPI	-Bitstream-Charter-Bold-I-Normal- -19-140-100-100-P-117-ISO8859-1
CHARTER_BOLD_ITALIC18_100DPI	-Bitstream-Charter-Bold-I-Normal- -25-180-100-100-P-154-ISO8859-1

(continued on next page)

Programming Features

4.6 X Window System Library (Xlib)

Table 4–10 (Cont.) Additional 100-dpi Fonts (.PCF File Extension)

File Name	Font Name
Charter	
CHARTER_BOLD_ITALIC24_100DPI	-Bitstream-Charter-Bold-I-Normal- -33-240-100-100-P-203-ISO8859-1
CHARTER_ITALIC08_100DPI	-Bitstream-Charter-Medium-I-Normal- -11-120-100-100-P-92-ISO8859-1
CHARTER_ITALIC10_100DPI	-Bitstream-Charter-Medium-I-Normal- -14-120-100-100-P-92-ISO8859-1
CHARTER_ITALIC12_100DPI	-Bitstream-Charter-Medium-I-Normal- -17-120-100-100-P-92-ISO8859-1
CHARTER_ITALIC14_100DPI	-Bitstream-Charter-Medium-I-Normal- -19-140-100-100-P-103-ISO8859-1
CHARTER_ITALIC18_100DPI	-Bitstream-Charter-Medium-I-Normal- -25-180-100-100-P-136-ISO8859-1
CHARTER_ITALIC24_100DPI	-Bitstream-Charter-Medium-I-Normal- -33-240-100-100-P-179-ISO8859-1
Lucida	
LUCIDABRIGHT08_100DPI	-B&H-LucidaBright-Medium-R-Normal- -11-80-100-100-P-63-ISO8859-1
LUCIDABRIGHT10_100DPI	-B&H-LucidaBright-Medium-R-Normal- -14-100-100-100-P-80-ISO8859-1
LUCIDABRIGHT12_100DPI	-B&H-LucidaBright-Medium-R-Normal- -17-120-100-100-P-96-ISO8859-1
LUCIDABRIGHT14_100DPI	-B&H-LucidaBright-Medium-R-Normal- -20-140-100-100-P-114-ISO8859-1
LUCIDABRIGHT18_100DPI	-B&H-LucidaBright-Medium-R-Normal- -25-180-100-100-P-142-ISO8859-1
LUCIDABRIGHT19_100DPI	-B&H-LucidaBright-Medium-R-Normal- -26-190-100-100-P-149-ISO8859-1
LUCIDABRIGHT24_100DPI	-B&H-LucidaBright-Medium-R-Normal- -34-240-100-100-P-193-ISO8859-1
LUCIDABRIGHT_DEMI08_100DPI	-B&H-LucidaBright-Demibold-R-Normal- -11-80-100-100-P-66-ISO8859-1
LUCIDABRIGHT_DEMI10_100DPI	-B&H-LucidaBright-Demibold-R-Normal- -14-100-100-100-P-84-ISO8859-1
LUCIDABRIGHT_DEMI12_100DPI	-B&H-LucidaBright-Demibold-R-Normal- -17-120-100-100-P-101-ISO8859-1
LUCIDABRIGHT_DEMI14_100DPI	-B&H-LucidaBright-Demibold-R-Normal- -20-140-100-100-P-118-ISO8859-1
LUCIDABRIGHT_DEMI18_100DPI	-B&H-LucidaBright-Demibold-R-Normal- -25-180-100-100-P-149-ISO8859-1
LUCIDABRIGHT_DEMI19_100DPI	-B&H-LucidaBright-Demibold-R-Normal- -26-190-100-100-P-155-ISO8859-1
LUCIDABRIGHT_DEMI24_100DPI	-B&H-LucidaBright-Demibold-R-Normal- -34-240-100-100-P-202-ISO8859-1
LUCIDABRIGHT_DEMITALIC08_100DPI	-B&H-LucidaBright-Demibold-I-Normal- -11-80-100-100-P-66-ISO8859-1
LUCIDABRIGHT_DEMITALIC10_100DPI	-B&H-LucidaBright-Demibold-I-Normal- -14-100-100-100-P-84-ISO8859-1
LUCIDABRIGHT_DEMITALIC12_100DPI	-B&H-LucidaBright-Demibold-I-Normal- -17-120-100-100-P-101-ISO8859-1
LUCIDABRIGHT_DEMITALIC14_100DPI	-B&H-LucidaBright-Demibold-I-Normal- -20-140-100-100-P-119-ISO8859-1
LUCIDABRIGHT_DEMITALIC18_100DPI	-B&H-LucidaBright-Demibold-I-Normal- -25-180-100-100-P-149-ISO8859-1

(continued on next page)

Table 4–10 (Cont.) Additional 100-dpi Fonts (.PCF File Extension)

File Name	Font Name
Lucida	
LUCIDABRIGHT_DEMIITALIC19_100DPI	-B&H-LucidaBright-Demibold-I-Normal- -26-190-100-100-P-156-ISO8859-1
LUCIDABRIGHT_DEMIITALIC24_100DPI	-B&H-LucidaBright-Demibold-I-Normal- -34-240-100-100-P-203-ISO8859-1
LUCIDABRIGHT_ITALIC08_100DPI	-B&H-LucidaBright-Medium-I-Normal- -11-80-100-100-P-63-ISO8859-1
LUCIDABRIGHT_ITALIC10_100DPI	-B&H-LucidaBright-Medium-I-Normal- -14-100-100-100-P-80-ISO8859-1
LUCIDABRIGHT_ITALIC12_100DPI	-B&H-LucidaBright-Medium-I-Normal- -17-120-100-100-P-96-ISO8859-1
LUCIDABRIGHT_ITALIC14_100DPI	-B&H-LucidaBright-Medium-I-Normal- -20-140-100-100-P-113-ISO8859-1
LUCIDABRIGHT_ITALIC18_100DPI	-B&H-LucidaBright-Medium-I-Normal- -25-180-100-100-P-142-ISO8859-1
LUCIDABRIGHT_ITALIC19_100DPI	-B&H-LucidaBright-Medium-I-Normal- -26-190-100-100-P-148-ISO8859-1
LUCIDABRIGHT_ITALIC24_100DPI	-B&H-LucidaBright-Medium-I-Normal- -34-240-100-100-P-194-ISO8859-1
LUCIDATYPEWRITER_BOLDSANS08_100DPI	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-11-80-100-100-m-70-ISO8859-1
LUCIDATYPEWRITER_BOLDSANS10_100DPI	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-14-100-100-100-m-80-ISO8859-1
LUCIDATYPEWRITER_BOLDSANS12_100DPI	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-17-120-100-100-m-100-ISO8859-1
LUCIDATYPEWRITER_BOLDSANS14_100DPI	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-20-140-100-100-m-120-ISO8859-1
LUCIDATYPEWRITER_BOLDSANS18_100DPI	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-25-180-100-100-m-150-ISO8859-1
LUCIDATYPEWRITER_BOLDSANS19_100DPI	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-26-190-100-100-m-159-ISO8859-1
LUCIDATYPEWRITER_BOLDSANS24_100DPI	-B&H-LucidaTypewriter-Bold-R-Normal-Sans-34-240-100-100-m-200-ISO8859-1
LUCIDATYPEWRITER_SANS08_100DPI	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-11-80-100-100-m-70-ISO8859-1
LUCIDATYPEWRITER_SANS10_100DPI	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-14-100-100-100-m-80-ISO8859-1
LUCIDATYPEWRITER_SANS12_100DPI	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-17-120-100-100-m-100-ISO8859-1
LUCIDATYPEWRITER_SANS14_100DPI	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-20-140-100-100-m-120-ISO8859-1
LUCIDATYPEWRITER_SANS18_100DPI	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-25-180-100-100-m-150-ISO8859-1
LUCIDATYPEWRITER_SANS19_100DPI	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-26-190-100-100-m-159-ISO8859-1
LUCIDATYPEWRITER_SANS24_100DPI	-B&H-LucidaTypewriter-Medium-R-Normal-Sans-34-240-100-100-m-200-ISO8859-1

(continued on next page)

Programming Features

4.6 X Window System Library (Xlib)

Table 4–10 (Cont.) Additional 100-dpi Fonts (.PCF File Extension)

File Name	Font Name
Lucida	
LUCIDA_BOLDITALICSANS08_100DPI	-B&H-Lucida-Bold-I-Normal-Sans-11-80-100-100-P-69-ISO8859-1
LUCIDA_BOLDITALICSANS10_100DPI	-B&H-Lucida-Bold-I-Normal-Sans-14-100-100-100-P-90-ISO8859-1
LUCIDA_BOLDITALICSANS12_100DPI	-B&H-Lucida-Bold-I-Normal-Sans-17-120-100-100-P-108-ISO8859-1
LUCIDA_BOLDITALICSANS14_100DPI	-B&H-Lucida-Bold-I-Normal-Sans-20-140-100-100-P-127-ISO8859-1
LUCIDA_BOLDITALICSANS18_100DPI	-B&H-Lucida-Bold-I-Normal-Sans-25-180-100-100-P-159-ISO8859-1
LUCIDA_BOLDITALICSANS19_100DPI	-B&H-Lucida-Bold-I-Normal-Sans-26-190-100-100-P-166-ISO8859-1
LUCIDA_BOLDITALICSANS24_100DPI	-B&H-Lucida-Bold-I-Normal-Sans-34-240-100-100-P-215-ISO8859-1
LUCIDA_BOLDANS08_100DPI	-B&H-Lucida-Bold-R-Normal-Sans-11-80-100-100-P-70-ISO8859-1
LUCIDA_BOLDANS10_100DPI	-B&H-Lucida-Bold-R-Normal-Sans-14-100-100-100-P-89-ISO8859-1
LUCIDA_BOLDANS12_100DPI	-B&H-Lucida-Bold-R-Normal-Sans-17-120-100-100-P-108-ISO8859-1
LUCIDA_BOLDANS14_100DPI	-B&H-Lucida-Bold-R-Normal-Sans-20-140-100-100-P-127-ISO8859-1
LUCIDA_BOLDANS18_100DPI	-B&H-Lucida-Bold-R-Normal-Sans-25-180-100-100-P-158-ISO8859-1
LUCIDA_BOLDANS19_100DPI	-B&H-Lucida-Bold-R-Normal-Sans-26-190-100-100-P-166-ISO8859-1
LUCIDA_BOLDANS24_100DPI	-B&H-Lucida-Bold-R-Normal-Sans-34-240-100-100-P-216-ISO8859-1
LUCIDA_ITALICSANS08_100DPI	-B&H-Lucida-Medium-I-Normal-Sans-11-80-100-100-P-62-ISO8859-1
LUCIDA_ITALICSANS10_100DPI	-B&H-Lucida-Medium-I-Normal-Sans-14-100-100-100-P-80-ISO8859-1
LUCIDA_ITALICSANS12_100DPI	-B&H-Lucida-Medium-I-Normal-Sans-17-120-100-100-P-97-ISO8859-1
LUCIDA_ITALICSANS14_100DPI	-B&H-Lucida-Medium-I-Normal-Sans-20-140-100-100-P-114-ISO8859-1
LUCIDA_ITALICSANS18_100DPI	-B&H-Lucida-Medium-I-Normal-Sans-25-180-100-100-P-141-ISO8859-1
LUCIDA_ITALICSANS19_100DPI	-B&H-Lucida-Medium-I-Normal-Sans-26-190-100-100-P-147-ISO8859-1
LUCIDA_ITALICSANS24_100DPI	-B&H-Lucida-Medium-I-Normal-Sans-34-240-100-100-P-192-ISO8859-1
LUCIDA_SANS08_100DPI	-B&H-Lucida-Medium-R-Normal-Sans-11-80-100-100-P-63-ISO8859-1
LUCIDA_SANS10_100DPI	-B&H-Lucida-Medium-R-Normal-Sans-14-100-100-100-P-80-ISO8859-1
LUCIDA_SANS12_100DPI	-B&H-Lucida-Medium-R-Normal-Sans-17-120-100-100-P-96-ISO8859-1
LUCIDA_SANS14_100DPI	-B&H-Lucida-Medium-R-Normal-Sans-20-140-100-100-P-114-ISO8859-1

(continued on next page)

Table 4–10 (Cont.) Additional 100-dpi Fonts (.PCF File Extension)

File Name	Font Name
Lucida	
LUCIDA_SANS18_100DPI	-B&H-Lucida-Medium-R-Normal-Sans-25-180-100-100-P-142-ISO8859-1
LUCIDA_SANS19_100DPI	-B&H-Lucida-Medium-R-Normal-Sans-26-190-100-100-P-147-ISO8859-1
LUCIDA_SANS24_100DPI	-B&H-Lucida-Medium-R-Normal-Sans-34-240-100-100-P-191-ISO8859-1
Present Bullets	
PRESENT_BULLETS8_100	-DEC-PresentBullets-Medium-R-Normal- -8-80-100-100-P-105-DEC-FontSpecific
PRESENT_BULLETS10_100	-DEC-PresentBullets-Medium-R-Normal- -10-100-100-100-P-123-DEC-FontSpecific
PRESENT_BULLETS12_100	-DEC-PresentBullets-Medium-R-Normal- -12-120-100-100-P-154-DEC-FontSpecific
PRESENT_BULLETS14_100	-DEC-PresentBullets-Medium-R-Normal- -14-140-100-100-P-172-DEC-FontSpecific
PRESENT_BULLETS18_100	-DEC-PresentBullets-Medium-R-Normal- -18-180-100-100-P-229-DEC-FontSpecific
PRESENT_BULLETS24_100	-DEC-PresentBullets-Medium-R-Normal- -24-240-100-100-P-305-DEC-FontSpecific
PRESENT_BULLETS36_100	-DEC-PresentBullets-Medium-R-Normal- -36-360-100-100-P-458-DEC-FontSpecific
PRESENT_BULLETS48_100	-DEC-PresentBullets-Medium-R-Normal- -48-480-100-100-P-609-DEC-FontSpecific
PRESENT_BULLETS72_100	-DEC-PresentBullets-Medium-R-Normal- -72-720-100-100-P-952-DEC-FontSpecific
Utopia	
UTOPIA10_100DPI	-Adobe-Utopia-Regular-R-Normal- -14-100-100-100-P-75-ISO8859-1
UTOPIA12_100DPI	-Adobe-Utopia-Regular-R-Normal- -17-120-100-100-P-91-ISO8859-1
UTOPIA14_100DPI	-Adobe-Utopia-Regular-R-Normal- -19-140-100-100-P-105-ISO8859-1
UTOPIA18_100DPI	-Adobe-Utopia-Regular-R-Normal- -25-180-100-100-P-135-ISO8859-1
UTOPIA24_100DPI	-Adobe-Utopia-Regular-R-Normal- -33-240-100-100-P-180-ISO8859-1
UTOPIA_BOLD10_100DPI	-Adobe-Utopia-Bold-R-Normal- -14-100-100-100-P-78-ISO8859-1
UTOPIA_BOLD12_100DPI	-Adobe-Utopia-Bold-R-Normal- -17-120-100-100-P-93-ISO8859-1
UTOPIA_BOLD14_100DPI	-Adobe-Utopia-Bold-R-Normal- -19-140-100-100-P-108-ISO8859-1
UTOPIA_BOLD18_100DPI	-Adobe-Utopia-Bold-R-Normal- -25-180-100-100-P-140-ISO8859-1
UTOPIA_BOLD24_100DPI	-Adobe-Utopia-Bold-R-Normal- -33-240-100-100-P-186-ISO8859-1
UTOPIA_BOLDITALIC10_100DPI	-Adobe-Utopia-Bold-I-Normal- -14-100-100-100-P-78-ISO8859-1
UTOPIA_BOLDITALIC12_100DPI	-Adobe-Utopia-Bold-I-Normal- -17-120-100-100-P-93-ISO8859-1
UTOPIA_BOLDITALIC14_100DPI	-Adobe-Utopia-Bold-I-Normal- -19-140-100-100-P-109-ISO8859-1
UTOPIA_BOLDITALIC18_100DPI	-Adobe-Utopia-Bold-I-Normal- -25-180-100-100-P-139-ISO8859-1
UTOPIA_BOLDITALIC24_100DPI	-Adobe-Utopia-Bold-I-Normal- -33-240-100-100-P-186-ISO8859-1
UTOPIA_ITALIC10_100DPI	-Adobe-Utopia-Regular-I-Normal- -14-100-100-100-P-74-ISO8859-1
UTOPIA_ITALIC12_100DPI	-Adobe-Utopia-Regular-I-Normal- -17-120-100-100-P-89-ISO8859-1

(continued on next page)

Programming Features

4.6 X Window System Library (Xlib)

Table 4–10 (Cont.) Additional 100-dpi Fonts (.PCF File Extension)

File Name	Font Name
Utopia	
UTOPIA_ITALIC14_100DPI	-Adobe-Utopia-Regular-I-Normal- -19-140-100-100-P-104-ISO8859-1
UTOPIA_ITALIC18_100DPI	-Adobe-Utopia-Regular-I-Normal- -25-180-100-100-P-134-ISO8859-1
UTOPIA_ITALIC24_100DPI	-Adobe-Utopia-Regular-I-Normal- -33-240-100-100-P-179-ISO8859-1

4.6.7.3 Additional Common Fonts

Table 4–11 lists previously undocumented Common fonts and their file names. The files for these fonts are located in the following directory:

DECW\$SYSCOMMON:[SYSFONT.DECW.COMMON]

Table 4–11 Additional Common Fonts (.PCF File Extension)

File Name	Font Name
Fixed Width	
5X7	-Misc-Fixed-Medium-R-Normal- -7-70-75-75-C-50-ISO8859-1
7X14B	-Misc-Fixed-Bold-R-Normal- -14-130-75-75-C-70-ISO8859-1
7X14RK	-Misc-Fixed-Medium-R-Normal- -14-130-75-75-C-70-JISX0201.1976-0
8X16RK	-Sony-Fixed-Medium-R-Normal- -16-120-100-100-C-80-JISX0201.1976-0
12X24	-Sony-Fixed-Medium-R-Normal- -24-170-100-100-C-120-ISO8859-1
12X24RK	-Sony-Fixed-Medium-R-Normal- -24-170-100-100-C-120-JISX0201.1976-0
Sun Open Look Glyph	
OLGL10	-Sun-Open Look Glyph- - - -10-100-75-75-P-101-SunOLGlyph-1
OLGL12	-Sun-Open Look Glyph- - - -12-120-75-75-P-113-SunOLGlyph-1
OLGL14	-Sun-Open Look Glyph- - - -14-140-75-75-P-128-SunOLGlyph-1
OLGL19	-Sun-Open Look Glyph- - - -19-190-75-75-P-154-SunOLGlyph-1
VT330	
VT33018	-DEC-VT330-Medium-R-Normal- -20-180-75-75-C-100-ISO8859-1
VT33036	-DEC-VT330-Medium-R-Normal- -40-360-75-75-C-200-ISO8859-1
VT330_BOLD18	-DEC-VT330-Bold-R-Normal- -20-180-75-75-C-100-ISO8859-1
VT330_BOLD36	-DEC-VT330-Bold-R-Normal- -40-360-75-75-C-200-ISO8859-1
VT330_BOLD_DBLWIDE18	-DEC-VT330-Bold-R-Double Wide- -20-180-75-75-C-200-ISO8859-1
VT330_BOLD_DBLWIDE_DECTECH18	-DEC-VT330-Bold-R-Double Wide- -20-180-75-75-C-200-DEC-DECTech
VT330_BOLD_DECTECH18	-DEC-VT330-Bold-R-Normal- -20-180-75-75-C-100-DEC-DECTech
VT330_BOLD_DECTECH36	-DEC-VT330-Bold-R-Normal- -40-360-75-75-C-200-DEC-DECTech
VT330_BOLD_NARROW18	-DEC-VT330-Bold-R-Narrow- -20-180-75-75-C-60-ISO8859-1
VT330_BOLD_NARROW36	-DEC-VT330-Bold-R-Narrow- -40-360-75-75-C-120-ISO8859-1

(continued on next page)

Table 4–11 (Cont.) Additional Common Fonts (.PCF File Extension)

File Name	Font Name
VT330	
VT330_BOLD_NARROW_DECTECH18	-DEC-VT330-Bold-R-Narrow- -20-180-75-75-C-60-DEC-DECTech
VT330_BOLD_NARROW_DECTECH36	-DEC-VT330-Bold-R-Narrow- -40-360-75-75-C-120-DEC-DECTech
VT330_BOLD_WIDE18	-DEC-VT330-Bold-R-Wide- -20-180-75-75-C-120-ISO8859-1
VT330_BOLD_WIDE_DECTECH18	-DEC-VT330-Bold-R-Wide- -20-180-75-75-C-120-DEC-DECTech
VT330_DBLWIDE18	-DEC-VT330-Medium-R-Double Wide- -20-180-75-75-C-200-ISO8859-1
VT330_DBLWIDE_DECTECH18	-DEC-VT330-Medium-R-Double Wide- -20-180-75-75-C-200-DEC-DECTech
VT330_DECTECH18	-DEC-VT330-Medium-R-Normal- -20-180-75-75-C-100-DEC-DECTech
VT330_DECTECH36	-DEC-VT330-Medium-R-Normal- -40-360-75-75-C-200-DEC-DECTech
VT330_NARROW18	-DEC-VT330-Medium-R-Narrow- -20-180-75-75-C-60-ISO8859-1
VT330_NARROW36	-DEC-VT330-Medium-R-Narrow- -40-360-75-75-C-120-ISO8859-1
VT330_NARROW_DECTECH18	-DEC-VT330-Medium-R-Narrow- -20-180-75-75-C-60-DEC-DECTech
VT330_NARROW_DECTECH36	-DEC-VT330-Medium-R-Narrow- -40-360-75-75-C-120-DEC-DECTech
VT330_WIDE18	-DEC-VT330-Medium-R-Wide- -20-180-75-75-C-120-ISO8859-1
VT330_WIDE_DECTECH18	-DEC-VT330-Medium-R-Wide- -20-180-75-75-C-120-DEC-DECTech
Language-Specific Fonts	
HANGLG16	-Daewoo-Gothic-Medium-R-Normal- -16-120-100-100-C-160-KSC5601.1987-0
HANGLM16	-Daewoo-Mincho-Medium-R-Normal- -16-120-100-100-C-160-KSC5601.1987-0
HANGLM24	-Daewoo-Mincho-Medium-R-Normal- -24-170-100-100-C-240-KSC5601.1987-0
HEB6X13	-Misc-Fixed-Medium-R-SemiCondensed- -13-120-75-75-C-60-ISO8859-8
HEB8X13	-Misc-Fixed-Medium-R-Normal- -13-120-75-75-C-80-ISO8859-8
JISKAN16	-JIS-Fixed-Medium-R-Normal- -16-150-75-75-C-160-JISX0208.1983-0
JISKAN24	-JIS-Fixed-Medium-R-Normal- -24-230-75-75-C-240-JISX0208.1983-0
K14	-Misc-Fixed-Medium-R-Normal- -14-130-75-75-C-140-JISX0208.1983-0
Miscellaneous Fonts	
NIL2	-Misc-Nil-Medium-R-Normal- -2-20-75-75-C-10-Misc-FontSpecific

Programming Features

4.6 X Window System Library (Xlib)

4.6.7.4 Bitstream Speedo Scalable Fonts

Table 4–12 lists the previously undocumented Bitstream Speedo scalable fonts and their file names. The files for these fonts are located in the following directory:

DECW\$SYSCOMMON:[SYSFONT.DECW.SPEEDO]

Table 4–12 Bitstream Speedo Scalable Fonts (.SPD File Extension)

File Name	Font Name
Charter	
FONT0648	-Bitstream-Charter-Medium-R-Normal- -0-0-0-0-P-0-ISO8859-1
FONT0649	-Bitstream-Charter-Medium-I-Normal- -0-0-0-0-P-0-ISO8859-1
FONT0709	-Bitstream-Charter-Bold-R-Normal- -0-0-0-0-P-0-ISO8859-1
FONT0710	-Bitstream-Charter-Bold-I-Normal- -0-0-0-0-P-0-ISO8859-1
Courier	
FONT0419	-Bitstream-Courier-Medium-R-Normal- -0-0-0-0-m-0-ISO8859-1
FONT0582	-Bitstream-Courier-Medium-I-Normal- -0-0-0-0-m-0-ISO8859-1
FONT0583	-Bitstream-Courier-Bold-R-Normal- -0-0-0-0-m-0-ISO8859-1
FONT0611	-Bitstream-Courier-Bold-I-Normal- -0-0-0-0-m-0-ISO8859-1

4.6.7.5 Agfa Monotype TrueType Scalable Fonts

Table 4–13 lists the new Agfa Monotype TrueType scalable fonts and their file names. The files for these fonts are located in the following directory:

DECW\$SYSCOMMON:[SYSFONT.DECW.TRUETYPE]

Table 4–13 Agfa Monotype TrueType Scalable Fonts (.TTF File Extension)

File Name	Font Name
Albany (Similar to Arial)	
ALBANYBD	-Agfa Monotype-Albany-Bold-R-Normal- -0-0-0-0-P-0-ISO8859-1
ALBANYBI	-Agfa Monotype-Albany-Bold-I-Normal- -0-0-0-0-P-0-ISO8859-1
ALBANYIT	-Agfa Monotype-Albany-Medium-I-Normal- -0-0-0-0-P-0-ISO8859-1
ALBANY_ _	-Agfa Monotype-Albany-Medium-R-Normal- -0-0-0-0-P-0-ISO8859-1
Cumberland (Similar to Courier)	
CUMBB	-Agfa Monotype-Cumberland-Bold-R-Normal- -0-0-0-0-M-0-ISO8859-1
CUMBBI	-Agfa Monotype-Cumberland-Bold-I-Normal- -0-0-0-0-M-0-ISO8859-1
CUMBI	-Agfa Monotype-Cumberland-Medium-I-Normal- -0-0-0-0-M-0-ISO8859-1
CUMBR	-Agfa Monotype-Cumberland-Medium-R-Normal- -0-0-0-0-M-0-ISO8859-1

(continued on next page)

Table 4–13 (Cont.) Agfa Monotype TrueType Scalable Fonts (.TTF File Extension)

File Name	Font Name
Screen	
SAN_M_21	-Agfa Monotype-Screen Sans-Medium-R-Normal- -0-0-0-0-M-0-ISO8859-1
SAN_P_21	-Agfa Monotype-Screen Sans-Medium-R-Normal- -0-0-0-0-P-0-ISO8859-1
SRF_M_21	-Agfa Monotype-Screen Serif-Medium-R-Normal- -0-0-0-0-M-0-ISO8859-1
SRF_P_21	-Agfa Monotype-Screen Serif-Medium-R-Normal- -0-0-0-0-P-0-ISO8859-1
Thorndale (Similar to Times New Roman)	
THOBI_ _ _	-Agfa Monotype-Thorndale-Bold-I-Normal- -0-0-0-0-P-0-ISO8859-1
THOB_ _ _ _	-Agfa Monotype-Thorndale-Bold-R-Normal- -0-0-0-0-P-0-ISO8859-1
THOI_ _ _ _	-Agfa Monotype-Thorndale-Medium-I-Normal- -0-0-0-0-P-0-ISO8859-1
THOR_ _ _ _	-Agfa Monotype-Thorndale-Medium-R-Normal- -0-0-0-0-P-0-ISO8859-1

4.6.7.6 Adobe Type1 Fonts

Table 4–14 lists previously undocumented Adobe Type1 fonts and their file names. The files for these fonts are located in the following directory:

DECW\$SYSCOMMON:[SYSFONT.DECW.TYPE1]

Table 4–14 Adobe Type1 Scalable Fonts (.PCA File Extension)

File Name	Font Name
Courier	
COUR	-Adobe-Courier-Medium-R-Normal- -0-0-0-0-P-0-ISO8859-1
COURI	-Adobe-Courier-Medium-I-Normal- -0-0-0-0-P-0-ISO8859-1
COURB	-Adobe-Courier-Bold-R-Normal- -0-0-0-0-P-0-ISO8859-1
COURBI	-Adobe-Courier-Bold-I-Normal- -0-0-0-0-P-0-ISO8859-1
Utopia	
UTRG_ _ _ _	-Adobe-Utopia-Medium-R-Normal- -0-0-0-0-P-0-ISO8859-1
UTI_ _ _ _ _	-Adobe-Utopia-Medium-I-Normal- -0-0-0-0-P-0-ISO8859-1
UTB_ _ _ _ _	-Adobe-Utopia-Bold-R-Normal- -0-0-0-0-P-0-ISO8859-1
UTBI_ _ _ _	-Adobe-Utopia-Bold-I-Normal- -0-0-0-0-P-0-ISO8859-1

4.6.8 UIDPATH Environment Variable

V1.2–6

When opening a hierarchy, DECwindows Motif searches the DECW\$USER_DEFAULTS and DECW\$SYSTEM_DEFAULTS areas for the User Interface Definition (UID) file. On UNIX systems, the search path is defined using the UIDPATH variable and its fallbacks.

Now DECwindows Motif also checks for the UIDPATH variable if the UID file is not found using either of the OpenVMS symbols listed above. This variable references a UNIX-style pathname (for example, /foo/bar) and allows

Programming Features

4.6 X Window System Library (Xlib)

the substitutions strings as specified by X11 standards. For more information on the UIDPATH variable, see the *OSF/Motif Programmer's Reference*.

Note

The UIDPATH variable does not work with OpenVMS directory specifications. Use the DECW\$xxx_DEFAULTS logicals to specify OpenVMS-style search paths.

4.6.9 Client Side Extension Library

V1.1

Starting with DECwindows Motif for OpenVMS Version 1.1, Xlib added a client side library, DECW\$XEXTLIBSHR.EXE, that allows OpenVMS clients to issue Shape, XInput, Multibuffer, and Shared Memory extension requests to servers that provide these features.

You must modify the linking file options for client applications that issue these extension requests to link to the Xlib extensions shareable image in SYS\$LIBRARY:DECW\$XEXTLIBSHR.EXE. Add the following line to your linker options file:

```
SYS$LIBRARY:DECW$XEXTLIBSHR/SHARE
```

For more information on Shape, XInput, and Multibuffer extensions, see the following text files in SYS\$HELP:

```
DECW$SHAPE.TXT  
DECW$XINPUT.TXT  
DECW$MULTIBUFFER.TXT
```

4.7 X Window System Extensions and Protocols

The following sections describe features related to X Window System extensions, protocols, and their libraries.

4.7.1 Additional X Display Server Extensions Supported with X11R6.6

V1.3

The following X11R6.6 protocol X Window System extensions have been integrated into the DECwindows X11 Display Server and are now supported by DECwindows Motif:

- Application Group (XC-APPGROUP)
- Big Requests (BIG-REQUESTS)
- Colormap Utilization Policy (TOG-CUP)
- Extended Visual Information (EVI)
- Low-Bandwidth X (LBX)
- Security (SECURITY)
- Synchronization (SYNC)
- XC-MISC
- X Double Buffer (DBE)

Programming Features

4.7 X Window System Extensions and Protocols

- XINERAMA (formerly Panoramix)
- X Keyboard (XKB)

BIG-REQUESTS, EVI, SYNC, TOG-CUP, and XC-MISC are a permanent part of the DECwindows X11 Display Server and are always enabled. DBE, LBX, SECURITY, XC-APPGROUP, XINERAMA, and XKB are dynamically loadable using the DECW\$SERVER_EXTENSIONS parameter defined in the DECW\$PRIVATE_SERVER_SETUP.COM file. With this symbol, each extension is converted to a shareable image, which is run at server startup. Note that some combinations of extensions can result in conflict; see *HP DECwindows Motif for OpenVMS Management Guide* for instructions on enabling these extensions.

To access these extensions, link applications against one or more of the following libraries:

Extension	Library
DBE EVI LBX SECURITY SYNC TOG-CUP XC-APPGROUP XINERAMA	DECW\$XEXTLIBSHR
BIG-REQUESTS XC-MISC XKB	DECW\$XLIBSHR

For more information about the updates made to the client-side extension library in support of X11R6.6, see Section 4.6.3.

The following sections briefly describe each extension, its function, and any variances from the standard X11R6.6 implementation provided by the X.Org Foundation. For a detailed description of each extension protocol and the available server requests, see the *X Window System* (Scheifler and Gettys) series of manuals described in the *HP DECwindows Motif for OpenVMS Documentation Overview*, or visit the X.Org Foundation web site (<http://www.x.org>) for protocol and library specifications.

4.7.1.1 Application Group Extension (XC-APPGROUP)

XC-APPGROUP enables multiple programs to manage X Window applications on the desktop. This extension allows X applications to be embedded into the window of another program, such as a web browser.

Sets of one or more applications, known as an Application Group, are managed by a controlling application, known as the Application Group Leader. The group shares the *Substructure-Redirect* attribute of the window with the Application Manager and one or more Application Group Leaders.

Code that uses XC-APPGROUP must include the following header files:

```
# include "DECW$INCLUDE:Xlib.h"
# include "DECW$INCLUDE:Xag.h"
```

Programming Features

4.7 X Window System Extensions and Protocols

This extension is dynamically loadable (along with the SECURITY extension) at server startup; see the *HP DECwindows Motif for OpenVMS Management Guide*. Call the following routine to check if XC-APPGROUP is available on the server system:

```
Bool XagQueryVersion (
    Display *dpy,
    int     *major_version_return,
    int     *minor_version_return
);
```

The following table lists each argument and its description.

Argument	Description
<i>dpy</i>	An input parameter that contains the current display.
<i>major_version_return</i>	Major version number of the extension implementation. Returned by XagQueryVersion.
<i>minor_version_return</i>	Minor version number of the extension implementation. Returned by XagQueryVersion.

4.7.1.2 Big Requests Extension (BIG-REQUESTS)

BIG-REQUESTS enables a client application to extend the length field of a protocol request from 2^{18} bytes to a 32-bit value. This is useful for clients and other extensions that frequently transmit complex information to the display server.

The only callable function associated with this extension is XExtendedMaxRequestSize, which has been incorporated into Xlib. As such, it is always available when connected to an X Window system that offers this extension.

4.7.1.3 Colormap Utilization Policy Extension (TOG-CUP)

TOG-CUP provides the following colormap management capabilities to the display server:

- A mechanism for a special application (such as a colormap manager) to recognize special colormap requirements. For example, this extension enables an application to locate and initialize a default colormap.
- A policy that encourages colormap sharing and reduces colormap flashing on low-end 8-bit frame buffers.
- A behavior in the color allocation scheme that reduces colormap flashing when colormaps are not shared.

Specifically, the TOG-CUP protocol provides methods that query the server for a list of reserved colormap entries and initialize shareable colormap entries at specific locations. If the core protocol does not contain information about the returned pixel values, the AllocColor and AllocNamedColor requests look in the default colormap for a matching color. If a match is found and the corresponding cell in the private colormap is empty, the color is allocated to that cell in the private colormap rather than the first available location. This minimizes colormap flashing when the main window's default visual class is using a private colormap and is set to GrayScale, PseudoColor, or DirectColor.

Code that uses the TOG-CUP extension must include the following header files:

```
# include "DECW$INCLUDE:Xlib.h"
# include "DECW$INCLUDE:Xcup.h"
```

This extension is a fixed part of the display server and is always enabled. Call the following routine to check if the TOG-CUP extension is available on the server system:

```
Bool XcupQueryVersion (display, &major, &minor)
    Display *display,
    int      major,minor;
```

Note that client applications must call `XcupQueryVersion` before calling any other TOG-CUP function.

The following table lists each argument and its description.

Argument	Description
<i>display</i>	An input parameter that contains the current display.
<i>major</i>	Major version number of the extension implementation. Returned by <code>XcupQueryVersion</code> .
<i>minor</i>	Minor version number of the extension implementation. Returned by <code>XcupQueryVersion</code> .

4.7.1.4 Extended Visual Information Extension (EVI)

EVI enables a client to query the display server for additional information about core X visuals, such as colormap information and framebuffer levels. Note that this extension only provides support for client applications and not other X Window System extensions.

Code that uses EVI must include the following header files:

```
# include "DECW$INCLUDE:Xlib.h"
# include "DECW$INCLUDE:Xevi.h"
```

This extension is a fixed part of the display server and is always enabled. Call the following routine to check if EVI is available on the server system:

```
Bool XeviQueryExtension (
    Display *dpy
);
```

The following table lists each argument and its description.

Argument	Description
<i>dpy</i>	An input parameter that contains the current display.

4.7.1.5 Low-Bandwidth X Extension (LBX)

LBX is a network-transparent protocol for running X Window System applications over transport channels whose bandwidth and latency are significantly lower than that available in local area networks. LBX combines a variety of caching and reencoding techniques that reduce the volume of data sent over the network.

By using a proxy server as an intermediary between the client applications and the X server, low-bandwidth/high-latency communication is maintained between the proxy and X server. The proxy server reencodes and compresses requests, events, replies and errors, as well as the resulting data stream. Additionally,

Programming Features

4.7 X Window System Extensions and Protocols

the proxy can cache information from the server to provide low-latency replies to client applications.

A proxy can serve multiple client applications and does not prevent clients from connecting directly to the server. The proxy can combine calls from multiple client applications into a single data stream.

Use of LBX is transparent to clients. The only interface to LBX available to client code is a query to check the availability of LBX. Code that uses this query must include the following header files:

```
# include "DECW$INCLUDE:Xlib.h"
# include "DECW$INCLUDE:XLbx.h"
```

This extension is dynamically loadable at server startup; see the *HP DECwindows Motif for OpenVMS Management Guide*. Call the following routines to check if LBX has been loaded and is available on the server system:

```
Bool XLbxQueryVersion(
    Display *display,
    int     *major_version_return,
    int     *minor_version_return
);
```

The following table lists each argument and its description.

Argument	Description
<i>display</i>	An input parameter that contains the current display.
<i>major_version_return</i>	Major version number of the extension implementation. Returned by XLbxQueryVersion.
<i>minor_version_return</i>	Minor version number of the extension implementation. Returned by XLbxQueryVersion.

4.7.1.6 Security Extension (SECURITY)

SECURITY contains a new protocol that provides for enhanced X server security. This extension adds the concepts of **trusted** and **untrusted** client connections to the X Window System protocol. The trust status of a client is determined by the authorization method used during the startup of a connection. All clients using host- or user-based authorization are considered trusted. Clients using token-based authorization protocols may be either trusted or untrusted depending on the authorization data included in the connection request.

The requests in SECURITY permit a trusted client to create multiple authorization entries related to a single authorization protocol. Each entry is tagged with a trust status, which is then associated with any client using that authorization entry.

When a connection identifying an untrusted client is accepted, the client is restricted from performing certain operations that would steal or modify data that is held by the server for trusted clients. An untrusted client performing a disallowed operation will receive protocol errors.

When a client is untrusted, the server can also limit the extensions that are available to the client. Each X protocol extension is responsible for defining what operations are permitted to untrusted clients. By default, the entire extension is hidden to untrusted clients.

Programming Features

4.7 X Window System Extensions and Protocols

With DECwindows Motif, the following extensions (standard and non-standard) are defined as secure:

BIG-REQUESTS
LBX
XC-MISC

All other extensions are considered insecure. See the *HP DECwindows Motif for OpenVMS Management Guide* for more information on how to select an appropriate authentication method and specify trusted or untrusted connections.

Code that uses SECURITY must include the following header files:

```
# include "DECW$INCLUDE:Xlib.h"
# include "DECW$INCLUDE:security.h"
```

This extension is dynamically loadable (along with the XC-APPGROUP extension) at server startup; see *HP DECwindows Motif for OpenVMS Management Guide*. Call the following routine to check if SECURITY is available on the server system:

```
Bool XSecurityQueryExtension (
    Display *dpy,
    int     *major_version_return,
    int     *minor_version_return
);
```

The following table lists each argument and its description.

Argument	Description
<i>dpy</i>	An input parameter that contains the current display.
<i>major_version_return</i>	Major version number of the extension implementation. Returned by XSecurityQueryExtension.
<i>minor_version_return</i>	Minor version number of the extension implementation. Returned by XSecurityQueryExtension.

4.7.1.7 XC-MISC Extension

XC-MISC allows client applications to retrieve previously-used resource ID ranges from the X server. Xlib handles this function automatically. This extension is useful for long-running applications that use many resource IDs over their runtime life.

Since the XC-MISC functions are part of Xlib, they are a standard part of the client. As such, they are always available when connected to an X Window system that offers this extension.

4.7.1.8 X Double Buffer Extension (DBE)

DBE provides a way to display flicker-free animation on an X Window system and is intended as a replacement to the Multibuffering extension. Successive frames of an animation sequence are rendered into the back buffer while the previously rendered frame is displayed in the front buffer. When a new frame is ready, the back and front buffers swap roles, making the new frame visible. Only completely rendered frames are shown; these frames remain visible during the entire time it takes to display the new frame.

Code that uses DBE must include the following header files:

```
# include "DECW$INCLUDE:Xlib.h"
# include "DECW$INCLUDE:Xdbe.h"
```

Programming Features

4.7 X Window System Extensions and Protocols

This extension is dynamically loadable at server startup; see the *HP DECwindows Motif for OpenVMS Management Guide*. Call the following routine to check if the extension has been loaded and is available on the server system:

```
Bool XdbeQueryExtension (
    Display *dpy,
    int     *major_version_return,
    int     *minor_version_return
);
```

The following table lists each argument and its description.

Argument	Description
<i>dpy</i>	An input parameter that contains the current display.
<i>major_version_return</i>	Major version number of the extension implementation. Returned by XdbeQueryExtension.
<i>minor_version_return</i>	Minor version number of the extension implementation. Returned by XdbeQueryExtension.

4.7.1.9 XINERAMA Extension

XINERAMA (formerly known as Panoramix) enables a system configured with multiple video monitors (multiheaded system) to function as a single large screen. This extension allows application windows and cursor movement to span multiple screens and move from one screen to another.

The overall size of the composite screen equals the combined size of all screens. Monitor configurations can be easily modified by enabling this extension in conjunction with the associated screen symbols (such as DECW\$SERVER_SCREENs). See the *HP DECwindows Motif for OpenVMS Management Guide* for the complete list of logicals associated with this extension and for instructions on how to setup and configure a multiheaded system that uses XINERAMA.

Note

This extension is only supported in a homogeneous graphics environment, which consists of common display devices, visual classes, depths, resolutions, etc. In addition, there may be some restrictions if operating in 3D mode (such as with the OpenGL layered product). See *HP DECwindows Motif for OpenVMS Release Notes* for the current restrictions regarding this extension.

4.7.1.10 X Keyboard Extension (XKB)

XKB enhances the control and customization of the keyboard under the X Window System by providing the following:

- Support for the ISO9996 standard for keyboard layouts
- Compatibility with the core X keyboard handling
- Standard methods for handling keyboard LEDs and locking modifiers
- Support for keyboard geometry

Note that all AccessX extension features for people with physical impairments have been incorporated into XKB. These accessibility features include StickyKeys, SlowKeys, BounceKeys, MouseKeys, and ToggleKeys, as well as complete control over the autorepeat delay rate.

Programming Features

4.7 X Window System Extensions and Protocols

Code that uses XKB must minimally include the following header files:

```
# include "DECW$INCLUDE:Xlib.h"
# include "DECW$INCLUDE:XKBlib.h"
```

To modify keyboard geometry descriptions, the names and identifiers of the predefined bells, or X Keyboard map definitions, additionally include the following header files:

```
# include "DECW$INCLUDE:XKBgeom.h"
# include "DECW$INCLUDE:XKBbells.h"
# include "DECW$INCLUDE:XKM.h"
# include "DECW$INCLUDE:XKMformat.h"
# include "DECW$INCLUDE:XKBfile.h"
# include "DECW$INCLUDE:XKBrules.h"
# include "DECW$INCLUDE:XKBconfig.h"
```

This extension is dynamically loadable at server startup; see the *HP DECwindows Motif for OpenVMS Management Guide*. Call the following routine to check if the extension has been loaded and is available on the server system:

```
Bool XkbQueryExtension(
    Display *dpy,
    int *opcodeReturn,
    int *eventBaseReturn,
    int *errorBaseReturn,
    int *majorReturn,
    int *minorReturn
);
```

The following table lists each argument and its description.

Argument	Description
<i>dpy</i>	An input parameter that contains the current display.
<i>opcodeReturn</i>	Major operation code of the extension. Returned by XkbQueryExtension.
<i>eventBaseReturn</i>	Base event code of the extension. Returned by XkbQueryExtension.
<i>errorBaseReturn</i>	Base error code of the extension. Returned by XkbQueryExtension.
<i>majorReturn</i>	Major version number of the extension implementation. Returned by XkbQueryExtension.
<i>minorReturn</i>	Minor version number of the extension implementation. Returned by XkbQueryExtension.

4.7.1.11 X Synchronization Extension (SYNC)

SYNC provides primitive calls that synchronize requests from multiple clients on different hosts running different operating systems. This extension enables applications to make the best use of buffering resources within the client, server, and network and eliminates network errors that can occur when two systems are running a distributed application.

Multimedia applications can use this extension to synchronize streams of audio, video, and graphics data. For example, simple animation applications can be implemented without having to use round-trip requests.

Code that uses SYNC must include the following header files:

```
# include "DECW$INCLUDE:Xlib.h"
# include "DECW$INCLUDE:sync.h"
```

Programming Features

4.7 X Window System Extensions and Protocols

This extension is a fixed part of the display server and is always enabled. Call the following routines to check if the extension has been loaded and is available on the server system:

```
Bool XSyncQueryExtension(  
    Display *dpy,  
    int     *event_base_return,  
    int     *error_base_return  
);  
  
Status XSyncInitialize(  
    Display *dpy,  
    int     *major_version_return,  
    int     *minor_version_return  
);
```

The following table lists each argument and its description.

Argument	Description
<i>dpy</i>	An input parameter that contains the current display.
<i>event_base_return</i>	An output parameter that indicates the base event code for the extension.
<i>error_base_return</i>	An output parameter that indicates the base error code for the extension.
<i>major_version_return</i>	Major version number of the extension implementation. Returned by XSyncInitialize.
<i>minor_version_return</i>	Minor version number of the extension implementation. Returned by XSyncInitialize.

4.7.2 Server Extensions Updated for X11R6.6

V1.3

The following existing X Window System extensions have been updated:

- DEC XTrap (DEC-XTRAP) and X Test (XTEST) – Now disabled with the DECW\$SERVER_DISABLE_TEST parameter
- MIT Screen Saver (MIT-SCREEN-SAVER) – Updated to work with XINERAMA
- Multibuffering (MBE) – New XmbufClearBufferArea function
- MIT Miscellaneous (MIT-SUNDRY-NONSTANDARD)
- MIT Shared Memory (MIT-SHM)
- Non-Rectangular Window Shape (SHAPE)
- X Image Extension (XIE) – Supports V3.0 and not the adopted standard of V5.0

4.7.3 Inter-Client Exchange (ICE) Protocol Support

V1.3

The Inter-Client Exchange (ICE) Protocol provides support for direct X Window System client-to-client communication without using the X server. This means that DECwindows Motif client applications can use ICE rather than connect to the X server. The standard protocol provides the basic mechanisms for establishing and closing network transport connections, performing authentication, negotiating versions, and reporting errors. The protocols running

Programming Features

4.7 X Window System Extensions and Protocols

within an ICE connection are known as **subprotocols**, of which Session Manager (described in Section 4.7.4) is a member.

A new client-side library, DECW\$ICELIB, is provided. Code that uses ICE must include the following header file:

```
# include "DECW$INCLUDE:Icelib.h"
```

The following sections describe the implementation of ICE provided with DECwindows Motif, highlighting any variances from or restrictions posed by the standard implementation. For a detailed description of the ICE protocol and the available server requests, see the *X Window System* (Scheifler and Gettys) series of manuals described in the *HP DECwindows Motif for OpenVMS Documentation Overview*, or visit the X.Org Foundation web site (<http://www.x.org>) for protocol and library specifications.

4.7.3.1 Multithreading Considerations

The ICE library supports multithreading after IceInitThreads has been called. IceInitThreads must be the first call on the ICE library if multithreading is required. Programs that call IceInitThreads must have been linked against PTHREAD\$RTL.

The following sections further describe issues with using ICE functions in a multithreaded environment.

Lock Nesting

Locks held by IceLockConn and IceAppLockConn are recursive. The corresponding unlock routine must be called the same number of times as the lock routine.

Deleting IceConn Objects

IceConn objects can be deleted by:

- IceProcessMessages returning IceProcessMessagesConnectionClosed
- IceCloseConnection returning IceClosedNow

In these cases, the IceConn object is freed without validation even though locks may still be held. To avoid race conditions, ensure that the deleted IceConn object is not being used by another thread.

Non-Atomic Functions and Macros

The following subset of the ICE functions that prepare and read messages are not atomic and do not acquire locks:

```
IceGetHeader  
IceGetHeaderExtra  
IceSimpleMessage  
IceErrorHandler  
IceWriteData  
IceWriteData16  
IceWriteData32  
IceSendData  
IceWritePad  
IceReadSimpleMessage  
IceReadCompleteMessage  
IceDisposeCompleteMessage  
IceReadMessageHeader  
IceReadData  
IceReadData16
```

Programming Features

4.7 X Window System Extensions and Protocols

IceReadData32
IceReadPad

Any multithreaded application that uses one or more of these macros or functions must explicitly acquire a lock on the connection before creating a message, and release the lock after the message is prepared. For read operations, this action is not required since the ICE process callbacks automatically lock the connection.

For example, the following is sample code for creating a message:

```
IceAppLockConn (iceConn);
IceGetHeaderExtra (iceConn, _SmcOpcode, SM_RegisterClient,
                  SIZEOF (smRegisterClientMsg), WORD64COUNT (extra),
                  smRegisterClientMsg, pMsg, pData);

*((CARD32 *) pData) = len;
pData += 4;

memcpy (pData, previousId, len);
pData += (len + 3) & (~3);
IceAppUnLockConn (iceConn);
```

Since an ICE connection can be shared between protocols, every protocol must use these locks, even if the protocol can only be used by a single thread.

Opening Connections

DECwindows Motif restricts multithreaded applications from concurrently calling `IceOpenConnection` and `IceCloseConnection`. `IceOpenConnection` can accept concurrent calls to itself as long as `IceCloseConnection` is not called at the same time.

ICE can maintain two open connections for the same protocol by using a major opcode check to the `IceOpenConnection` call. Since a protocol is registered only after it calls `IceProtocolSetup`, a conflict can occur if two threads simultaneously establish ICE connections for the same protocol and request that the connection is not shared.

To prevent this conflict from occurring, code for opening an ICE connection with a major opcode check should follow a format similar to the following:

```
IceConn conn;
IceProtocolSetupStatus status;
while (1) {
    conn = IceOpenConnection (...);
    if (conn == 0) break;
    status = IceProtocolSetup (...)
    if (status != IceProtocolAlreadyActive) break;
    IceCloseConnection (conn);
    /* Try again as another thread set up the protocol on this connection */
}
```

4.7.3.2 Differences from the Standard Implementation

The following sections describe differences from and issues in the standard ICE implementation provided with X11R6.6.

Connection and Protocol Authentication

The implementation of ICE included with DECwindows Motif does not include any authentication mechanisms for ICE connections. All listen objects must use `IceSetHostBasedAuthProc` to register host-based authentication.

For protocol authentication, all authentication schemes provided when the protocol is registered are permitted. This differs from the standard ICE implementation, where only those schemes defined in the ICE Authority (IceAuth) file are allowed.

Object Name Changes

The sample implementation of ICE provided by the X.Org Foundation contained objects whose name differed from that described in the ICE specification. The following table lists those objects and specifies which name was used in the DECwindows Motif implementation of ICE.

Documented Object Name	Implemented Object Name
IceGetContext	IceConnectionGetContext
major_opcode	majorOpcode
minor_opcode	minorOpcode

IceGetHeaderExtra Structure

The header structure used with IceGetHeaderExtra must have a sizeof value that is a multiple of 8 bytes.

4.7.3.3 IPv6 Considerations

V1.5

For ICE clients, the ICE library has been upgraded to support all the transport names described in the *HP DECwindows Motif for OpenVMS Management Guide*.

For ICE servers, specifying a value of TCPIP or TCP now opens listen sockets on all configured TCP/IP interfaces. If some of the interfaces are IPv6 interfaces, the library supports accepting connections on IPv6 interfaces.

After using the IceListenForConnections call, you can use the IceComposeNetworkIdList call to obtain the address for the TCP/IP listen object in the format "tcp/hostname", where *hostname* is the value of the system logical TCPIP\$INET_HOST. Use of the generic TCP transport allows the listen object to be used by both IPv4-based and IPv6-based peers.

The host-based authentication callback is provided the peer's host name as returned by the TCP/IP Services getnameinfo function. If the peer's host name is unavailable, the callback is provided with the peer's host address in either IPv4 or IPv6 format (depending on the protocol used to establish the connection). If the address is an IPv4-mapped IPv6 address, the address is converted to an IPv4 address before being provided to the callback. If the getnameinfo returns a name that includes a scope identifier, the scope identifier is included in the value passed to the callback.

4.7.4 X Session Management Protocol (XSMP) Support

V1.3

The X Session Management Protocol (XSMP) provides a standard way for users to save client sessions. Each session is controlled by a network service known as the **session manager**. The session manager issues commands that direct client applications to save their state information for use during subsequent sessions.

This protocol is built on top of ICE, which manages the client connections to the session manager server.

Programming Features

4.7 X Window System Extensions and Protocols

Code that uses XSMP must include the following header files:

```
# include "DECW$INCLUDE:SM.h"
# include "DECW$INCLUDE:Smlib.h"
# include "DECW$INCLUDE:SMproto.h"
```

The following sections describe the implementation of XSMP provided with DECwindows Motif, highlighting any variances from or restrictions posed by the standard implementation. For a detailed description of the XSMP protocol and the available server requests, see the *X Window System* (Scheifler and Gettys) series of manuals described in the *HP DECwindows Motif for OpenVMS Documentation Overview*, or visit the X.Org Foundation web site (<http://www.x.org>) for protocol and library specifications.

4.7.4.1 Multithreading Considerations

The implementation of XSMP is thread safe, using locks on the underlying ICE connection as needed. All send message operations are thread cancellation points; all callback operations are made by locking the associated ICE connection.

When `SmcOpenConnection` is called, it opens an ICE connection and processes messages until the session manager registers the client. The open connection subsequently causes a series of ICE watch procedures to be called. Typically, these procedures add the connection to a list monitored for input. `IceProcessMessages` is called when input to the list arrives. The thread issuing the `IceProcessMessages` calls will be blocked if it tries to handle a new connection.

4.7.4.2 Differences from the Standard Implementation

The following sections describe differences from and issues in the standard XSMP implementation provided with X11R6.6.

SmcCloseConnection and SmsCleanUp

In the standard implementation, `SmcCloseConnection` disables shutdown negotiation for an ICE connection, which results in abrupt termination of the connection. This can prevent the session manager from receiving all `SmcCloseConnection` messages.

In the DECwindows Motif implementation, shutdown negotiation is enabled. `SmcCloseConnection` returns `SmcClosedASAP`, and the connection is closed only after the session manager calls `SmsCleanUp`.

Note also that the sample session manager code does not specify whether `SmsCleanUp` closes an ICE Connection. In the DECwindows Motif implementation, an `IceCloseConnection` call is issued.

POSIX Property Names and Data Type Definitions

The standard specification defines the data types and property names supported for POSIX. The DECwindows Motif implementation specifies the property names; however, the data types definitions are not provided, since they may vary based on the use of session manager in the OpenVMS Alpha and OpenVMS I64 environments.

SmsGenerateClientId and IPv6

On a host where the primary TCP/IP interface limited to IPv6, the ID generated by `SmsGenerateClientId` will follow the IPv6 format (as defined in the X11R6.7 specification published by the X.Org Foundation).

4.7.5 MIT Shared Memory Extension (MIT-SHM) Support

V1.2

Shared memory extension support provides the capability to share memory XImages. This is a version of the XImage interface where the actual image data is stored in a shared-memory segment. Consequently, the image does not need to be moved through the Xlib interprocess communication channel. For large images, use of this extension can result in dramatic performance increases.

Support for shared memory pixmaps is also provided. Shared memory pixmaps are two-dimensional arrays of pixels in a format specified by the X server, where the image data is stored in the shared memory segment. Through the use of shared memory pixmaps, you can change the contents of these pixmaps without using any Xlib routines.

These routines are included in the client side extension library. See Section 4.6.9 for details on linking this library.

4.7.5.1 How to Use Shared Memory Extension

Code that uses the shared memory extension must include the following header files:

```
# include "DECW$INCLUDE:Xlib.h"
# include "DECW$INCLUDE:shm.h"
# include "DECW$INCLUDE:XShm.h"
```

Any code that uses the shared memory extension should first check that the server provides the extension. In some cases, such as running over the network, the extension does not work.

To check if the shared memory extension is available on your system, call one of the following routines:

```
Status XShmQueryExtension (display)
    Display *display

Status XShmQueryVersion (display, major, minor, pixmaps)
    Display *display;
    int *major, *minor;
    Bool *pixmaps
```

The following table lists each argument and its description.

Argument	Description
<i>display</i>	The current display. If the shared memory extension is used, the return value from either function is True. Otherwise, your program operates using conventional Xlib calls.
<i>major</i>	Major version number of the extension implementation. Returned by XShmQueryVersion.
<i>minor</i>	Minor version number of the extension implementation. Returned by XShmQueryVersion.
<i>pixmaps</i>	True, indicates that shared memory pixmaps are supported.

Programming Features

4.7 X Window System Extensions and Protocols

4.7.5.2 Using Shared Memory XImages

The following sequence shows the process for creating and using shared memory XImages:

1. Create the shared memory XImage structure.
2. Create a shared memory segment to store the image data.
3. Attach the shared memory segment.
4. Inform the server about the shared memory segment.
5. Use the shared memory XImage.

The following sections explain each step in this process:

Step 1—Creating a Shared Memory XImage Structure

To create a shared memory XImage, use the `XShmCreateImage` routine, which has the following format:

```
XImage *XShmCreateImage (display, visual, depth, format, data,
                        shminfo, width, height)
    Display *display;
    Visual *visual;
    unsigned int depth, width, height;
    int format;
    char *data;
    XShmSegmentInfo *shminfo;
```

Most of the arguments are the same as for `XCreateImage` (See the *X Window System* for a description of the `XCreateImage` routine.) Note that there are no *offset*, *bitmap_pad*, or *bytes_per_line* arguments. These quantities are set by the server, and your code needs to abide by them. Unless you have already allocated the shared memory segment (see step 2), you pass in `NULL` for the *data* pointer.

The argument *shminfo* is a pointer to a structure of type `XShmSegmentInfo`. Allocate one of these structures so that it has a lifetime at least as long as that of the shared memory XImage. There is no need to initialize this structure before the call to `XShmCreateImage`.

If successful, an XImage structure is returned, which you can use for the subsequent steps.

Step 2—Creating the Shared Memory Segment

Create the shared memory segment after the creation of the XImage because the XImage returns information that indicates how much memory to allocate.

The following example illustrates how to create the segment:

```
shminfo.shmid = shmget (IPC_PRIVATE,
                        image->bytes_per_line * image->height, IPC_CREAT|0777);
```

This example assumes that you called your shared memory XImage structure. A return value of 0 indicates the shared memory allocation has failed. Use the *bytes_per_line* field, not the *width* you used to create the XImage, as they may be different.

Note that the shared memory ID returned by the system is stored in the *shminfo* structure. The server needs that ID to attach itself to the segment.

Step 3—Attaching the Shared Memory Segment

Attach the shared memory segment to your process as in the following example:

```
shminfo.shmaddr = image->data = shmat (shminfo.shmid, 0, 0);
```

The address returned by `shmat` is stored in *both* the `XImage` structure and the `shminfo` structure.

To finish supplying arguments in the `shminfo` structure, decide how you want the server to attach to the shared memory segment, and set the `shminfo.readOnly` field as follows:

```
shminfo.readOnly = False;
```

If you set the structure to `True`, the server cannot write to this segment, and `XShmGetImage` calls fail.

Note

The shared memory segment routines are provided with DECwindows Motif. Using global sections, these routines emulate the shared memory routines on UNIX systems.

Step 4—Informing the Server About the Shared Memory Segment

Tell the server to attach to your shared memory segment as in the following example:

```
Status XShmAttach (display, shminfo);
```

If successful, a nonzero status is returned, and your `XImage` is ready for use.

Step 5—Using the Shared Memory XImage

To write a shared memory `XImage` into an X drawable, use the `XShmPutImage` routine. The `XShmPutImage` routine uses the following format:

```
XShmPutImage (display, d, gc, image, src_x, src_y,
               dest_x, dest_y, width, height, send_event)
    Display *display;
    Drawable d;
    GC gc;
    XImage *image;
    int src_x, src_y, dest_x, dest_y;
    unsigned int width, height;
    Bool send_event;
```

The interface is identical to the `XPutImage` routine (see the *X Window System*), except for one additional parameter, *send_event*. If this parameter is passed as `True`, the server generates a completion event when the image write is complete. This allows your program to know when it is safe to begin manipulating the shared memory segment again.

Programming Features

4.7 X Window System Extensions and Protocols

The completion event is of the type `XShmCompletionEvent`, which is defined as follows:

```
typedef struct {
    inttype; /* of event */
    unsigned long serial; /* # of last request processed */
    Bool send_event; /* true if came from a SendEvent request */
    Display *display; /* Display the event was read from */
    Drawable drawable; /* drawable of request */
    int major_code; /* ShmReqCode */
    int minor_code; /* X_ShmPutImage */
    ShmSeg shmseg; /* the ShmSeg used in the request */
    unsigned long offset; /* the offset into ShmSeg used */
} XShmCompletionEvent;
```

To determine the event type value that is used at run time, use the `XShmGetEventBase` routine as in the following example:

```
int CompletionType = XShmGetEventBase (display) + ShmCompletion;
```

Note

If you modify the shared memory segment before the arrival of the completion event, the results may be inconsistent.

To read image data into a shared memory `XImage`, use the `XShmGetImage` routine, which uses the following format:

```
Status XShmGetImage (display, d, image, x, y, plane_mask)
    Display *display;
    Drawable d;
    XImage *image;
    int x, y;
    unsigned long plane_mask;
```

The following table lists each argument and its description.

Argument	Description
<i>display</i>	The display of interest.
<i>d</i>	The source drawable.
<i>image</i>	The destination <code>XImage</code> .
<i>x</i>	X-offset within the source drawable.
<i>y</i>	Y-offset within the source drawable.
<i>plane_mask</i>	The planes that are to be read.

To destroy a shared memory `XImage`, first instruct the server to detach from it, then destroy the segment itself. The following example illustrates how to destroy a shared memory `XImage`:

```
XShmDetach (display, shminfo);
XDestroyImage (image);
shmdt (shminfo.shmaddr);
shmctl (shminfo.shmid, IPC_RMID, 0);
```

4.7.5.3 Using Shared Memory Pixmap

Unlike X images, for which any image format is usable, the shared memory extension supports only a single format for the data stored in a shared memory pixmap (XYPixmap or ZPixmap). This format is independent of the depth of the image and independent of the screen. (For 1-bit pixmaps the format is irrelevant.)

The XShmPixmapFormat routine returns the shared memory pixmap format for the server. The XShmPixmapFormat routine has the following format:

```
int XShmPixmapFormat (display)
    Display *display;
```

Your application can only use shared memory pixmaps in the format returned by the XShmPixmapFormat routine (including bits-per-pixel). To create a shared memory pixmap do the following:

- Create a shared memory segment and shminfo structure exactly the same way as is listed for shared memory XImages steps 1 through 4 (see Section 4.7.5.2). While it is not necessary to create an XImage first (step 1), doing so incurs little overhead and provides an appropriate bytes_per_line value to use.
- Call the XShmCreatePixmap routine, which has the following format:

```
Pixmap XShmCreatePixmap (display, d, data, shminfo, width,
                        height, depth);
    Display *display;
    Drawable d;
    char *data;
    XShmSegmentInfo *shminfo;
    unsigned int width, height, depth;
```

The arguments are the same as for XCreatePixmap (see the *X Window System*) except for two additional parameters, *data* and *shminfo*. The *data* parameter is the pointer to the shared memory segment and is the same as the *shminfo.shmaddr* field. The *shminfo* parameter is the same as the previous structure.

If successful, a pixmap is returned, which you can manipulate. You can manipulate its contents directly through the shared memory segment. Shared memory pixmaps are destroyed with the XFreePixmap routine, although you should detach and destroy the shared memory segment (see step 4 in Section 4.7.5.2).

4.7.6 X Image Extension (XIE) Support

V1.1

Starting with DECwindows Motif for OpenVMS Version 1.1, DECwindows Motif supports the X Image Extension (XIE). XIE allows image display processing using resources on the server side of the X client-server model. XIE eliminates the need to transmit image data repeatedly from the client to the server and also allows data to be transmitted in compressed form, reducing the network load.

DECwindows Motif includes the XIE client side shareable library (XIE\$SHRLIB.EXE) and C language header files. These allow applications to communicate with any X11 server that supports the XIE extension.

Programming Features

4.7 X Window System Extensions and Protocols

An XIE program uses a structure called the XIEImage to describe image data on the client side. This general mechanism describes data that the destination server is incapable of processing. Consult the documentation for the server system for information on what data types and sizes are supported. Unless the documentation specifies different limits, the server is capable of processing unsigned byte (UdpK_DTypeBU), unaligned bit field (UdpK_DTypeVU), and aligned bit field (UdpK_DTypeV) data, with a maximum depth of 8 bits per pixel per component. The XIE client library supports these data types, as well as unsigned word (UdpK_DTypeWU), and a depth of up to 16 bits per pixel per component.

Although the XIE protocol and programming interface have been standardized for X11R6, DECwindows Motif has not yet migrated to the latest implementation of this protocol.

4.8 Transport Programming

The following sections contain features related to DECwindows Motif transport interfaces.

4.8.1 Support for the LAT Transport Interface Available

V1.3–1

Support for the DECwindows Motif interface to the LAT transport, which was withdrawn with DECwindows Motif Version 1.3, has been restored and is available on the OpenVMS Alpha and OpenVMS I64 platforms. This support enables users to start LAT X sessions and communicate over low-capacity networks with systems running DECwindows Motif Version 1.3–1 or higher. It also allows client applications running on these DECwindows Motif systems to use the LAT transport to connect to X terminal systems.

Note that the restored LAT interface included with the OpenVMS operating system can be used as a valid network transport for communication with the DECwindows Motif Version 1.3–1 (or greater) and OpenVMS Version 7.3–2 (or greater) display servers. However, use with any other communication protocols in the X11R6.6 environment is not supported. This includes communication by or with the following:

- Inter-Client Exchange (ICE) and Session Manager protocols
- Low-Bandwidth X (LBX) proxy servers
- Proxy manager applications
- Font servers

Additionally, HP does not support the use of a token-based authentication protocol (such as MIT-MAGIC-COOKIE-1 or MIT-KERBEROS-5) with the restored LAT transport interface.

4.8.2 Support for the Logical Connection Number (LCN) Interface

V1.3

DECwindows Motif now includes an interface for determining when an I/O channel is ready and available for use. The logical connection number (LCN) interface is now used to signal when DECwindows Motif I/O channels are available, including those for Inter-Client Exchange (ICE), local and remote X server, and for Input Method Server connections.

Previously, DECwindows Motif used an OpenVMS event flag number (EFN) to signal when input was received from the X server. However, EFNs cannot be used safely in a multithreaded environment. The LCN interface allows multiple threads to handle the same, or different, connections without any thrashing or unnecessary delays.

The following sections further describe the functions of the LCN interface and provide detailed information about the supported routines.

4.8.2.1 LCN Functions

The principal function of the LCN interface is to test the readiness of an I/O channel. The design of the interface is based on the UNIX select function, which tests the state of UNIX file descriptors and returns when one of them is ready or a timeout occurs.

On OpenVMS, the LCN routines perform the following operations:

- Initialize support for multithreading
- Allocate a connection number
- Query the status of a connection number
- Signal when input is available

4.8.2.1.1 Initializing Thread Support LCN routines can execute in a single-threaded environment using EFNs to signal input or in a multithreaded environment using POSIX Threads routines. Multithreading is enabled with the DECW\$LCN_THREAD_INIT routine.

With multithreading enabled, the select routines (DECW\$LCN_SELECT_ONE and DECW\$LCN_SELECT) can be called concurrently from multiple kernel threads in user mode and one kernel thread in exec mode. Calls from user mode ASTs are not allowed.

With single threading, the select routines can be called from user mode and exec mode ASTs. Note, however, that the only concurrent calls allowed are one call from user mode followed by one call from an AST in user mode.

4.8.2.1.2 Allocating Connection Numbers LCNs are allocated to a connection using the DECW\$LCN_ALLOCATE routine. Values for LCNs start at 64 to distinguish them from local event flags. The maximum number of concurrently allocated LCNs equals the open file limit of the process. If the quota is 0, a default value of 1023 is used. If the quota exceeds the maximum value, a value of 2047 is used.

Once an LCN is allocated, it is unavailable for reuse until freed by the DECW\$LCN_FREE routine.

4.8.2.1.3 Querying Status and Signaling Input Each LCN has three status flags, which signify whether an LCN is ready and has received input from a particular operation. Each flag can be either set (1) using the DECW\$LCN_SET_x_READY routine or cleared (0) using the DECW\$LCN_CLEAR_x_READY routine.

Programming Features

4.8 Transport Programming

Input is signaled by setting the appropriate ready flag. The following table lists each LCN flag and describes when it is typically set and cleared.

Flag	Description
read ready	Set when there is data available to read.
write ready	Set when there is space in internal buffers to which data can be written.
except ready	Set when there is high-priority (exceptional) input.

Each flag can be set individually, and a select operation can test any combination of them. There are two routines that essentially mirror the UNIX select function, and test the ready state of an LCN. DECW\$LCN_SELECT selects and tests the status of a range of LCNs or EFNs. DECW\$LCN_SELECT_ONE performs the same function, however only tests the status of a single LCN.

4.8.3 LCN Routines

This section describes each of the LCN routines, which are available from the library image DECW\$LCNLIBSHR.EXE. To support use from protected images that cannot use the client library, some functions are also available as part of the X Transport system services (DECW\$XPORT_SERVICES).

4.8.3.1 DECW\$LCN_ALLOCATE

Assigns an LCN.

Format

DECW\$LCN_ALLOCATE lcn

Returns

type: longword (unsigned)
access: write
mechanism: by value

Returns a longword condition value in R0. Condition values returned by this routine are listed under Condition Values Returned.

Arguments

lcn

type: longword
access: write only
mechanism: by reference

The value of the allocated LCN.

Description

DECW\$LCN_ALLOCATE assigns a logical connection number. Initially, each allocated LCN is assigned with all ready flags (read, write, and except) in the clear state (0). The state of these flags can be changed using the DECW\$LCN_SET_x_READY routines.

Once allocated, the LCN cannot be reused until it is released by DECW\$LCN_FREE. DECW\$LCN_ALLOCATE must be called before any query, wait, or signaling operations can be performed.

DECW\$LCN_ALLOCATE is both thread- and AST-reentrant and is callable from exec and lower modes.

The equivalent function of DECW\$LCN_ALLOCATE is also available as a system service (DECW\$XPORT_LCN_ALLOCATE) from the set of transport-common routines (DECW\$XPORT_SERVICES.EXE).

Condition Values Returned

SS\$_NORMAL	Routine successfully completed.
DECW\$_INSFMEM	There is insufficient memory to perform the operation.
DECW\$_NOFREELCN	All LCNs are currently allocated.

4.8.3.2 DECW\$LCN_CLEAR_x_READY

Changes the ready bit for read, write, or except operations to the clear state.

Format

DECW\$LCN_CLEAR_READ_READY lcn [, prior]

DECW\$LCN_CLEAR_WRITE_READY lcn [, prior]

DECW\$LCN_CLEAR_EXCEPT_READY lcn [, prior]

Returns

type: longword (unsigned)
 access: write
 mechanism: by value

Returns a longword condition value in R0. Condition values returned by this routine are listed under Condition Values Returned.

Arguments

lcn

type: longword
 access: read only
 mechanism: by value

The value of the LCN whose ready bit for read, write, and except operations will be changed to the clear state (0).

[prior]

type: longword
 access: write only
 mechanism: by reference

The previous state of the associated ready flag, either clear (0) or set (1).

Description

DECW\$LCN_CLEAR_x_READY clears the read, write, or except ready bit of an LCN. This indicates that the LCN is not available for input from the specified operations.

These routines are thread- and AST-reentrant and callable from exec and lower modes.

The equivalent functions of DECW\$LCN_CLEAR_x_READY are also available as system services (DECW\$XPORT_LCN_CLEAR_x) from the set of transport-common routines (DECW\$XPORT_SERVICES.EXE). Note that when using the system service, the **prior** argument is required; use a 0 value to prevent the prior state from being returned.

Programming Features

4.8 Transport Programming

Condition Values Returned

SS\$_NORMAL	Routine successfully completed.
DECW\$_NOT_INITIALIZED	The LCN has not been initialized; DECW\$LCN_ALLOCATE must be called prior to this operation.
DECW\$_INVLCN	The LCN has not been allocated.

4.8.3.3 DECW\$LCN_FREE

Deassigns an allocated LCN.

Format

DECW\$LCN_FREE lcn

Returns

type: longword (unsigned)
access: write
mechanism: by value

Returns a longword condition value in R0. Condition values returned by this routine are listed under Condition Values Returned.

Arguments

lcn

type: longword
access: read only
mechanism: by value

The value of the LCN to be freed.

Description

DECW\$LCN_FREE deassigns the specified LCN. Once freed, the LCN is available for immediate reallocation.

This routine is thread- and AST-reentrant and callable from exec and lower modes.

The equivalent function of DECW\$LCN_FREE is also available as a system service (DECW\$XPORT_LCN_FREE) from the set of transport-common routines (DECW\$XPORT_SERVICES.EXE).

Note

If either DECW\$LCN_SELECT_ONE or DECW\$LCN_SELECT has been called to test a state of the LCN which has been freed, then the status DECW\$_INVLCN is returned from the select call.

Condition Values Returned

SS\$_NORMAL	Routine successfully completed.
SS\$_INSFMEM	There is insufficient memory to perform the operation.
DECW\$_NOT_INITIALIZED	The LCN has not been initialized; DECW\$LCN_ALLOCATE must be called prior to this operation.
DECW\$_INVLCN	The LCN has not been allocated or is protected.

4.8.3.4 DECW\$LCN_SELECT

Tests the ready state(s) of one or more LCNs and returns when one of the tested states is set, a timeout occurs, or a specified OpenVMS event flag is set.

Format

DECW\$LCN_SELECT retcount, rmask, wmask, emask, [timeout], [efn], [efn_mask]

Returns

type: longword (unsigned)
access: write
mechanism: by value

Returns a longword condition value in R0. Condition values returned by this routine are listed under Condition Values Returned.

Arguments

retcount

type: longword
access: write only
mechanism: by reference

The total number of entries set in the three mask structures (rmask, wmask, emask).

rmask, wmask, emask

type: mask
access: read,write
mechanism: by reference

Specifies whether to check the read (rmask), write (wmask), or exception (emask) status of one or more LCNs. A value indicates that the status check be performed; a null value indicates that no check be made. On completion, the mask is updated to reflect which LCNs have their ready state set.

The format of each mask is an array of word values. The first entry is the number of remaining entries in the array. Each subsequent entry represents an LCN value.

timeout

type: quadword
access: read only
mechanism: by reference

The time by which the select operation will timeout if no input is received. The time value is expressed in OpenVMS binary delta-time format. A null value indicates no timeout. A value of 0 indicates the operation is in polling mode and will timeout immediately if none of the specified status bits are set.

efn

type: longword
access: read only
mechanism: by value

An event flag number (EFN) in the cluster to which the **efn_mask** argument applies. EFNs are typically used for single-threaded or inner-mode operations. In this environment, **efn** identifies an event flag for the wait operation.

Programming Features

4.8 Transport Programming

If no EFN value is provided in single-thread mode, SYS\$HIBER and SYS\$WAKE are used. In these instances, SYS\$HIBER must not be used concurrently within the process. In particular, POSIX Threads must not be loaded into the image, even if not in use.

For multithreaded, user mode operations, this argument can be optional depending on whether an EFN has been provided previously to DECW\$LCN_THREAD_INIT. If the EFN was specified and the value of **efn_mask** is 0, the argument is optional. Otherwise the value of this argument is required and will be used as if it had been provided to DECW\$LCN_THREAD_INIT.

efn_mask

type: longword
access: read only
mechanism: by value

A mask of EFNs to be tested. Requires the **efn** argument.

Description

DECW\$LCN_SELECT waits until one of the specified LCN ready states has been set, timed out, or until the event flag condition is met. This routine checks whether the selected LCNs have been allocated and returns an error (DECW\$_INVLCN) if one or more LCNs have either not been allocated or freed for reuse.

With multithreading enabled, this routine is thread-reentrant and callable from exec or lower modes. Calls from ASTs are not supported.

With single threading, the select routines can be called from user mode and exec mode ASTs. Note, however, that the only concurrent calls allowed are one call from user mode followed by one call from an AST in user mode.

Condition Values Returned

SS\$_NORMAL	Routine successfully completed. One or more LCNs have their ready bit set as indicated in the updated mask values.
SS\$_EXQUOTA	A process quota has been exceeded, this can be due to the timer entry or AST limit quota.
SS\$_INSFMEM	There is insufficient memory to perform the operation.
SS\$_UNASEFC	The process is not associated with the cluster that contains the specified event flag.
DECW\$_BAD_EFN_CLUSTER	An event flag was not provided to DECW\$LCN_THREAD_INIT, or the specified flag resides in a different event flag cluster.
DECW\$_EFN_SET	One or more event flags in the mask have been set.
DECW\$_INVARG	The array count in a read, write, or exception mask, or the timeout value is not valid.
DECW\$_INVLCN	One or more LCNs have not been allocated or were freed during the operation.
DECW\$_NOHIBER	This call was made from inner-mode with multithreading enabled. No EFN was specified.
DECW\$_PTHREAD_INVALID	A POSIX Threads routine returned an unexpected error.
DECW\$_TIMEDOUT	The end of the timeout period was reached.

4.8.3.5 DECW\$LCN_SELECT_ONE

Tests the ready state of an LCN and returns when one of the tested states is set, a timeout occurs, or a specified OpenVMS event flag is set.

Format

DECW\$LCN_SELECT_ONE lcn, read, write, except, [timeout], [efn], [efn_mask]

Returns

type: longword (unsigned)
access: write
mechanism: by value

Returns a longword condition value in R0. Condition values returned by this routine are listed under Condition Values Returned.

Arguments

lcn

type: longword
access: read only
mechanism: by value

The value of the LCN.

read, write, except

type: longword
access: write only
mechanism: by reference

Specifies whether to check the read, write, or exception status of an LCN. A non-zero value signifies that the status check be performed and the result stored in the referenced address on completion. A zero or null value indicates that no check be made.

timeout

type: quadword
access: read only
mechanism: by reference

The time by which the select operation will timeout if no input is received. The time value is expressed in OpenVMS binary delta-time format. A null value indicates no timeout. A value of 0 indicates the operation is in polling mode and will timeout immediately if none of the specified status bits are set.

efn

type: longword
access: read only
mechanism: by value

An event flag number (EFN) in the cluster to which the **efn_mask** argument applies. EFNs are typically used for single-threaded or inner-mode operations. In this environment, **efn** identifies an event flag for the wait operation.

If no EFN value is provided in single-thread mode, SYS\$HIBER and SYS\$WAKE are used. In these instances, SYS\$HIBER must not be used concurrently within the process. In particular, POSIX Threads must not be loaded into the image, even if not in use.

Programming Features

4.8 Transport Programming

For multithreaded, user mode operations, this argument can be optional depending on whether an EFN has been provided previously to `DECW$LCN_THREAD_INIT`. If the EFN was specified and the value of **efn_mask** is 0, the argument is optional. Otherwise the value of this argument is required and will be used as if it had been provided to `DECW$LCN_THREAD_INIT`.

efn_mask

type: longword
access: read only
mechanism: by value

A mask of EFNs to be tested. Requires the **efn** argument.

Description

`DECW$LCN_SELECT_ONE` waits until one of the specified LCN ready states has been set, timed out, or until the event flag condition is met. This routine checks whether the selected LCN has been allocated and returns an error (`DECW$INVLCN`) if the LCN has not been allocated or freed for reuse.

With multithreading enabled, this routine is thread-reentrant and callable from exec or lower modes. Calls from ASTs are not supported.

With single threading, the select routines can be called from user mode and exec mode ASTs. Note, however, that the only concurrent calls allowed are one call from user mode followed by one call from an AST in user mode.

Condition Values Returned

<code>SS\$_NORMAL</code>	Routine successfully completed. One or more LCNs have their ready bit set as indicated in the updated mask values.
<code>SS\$_EXQUOTA</code>	A process quota has been exceeded, this can be due to the timer entry or AST limit quota.
<code>SS\$_INSFMEM</code>	There is insufficient memory to perform the operation.
<code>SS\$_UNASEFC</code>	The process is not associated with the cluster that contains the specified event flag.
<code>DECW\$_BAD_EFN_CLUSTER</code>	An event flag was not provided to <code>DECW\$LCN_THREAD_INIT</code> , or the specified flag resides in a different event flag cluster.
<code>DECW\$_INVARG</code>	The timeout period or EFN is not accessible or within the range of valid values.
<code>DECW\$_INVLCN</code>	The LCN has not been allocated or was freed during the operation.
<code>DECW\$_NOHIBER</code>	This call was made from inner-mode with multithreading enabled. No EFN was specified.
<code>DECW\$_PTHREAD_INVALID</code>	A POSIX Threads routine returned an unexpected error.
<code>DECW\$_TIMEDOUT</code>	The end of the timeout period was reached.

4.8.3.6 DECW\$LCN_SET_x_READY

Signals that input from a read, write, or exception operation has been received by changing the ready bit to the set state.

Format

DECW\$LCN_SET_READ_READY lcn

DECW\$LCN_SET_WRITE_READY lcn

DECW\$LCN_SET_EXCEPT_READY lcn

Returns

type: longword (unsigned)
access: write
mechanism: by value

Returns a longword condition value in R0. Condition values returned by this routine are listed under Condition Values Returned.

Arguments

lcn

type: longword
access: read only
mechanism: by value

The value of the LCN whose ready bit will be changed to the set state (1).

Description

DECW\$LCN_SET_x_READY sets the read, write, or except ready bit of an LCN. This signals input has been received from the specified operations.

These routines are thread- and AST-reentrant and callable from exec and lower modes. When any of the DECW\$LCN_SET_x_READY routines are called from exec mode with multithreading enabled, the call declares a user mode AST. This AST performs the signaling for any user-mode processes that are currently in a wait state.

The equivalent functions of DECW\$LCN_SET_x_READY are also available as system services (DECW\$XPORT_LCN_SET_x) from the set of transport-common routines (DECW\$XPORT_SERVICES.EXE).

Condition Values Returned

SS\$_NORMAL	Routine successfully completed.
SS\$_EXQUOTA	The current memory quota has been exceeded.
SS\$_INSFMEM	There is insufficient memory to perform the operation.
DECW\$_INVLCN	The LCN has not been allocated.
DECW\$_NOT_INITIALIZED	The LCN has not been initialized; DECW\$LCN_ALLOCATE must be called prior to this operation.

Programming Features

4.8 Transport Programming

4.8.3.7 DECW\$LCN_THREAD_INIT

Initializes multithreading support for LCN operations.

Format

DECW\$LCN_THREAD_INIT [efn]

Returns

type: longword (unsigned)
access: write
mechanism: by value

Returns a longword condition value in R0. Condition values returned by this routine are listed under Condition Values Returned.

Arguments

efn

type: longword
access: read only
mechanism: by value

The value of the specified EFN. When multithreading is enabled, an EFN value must be specified when performing select operations (DECW\$LCN_SELECT or DECW\$LCN_SELECT_ONE) with event flag masks. The EFN is supplied either by DECW\$LCN_THREAD_INIT or from the first select call that provides an **efn** argument.

The value of the **efn** argument must match the value supplied for any previous select operations or calls to DECW\$LCN_THREAD_INIT. The event flag must also be in the same event flag cluster as the efn value supplied to subsequent select or initialization operations.

Description

DECW\$LCN_THREAD_INIT enables multithreaded LCN operations. This routine is only callable from user mode with user mode ASTs enabled and can be called multiple times by a single process.

Condition Values Returned

SS\$_NORMAL	Routine successfully completed.
DECW\$_CHANGED_EFN	An event flag was specified that differs from that specified in a previous initialization or select call.
DECW\$_INSFMEM	There is insufficient memory to perform the operation.
DECW\$_NOPTHREADRTL	The POSIX Thread Library (PTHREAD\$RTL) has not been loaded.
DECW\$_NOTUSERMODE	This routine was not called from user mode.
DECW\$_PTHREAD_INVALID	A POSIX Thread routine returned an unexpected error.

A

accessx
 See AccessX Keyboard utility

AccessX features
 overview, 2–19
 setting with utility, 2–20

AccessX Keyboard utility (accessx), 2–19
 configuration file, 2–20
 resource settings, 2–21

ACME
 See Authentication and Credentials
 Management Extensions (ACME)

Agfa Monotype iType rasterizer, 4–22

Application Group extension (XC-APPGROUP), 4–37

Application Manager
 selecting screen for application display, 2–8

Asynchronous system traps (ASTs), 4–2

Authentication and Credentials Management Extensions (ACME), 3–1

B

Big Requests extension (BIG-REQUESTS), 4–38

C

CDA
 applications, 2–13
 dynamic font support, 2–12
 packing, 2–14
 unpacking, 2–14
 changes to external reference processing, 4–7
 converters, 2–11
 converting Asian-language text files, 2–11
 defining logical names, 2–10
 interface changes, 4–6
 internationalization support, 2–9
 message, 2–16
 packing and unpacking error messages, 2–15
 programming, 4–6
 restructuring of shareable images, 4–8
 specifying an options file, 2–10
 WRITE\$FONTS logical name, 2–12

Color Customizer, 2–3
 auto shadowing toggle button, 2–6
 building, 2–3
 command summary, 2–4
 mapping color resources and color cells, 2–5
 modifying DECW\$LOGIN.COM, 2–4
 running, 2–4
 supported
 applications, 2–3
 displays, 2–3
 using on multiheaded systems, 2–6
 xsetroot_cust demo, 2–7

Colormap Utilization Policy extension (TOG-CUP), 4–38

CompositeClassExtensionRec
 new option, 4–13

Compound Document Architecture
 See CDA

D

DBE
 See X Double Buffer extension

DECterm
 automatic window positioning, 2–17
 escape sequences, 2–17
 font sizes, 2–16
 page-movement sequences, 4–9
 programming, 4–8
 ReGIS input cursors, 4–8
 scrolling using the keyboard, 2–16

DECW\$IPV6_FONT_SUPPORT parameter, 3–2

DECW\$IPV6_SUPPORT global symbol, 2–1

DECW\$LCN_ALLOCATE routine, 4–55, 4–56

DECW\$LCN_CLEAR_x_READY routine, 4–55, 4–57

DECW\$LCN_FREE routine, 4–58

DECW\$LCN_SELECT routine, 4–59

DECW\$LCN_SELECT_ONE routine, 4–61

DECW\$LCN_SET_x_READY routine, 4–55, 4–63

DECW\$LCN_THREAD_INIT routine, 4–55, 4–64

DECW\$STARTUP procedure
 log file, 3–1

DECW\$XAUTHORITY logical, 2–26

DECwindows Extensions to Motif
 DXmCSText widget
 input method support, 4–10

Drag-and-drop, 2–7
DXmCSText widget
 input method support, 4–10

E

Easy resource configuration, 4–12
EFS
 See Extended File Specifications
Escape sequences
 DECterm, 2–17
EVI
 See Extended Visual Information extension
Extended File Specifications (EFS)
 File Manager support, 2–2
 file selection popup, 2–2
 FileView support, 2–2
 programming library support, 2–2
 Support overview, 2–2
 translated image support, 2–2
Extended Visual Information extension (EVI),
 4–39

F

File Manager
 refreshing views, 2–8
 selecting screen for application display, 2–8
Fonts
 Common
 Fixed Width, 4–32
 Language-Specific, 4–33
 Miscellaneous, 4–33
 Sun Open Look Glyph, 4–32
 VT330, 4–32
 75 dpi
 Charter, 4–23
 Lucida, 4–24
 Present Bullets, 4–26
 Utopia, 4–26
 100 dpi
 Charter, 4–27
 Lucida, 4–28
 Present Bullets, 4–31
 Utopia, 4–31
 Scalable
 Adobe Courier, 4–35
 Adobe Utopia, 4–35
 Agfa Monotype Albany, 4–34
 Agfa Monotype Cumberland, 4–34
 Agfa Monotype Screen, 4–35
 Agfa Monotype Thorndale, 4–35
 Bitstream Charter, 4–34
 Bitstream Courier, 4–34
Font server
 proxy server
 support with IPv6, 3–2

Front Panel
 selecting screen for application display, 2–8
 support for MB3 operations, 2–9

G

Global symbols
 DECW\$IPV6_SUPPORT, 2–1

H

Header files
 languages, 4–17
HP Extensions to Motif, 4–10

I

ICE
 See Inter-Client Exchange protocol
Input cursors
 cross-hair, 4–8
 diamond, 4–8
 rubber-band line, 4–8
 rubber-band rectangle, 4–8
 selecting, 4–8
Inter-Client Exchange protocol (ICE), 4–44
 client-side library, 4–45
 differences from the X11R6.6 implementation,
 4–46
 IPv6 considerations, 4–47
 multithreading considerations, 4–45
Internationalization
 converting files, 2–11
 viewing files, 2–9
Internet Protocol Version 6 transport
 See IPv6
IPv6, 2–1
 changes to ICE library, 4–47
 impact on xauth, 2–27
 support with font server, 3–2
 Xlib changes, 4–14

K

Keyboard
 scrolling in DECterm, 2–16

L

Language bindings, 4–17
LBX
 See Low-Bandwidth X extension
LCN
 See Logical Connection Number interface
Log files
 DECW\$STARTUP_n.LOG, 3–1

Logical Connection Number (LCN) interface, 4–54
 allocating connection numbers, 4–55
 functions, 4–55
 initializing thread support, 4–55
 querying status, 4–55
 related Xlib routines, 4–16
 routines, 4–56
 DECW\$LCN_ALLOCATE, 4–56
 DECW\$LCN_CLEAR_x_READY, 4–57
 DECW\$LCN_FREE, 4–58
 DECW\$LCN_SELECT, 4–59
 DECW\$LCN_SELECT_ONE, 4–61
 DECW\$LCN_SET_x_READY, 4–63
 DECW\$LCN_THREAD_INIT, 4–64
 signaling input, 4–55
Logical names
 defining in the CDA Viewer, 2–10
Login screen
 welcome message, 2–8
Low-Bandwidth X (LBX) extension, 4–39

M

MIT Shared Memory extension (MIT-SHM), 4–49
Motif
 variables
 UIDPATH, 4–35
Motif Window Manager
 resource for XUI application support, 2–9
Multithreading
 binary compatibility, 4–1
 enabling support for existing applications, 4–4
 supported libraries, 4–1
 thread safety and concurrency, 4–2
 use of ASTs, 4–2

N

New Desktop
 logging in, 2–8

O

Options file
 specifying, 2–10
Overlay support
 See also Window Manager
 colormap
 avoiding potential problems, 2–17
 modifying applications, 2–17, 2–18
 sharing overlay colormaps with the Window Manager, 2–18

R

Rasterizer
 See Agfa Monotype iType rasterizer
ReGIS input cursors
 See also Input cursors

S

Security extension (SECURITY), 4–40
Shared memory
 creating and using XImages, 4–50
 extension support, 4–49
 pixmap, 4–53
Startup
 log file, 3–1
SVN widget
 extended selection, 4–10
SYNC
 See X Synchronization extension

T

Tear-off menus
 using, 2–8
TOG-CUP
 See Colormap Utilization Policy extension
Toolkit
 changes to CompositeClassExtensionRec, 4–13
 easy resource configuration, 4–12
 extensions
 DXmCSText widget, 4–10
 SVN widget, 4–10
Translated image support
 Extended File Specifications (EFS), 2–2
Transports
 LAT interface support, 4–54

U

Utilities
 AccessX Keyboard utility (accessx), 2–19
 Window Dump to Print File utility (xpr), 2–33
 X Authority utility (xauth), 2–22
 X Keyboard Compiler utility (xkbcomp), 2–31

W

Window Dump to Print File utility (xpr), 2–33
Window Manager
 overlay support, 2–17

X

X\$CLOSE_OM routine, 4-17
X\$CONTEXTUAL_DRAWING routine, 4-18
X\$CONVERT_CASE routine, 4-18
X\$DESTROY_OC routine, 4-18
X\$DIRECTIONAL_DEPENDENT_DRAWING routine, 4-18
X\$DISPLAY_OF_OM routine, 4-19
X\$EXTENDED_MAX_REQUEST_SIZE routine, 4-19
X\$INIT_IMAGE routine, 4-19
X\$INIT_THREADS routine, 4-19
X\$INTERNAL_CONNECTION_NUMBERS routine, 4-20
X\$LOCALE_OF_OM routine, 4-20
X\$LOCK_DISPLAY routine, 4-20
X\$OPEN_OM routine, 4-20
X\$PROCESS_INTERNAL_CONNECTION routine, 4-21
X\$REGISTER_IM_INSTANTIATE_CB routine, 4-21
X\$SET_AUTHORIZATION routine, 4-21
X\$UNLOCK_DISPLAY routine, 4-21
X\$UNREGISTER_IM_INSTANTIATE_CB routine, 4-22
xauth
 See X Authority utility
X authority file
 adding and removing file entries, 2-29
 copying entries between files, 2-30
 creating, 2-27
 definition, 2-24
 displaying file entries, 2-28
 format, 2-25
 generating authorization keys, 2-31
 specifying, 2-26
 viewing file information, 2-27
X Authority utility (xauth), 2-22
 entering commands, 2-26
 IPv6 considerations, 2-27
 viewing and editing files, 2-28
XC-APPGROUP
 See Application Group extension
XC-MISC extension, 4-41
X Display Server
 extensions, 4-44
 Application Group extension, 4-37
 Big Requests extension, 4-38
 Colormap Utilization Policy extension, 4-38
 Extended Visual Information extension, 4-39
 Low-Bandwidth X extension, 4-39
 MIT Shared Memory extension (MIT-SHM), 4-49
 Security extension, 4-40

X Display Server

 extensions (cont'd)
 XC-MISC extension, 4-41
 X Double Buffer extension, 4-41
 XINERAMA extension, 4-42
 X Keyboard extension, 4-42
 X Synchronization extension, 4-43
 parameters
 DECW\$IPV6_FONT_SUPPORT, 3-2
 shared memory pixmaps, 4-53
 shared memory XImages, 4-50
 supported extensions, 4-36
X Double Buffer extension (DBE), 4-41
X Image extension, 4-53
XINERAMA extension, 4-42
X Keyboard Compiler utility (xkbcomp), 2-31
X Keyboard extension (XKB), 4-42
Xlib
 changes for IPv6 support, 4-14
 client-side extension library updates for
 X11R6.6, 4-15
 extensions
 client side library, 4-36
LCN support, 4-16
routines
 XAddConnectionWatch, 4-14
 _XAllocTemp, 4-14
 XCloseOM, 4-14
 XcmsSetCCCOFColormap, 4-14
 XContextualDrawing, 4-14
 XConvertCase, 4-14
 XCreateOC, 4-14
 XDestroyOC, 4-14
 XDirectionalDependentDrawing, 4-14
 XDisplayOfOM, 4-14
 XESetBeforeFlush, 4-15
 XExtendedMaxRequestSize, 4-15
 _XFreeTemp, 4-14
 XGetAtomNames, 4-15
 XGetOCValues, 4-15
 XGetOMValues, 4-15
 XInitThreads, 4-15
 XInternalConnectionNumbers, 4-15
 XInternAtoms, 4-15
 XLocaleOfOM, 4-15
 XLockDisplay, 4-15
 XOMOfOC, 4-15
 XOpenOM, 4-15
 XProcessInternalConnection, 4-15
 XReadBitmapFileData, 4-15
 XRegisterIMInstantiateCallback, 4-15
 XRemoveConnectionWatch, 4-15
 XSetOCValues, 4-15
 XSetOMValues, 4-15
 XUnlockDisplay, 4-15
 XUnregisterIMInstantiateCallback, 4-15
 updates for X11R6.6, 4-14
 variables

- Xlib
 - variables (cont'd)
 - DISPLAY, 4-17
 - RESOURCE_NAME, 4-17
- XmNinputMethod resource
 - using the shell to specify input methods, 4-10
- xpr
 - See Window Dump to Print File utility
- X Session Management protocol (XSMP), 4-47
 - differences from the X11R6.6 implementation, 4-48
 - multithread considerations, 4-48
- XSMP
 - See X Session Management protocol
- X Synchronization extension (SYNC), 4-43
- XtAppMainLoop routine, 4-13
- X Toolkit
 - routines
 - XtAppAddBlockHook, 4-11
 - XtAppAddSignal, 4-11
 - XtAppGetExitFlag, 4-11
 - XtAppLock, 4-11
 - XtAppMainLoop, 4-13
 - XtAppSetExitFlag, 4-11
 - XtAppUnlock, 4-11
 - XtCancelSelectionRequest, 4-11
 - XtChangeManagedSet, 4-11
 - XtCreateSelectionRequest, 4-11
 - XtDispatchEventToWidget, 4-11
 - XtGetClassExtension, 4-11
 - XtGetDisplays, 4-11
 - XtGetKeyboardFocusWidget, 4-11
 - XtGetSelectionParameters, 4-11
 - XtHooksOfDisplay, 4-11
 - XtInsertEventTypeHandler, 4-11
 - XtIsSessionShell, 4-11
 - XtLastEventProcessed, 4-11
 - XtNoticeSignal, 4-11
 - XtOpenApplication, 4-11
 - XtProcessLock, 4-11
 - XtProcessUnlock, 4-11
 - XtRegisterDrawable, 4-11
 - XtRegisterExtensionSelector, 4-12
 - XtReleasePropertyAtom, 4-12
 - XtRemoveBlockHook, 4-12
 - XtRemoveEventTypeHandler, 4-12
 - XtRemoveSignal, 4-12
 - XtReservePropertyAtom, 4-12
 - XtResolvePathname, 4-13
 - XtSendSelectionRequest, 4-12
 - XtSessionGetToken, 4-12
 - XtSessionReturnToken, 4-12
 - XtSetEventDispatcher, 4-12
 - XtSetSelectionParameters, 4-12
 - XtToolkitThreadInitialize, 4-12
 - XtUnregisterDrawable, 4-12
 - XtVaOpenApplication, 4-12
- X Toolkit (cont'd)
 - updates for X11R6.6, 4-11
 - variables
 - sessionShellClassRec, 4-12
 - sessionShellWidgetClass, 4-12

