graphics administration guide
for HP-UX 11.X servers

HP Part Number: 5992-4651
Published: July 2008
Edition: 2

Legal Notices

Table of Contents

1 preface....9
publishing history....9
document conventions....9

2 configuring X Server on HP-UX (OEM graphics cards)....11
using SAM/SMH to configure the X Server....11
using setmon to configure the monitor....13
the XF86Config file....13
the XF86Config file format....13
ServerLayout section....14
Files section....16
Module section....17
InputDevice section....18
Screen section....19
Display subsection....21
Monitor section....22
Device section....22
sample XF86Config file....23
extensions....26
double buffer extension (DBE)....26
display power management signaling (DPMS)....26
dynamic library loading....28
features....28
cursor scaling....28
virtual frame buffer (Xvfb)....29
security....29
connecting to the network....29
granting access....29
signals....29
starting the X Server from the command line....29
mapping options from the previous hp X Server to the XF86 hp X Server....30
defaultVisual option....30
HPCursorScaleFactor <n>....30
input devices....30
keyboards....30
supported keyboard drivers....30
supported keyboard options....30
pointers....31
supported pointer drivers....31
supported pointer options....31

output devices....31
ATI Radeon™ family device-dependent information....31
supported visuals....31
supported monitor configurations....32
Framebuffer depth and colormaps....33
ATI Radeon family configuration hints....33
hardware compatibility....33
compatibility table with other display adapters....34
miscellaneous....34
fonts....34
files....35

3 configuring X server on HP-UX (HP Visualize-FXE)....37
X Server configuration....37
using setmon to configure the monitor....39
X*screens file....39
description of the X*screens configuration file....39
syntax guidelines....40
the X*screens file format....40
server options....40
sample X*screens files....42
miscellaneous topics....43
double buffer extensions....43
performing buffer swaps on vertical blank....43
determining swap performance....43
display power management signaling (DPMS)....44
shared memory extension (MIT_SHM)....44
hp Visualize-FXE device-dependent information....45
supported visuals....45
supported screen options....46
hp VISUALIZE-FXE configuration hints....46
overlay visuals and overlay transparency....46
disabling the GLX visuals....46
hp VISUALIZE-FXE colormaps....47
changing the monitor type....47

4 X Server configuration details....49
making an X*.hosts file....49
using an /etc/hosts file....49
customizing the mouse and keyboard....49
changing mouse button actions....50
modifying modifier key bindings with xmodmap....50
specifying key remapping expressions....51

examples 52
printing a key map.....52
stopping the X Window system 53

List of Figures

3-1 Results of minimal legal X*screens file.....42

List of Tables

1-1 publishing history details....9
2-1 Options for ServerLayout section....16
2-2 Options for InputDevice section....19
2-3 Options for Screen section....20
2-4 DPMS Levels....26
2-5 DPMS Settings....28
2-6 Keyboard Options Supported by HP....31
2-7 Pointer Options Supported by HP....31
2-8 Supported Monitor Options....32
3-1 Power saving states defined by VESA....44
4-1 Expression Types....51

1 preface

The purpose of this document is to collect, in one place, all the information necessary to configure and administer graphics cards supported in HP-UX servers running the 11.00, 11i version 1 (11.11), 11i version 2 (11.23), and 11i version 3 (11.31) Operating Systems.

NOTE: Previous versions of this document contained information for HP-UX workstations and the graphics adapters supported on workstations.

This document deals only with HP-UX servers and the graphics adapters available on servers.

publishing history

This section provides the publishing history of the document.

Table 1-1 publishing history details

Date	Edition	Part Number (MPN)	Supported Operating System
February 2007	1	5991-7583	HP-UX 11.x
July 2008	2	5992-4651	HP-UX 11.x

document conventions

Below is a list of the typographical conventions used in this document:

`ls /usr/include`

Verbatim computer literals are in computer font. Text in this style is letter-for-letter verbatim and, depending on the context, should be typed in exactly as specified, or is named exactly as specified.

In every case...

Emphasized words are in italic type.

. . .to configure a **Single Logical Screen**. . .

New terms being introduced are in bold-faced type.

. . .the <*device_id*. . .>

Conceptual values are in italic type, enclosed in angle brackets. These items are not verbatim values, but are descriptors of the type of item it is, and the user should replace the conceptual item with whatever value is appropriate for the context.

2 configuring X Server on HP-UX (OEM graphics cards)

This chapter documents information specific to the HP Xf86 X Server. The Xf86 X Server is based on the XFree86 version 4.2.0 X Server. This section describes features unique to HP's implementation of the X Server, provides information on how to configure the X Server and includes a list of supported X server configurations. For each supported graphics device, device-dependent configuration information is provided.

NOTE: This chapter deals with configuration requirements for graphics cards and built-in core graphics devices made by OEM graphics manufacturers such as ATI. For configuration information for HP Visualize cards, refer to Chapter 3 in this document.

using SAM/SMH to configure the X Server

Configuration of the X Server is supported through SAM (HP-UX 11.23 and earlier) or SMH (HP-UX 11.31 and later) via a utility titled “X Server Configuration.” This utility resides under the “Display” panel.

The SAM/SMH graphical user interface for X Server configuration is provided to simplify and facilitate ease of use in modifying or creating the X Server configuration file, XF86Config. The XF86 Server uses the XF86Config file for its configurations. While it is possible to modify this file manually (see below), using the SAM/SMH interface can greatly simplify the process.

The SAM/SMH components have the following actions:

- Configure Print Server...
- Configure How X Starts...
- Modify Server Options...
- Modify Multi-Screen Layout...
- Single Logical Screen (SLS) ->
- Modify Default Depth...

- Describe Screen...
- Identify Screen
- Modify Screen Options...

- Add Screen to Configuration
- Remove Screen from Configuration

The first group of Actions menus can be thought of as global actions. They will typically be active regardless of what has been selected. If any of these menu items are not visible it is because they are not supported under the current configuration.

The Configure Print Server item allows you to manage print servers. From this menu item you can create, stop or remove print servers.

On systems that contain a mix of HP Visualize and OEM graphics cards, the Configure How X Starts item allows you to choose on which graphics devices the X Server should start. From this action, you can assign which of your configuration files to use as a display connection - the X* screens file for HP Visualize graphics cards or the XF86Config file with OEM graphics cards. Running independent X Servers on an HP Visualize graphics device and an OEM graphics device simultaneously is not supported.

Specific XF86 server options can be set with the Modify Server Options menu item. See the item for information on specific options.

If you have multiple OEM graphics display devices or a display card with two video outputs, you can configure the X server to have multiple independent screens. Use the Modify Multi-Screen Layout menu item to do this.

To configure a multi-screen layout, first select the devices you want to include. In the X Server Configuration panel click the mouse on the first device, and **[Ctrl]**-click on the others. All the devices you want to combine into a multi-screen configuration should now be highlighted. Then use the Modify Multi-Screen Layout menu item in the Actions menu to finish the configuration.

SLS is a mechanism for treating homogeneous multi-display configurations as a single logical screen. This allows the moving/spanning of windows across multiple physical monitors. The word homogeneous is included because SLS only works if the graphics devices included in the SLS Configuration are of the same type.

Enabling an SLS configuration is similar to enabling a multi-screen layout. First, select the devices you want to combine into an SLS configuration. In the X Server Configuration panel, click the mouse on the first device, and **[Ctrl]**-click on the others. All of the devices you want to combine into a multi-screen configuration should now be highlighted. Then from the "Actions" menu, choose the Single Logical Screen (SLS) -> Create SLS... menu item to finish the configuration. The "X Server Configuration" window should now show a single icon denoting an SLS confguration.

The Modify Default Depth menu item lets you set the default depth of the framebuffer to either 8 or 24 bits. Changing the default depth will also change the default visual. Depth 8 will use a PseudoColor visual and depth 24 will use a TrueColor visual.

The second group of “Actions” menus can be thought of as screen actions. They will be active depending on which screens have been chosen.

The Describe Screen and Identify Screen menu selections provide information about the device. Identify Screen flashes the monitor that is connected to the graphics device.

The Modify Screens Options item contains options that are specific to each graphics device. This list might be different for cards depending on the capabilities of each card.

Grayed out screen icons represent screens that have not been configured for use by the X Server. You can select these grayed out icons and choose the Add Screen to Configuration menu item to add screens to the XF86Config file.

More information on configuration of the X Server and each of the above actions can be obtained from SAM/SMH on-line Help.

using setmon to configure the monitor

setmon is a configuration tool used to change the monitor settings for a monitor attached to a graphics device. This tool permits you to change the monitor's refresh rate and frame buffer resolution. To change the monitor setting, the setmon command can be executed directly or done through SAM/SMH.

The setmon executable is located at `/opt/graphics/common/bin/setmon`. Under SAM/SMH this component is located under the top-level "Display" panel, under "Monitor Configuration".

NOTE: Changing the monitor type while the X Server is running will necessitate killing and restarting the X Server. In order to change the monitor settings, the X Server needs to be running on the device specified. For these graphics cards, it may not be possible to test some of the monitor settings before making the change permanent.

the XF86Config file

The XF86Config file is located in `/etc/X11/XF86Config`. It can be generated automatically or modified using SAM/SMH. A working configuration file is also delivered on the system. You must be root to create or edit this file. The XF86Config man page provides additional information regarding the configuration file. It is necessary to re-start the X Server for changes made to the XF86Config file to take effect.

the XF86Config file format

Most of the content in this section has been copied from the XF86Config(5) man page listed on "The XFree86 Project, Inc." web site (http://www.xfree86.org). The man pages are available from http://www.xfree86.org/4.2.0

Config file keywords are case-insensitive, and underscore "_" characters are ignored. Most strings (including Option names) are also case-insensitive, and insensitive to white space and underscore "_" characters.

Each config file entry usually takes up a single line in the file. They consist of a keyword, which is possibly followed by one or more arguments, with the number and types of the arguments depending on the keyword. The argument types are:

- Integer - an integer number in decimal, hex or octal
- Real - a floating point number
- String - a string enclosed in double quote marks (")

NOTE: Hex integer values must be prefixed with “0x”, and octal values with “0”.

A special keyword called Option may be used to provide free-form data to various components of the server. The Option keyword takes either one or two string arguments. The first is the option name, and the optional second argument is the option value. Some commonly used option value types include:

- Integer - an integer number in decimal, hex or octal
- Real - a floating point number
- String - a sequence of characters
- Boolean - a boolean value (see below)
- Frequency - a frequency value (see below)

NOTE: All Option values, not just strings, must be enclosed in quotes.

Boolean options may optionally have a value specified. When no value is specified, the option's value is TRUE. The following boolean option values are recognized as TRUE:

1, on, true, yes

and the following boolean option values are recognized as FALSE:

0, off, false, no

If an option name is prefixed with “No”, then the option value is negated.

Frequency option values consist of a real number that is optionally followed by one of the following frequency units:

Hz, k, kHz, M, MHz

When the unit name is omitted, the correct units will be determined from the value and the expectations of the appropriate range of the value. It is recommended that the units always be specified when using frequency option values to avoid any errors in determining the value.

ServerLayout section

The ServerLayout section is used to identify which Screen sections are to be used in a multi-headed configuration, the relative layout of those screens, and which InputDevice sections are to be used. Each ServerLayout section has an Identifier, a list of Screen section identifiers, and a list of InputDevice section identifiers. Options may also be included in the ServerLayout section. A ServerLayout section may be made active by referencing (via its Identifier) on the command line that starts X. In the absence of this, the first one found in the file will be chosen by default, as there may be multiple ServerLayout sections in the config file. The format of the ServerLayout section is as follows:

```
Section “ServerLayout”
   Identifier “ServerLayoutName”
   Screen [ScreenNumber] “ScreenID” [Position] [Xcoor] [Ycoor]
   . . .
   InputDevice  “InputDeviceID” “InputDeviceOption”
   . . .
   [Option …]
   . . .
EndSection
```

Keywords, options and values enclosed in [ ] are optional.

A number specifying the preferred screen number for that screen may optionally follow each Screen. When no screen number is specified, it is numbered according to the order in which it is listed. Next comes the ScreenID, a required field that must be enclosed in double quotes. The ScreenID must match an Identifier in a Screen section. The remaining information on the line is optional. Next comes the physical position of the screen, either in absolute terms or relative to another screen (or screens). Finally the XY coordinates of the screen may be specified.

The position keywords are:

```
Absolute
RightOf
LeftOf
Above
Below
Relative
```

The preferred method of specifying the layout is to explicitly specify the screen's location in absolute terms or relative to another screen.

The examples are based on the examples listed in the DESIGN document from XFree86.

In the absolute case, the upper left corner's coordinates are given after the Absolute keyword. If the coordinates are omitted, a value of (0,0) is assumed. An example of absolute positioning follows:

```
Section "ServerLayout"
   Identifier "Main Layout"
   Screen 0 "Screen 0" Absolute
   Screen 1 "Screen 1" Absolute 1024 0
   Screen 2 "Screen 2" Absolute 2048 0
   ...
EndSection
```

When the Relative keyword is used, the coordinates of the new screen's origin relative to the reference screen follow the reference screen name. The following example shows how to use some of the relative positioning options:

```
Section "ServerLayout"
   Identifier "Main Layout"
   Screen 0 "Screen 0"
   Screen 1 "Screen 1" RightOf "Screen 0"
   Screen 2 "Screen 2" RightOf "Screen 1"
```

```
    ...
EndSection
```

Each InputDevice is followed by an InputDeviceID, a required field that must be enclosed in double quotes. The InputDeviceID must match an Identifier in an InputDevice section. Last, an option may be provided. The option can also be specified in the InputDevice section. Typical options specified here are: CorePointer, CoreKeyboard, and SendCoreEvents. The option must be enclosed in double quotes. See the InputDevice section for more information regarding the options. Normally, at least two InputDevices are present: a keyboard and a mouse.

Options that apply to the X Server may also be specified in this section. The following table lists all options that may be set in the ServerLayout section.

Table 2-1 Options for ServerLayout section

Option	Value	Default	Description
DontZap	Boolean	Off	Disallows use of the Ctrl+Shift+Break sequence. That sequence is normally used to terminate the X Server. When this option is enabled, that key sequence has no special meaning and is passed to clients. Source: XF86Config man page.
AllowMouseOpenFail	Boolean	false	Allows the server to start up even if the mouse device can't be opened/initialized.Source: XF86Config man page.
CursorScaleFactor	Integer	1	See the section in “Features: Cursor Scaling” for more details regarding these options.
MaxCursorSize	Integer	64	See the section in “Features: Cursor Scaling” for more details regarding these options.

Files section

The Files section is used to specify paths to where fonts and modules are located, the location of the rgb database and the user specified logfile. The Files section format is:

```
Section "Files"
    [FontPath   "PathName"]
    .
    .
    [ModulePath "PathName"]
    .
    .
    [RgbPath    "PathName"]
    [LogPath    "PathName"]
EndSection
```

Multiple Font Paths and Module Paths may be specified in two ways, either by multiple lines or by using a “,” delimiter between paths on the same line.

Font Path elements may be either absolute directory paths or a font server identifier. Font server identifiers have the form:

<trans>/<hostname>:<port-number>

where <trans> is the transport type to use to connect to the font server (e.g., Unix for UNIX-domain sockets or tcp for a TCP/IP connection), <hostname> is the hostname of the machine running the font server, and <port-number> is the port number that the font server is listening on (usually 7000). The default Font Path is:

```
tcp:/7000,
/usr/lib/X11/fonts/hp_roman8/75dpi/,
/usr/lib/X11/fonts/iso_8859.1/100dpi/,
/usr/lib/X11/fonts/iso_8859.1/75dpi/,
/usr/lib/X11/fonts/hp_kana8/,
/usr/lib/X11/fonts/hp_japanese/100dpi/,
/usr/lib/X11/fonts/hp_japanese/75dpi/,
/usr/lib/X11/fonts/hp_korean/75dpi/,
/usr/lib/X11/fonts/hp_chinese_s/75dpi/,
/usr/lib/X11/fonts/hp_chinese_t/75dpi/,
/usr/lib/X11/fonts/iso_8859.2/75dpi/,
/usr/lib/X11/fonts/iso_8859.5/75dpi/,
/usr/lib/X11/fonts/iso_8859.6/75dpi/,
/usr/lib/X11/fonts/iso_8859.7/75dpi/
/usr/lib/X11/fonts/iso_8859.8/75dpi/,
/usr/lib/X11/fonts/iso_8859.9/75dpi/,
/usr/lib/X11/fonts/misc/
```

Xf86 uses ModulePaths as locations to look for loadable modules. The default ModulePath is:

```
/usr/lib/X11/Xserver/modules/xf86/,
/opt/graphics/common/lib/
```

RgbPath can be used to specify the RGB database path. Normally it is never changed. If it is not specified the built-in path `/etc/X11` is used.

In addition, the LogPath can be specified, if server logging information is to be sent somewhere other than the default log file. The default log file is located at `/var/X11/Xserver/logs/Xf86.n.log`, where n is the display number.

All names must be enclosed within double quotes. There may be only one Files section in the config file. This section does not recognize Option as a keyword.

Module section

The Module section is used to specify which X Server modules should be loaded. The types of modules normally loaded in this section are X Server extension modules, and font rasterizer modules. Most other module types are loaded automatically when they are needed via other mechanisms. There may only be one Module section in the config file. The format of the Module section is as follows:

```
Section "Module"
         Load "ModuleName"
```

```
            . . .
           [SubSection "ModuleName"
               Option . . .
           EndSubSection]
            . . .
    EndSection.
```

Load instructs the server to load the module called ModuleName. The module name given should be the module's extension name, not the module file name. The extension name is case sensitive, and does not include the "lib" prefix, or the ".so.1" or ".1" suffix.

Example: the Double Buffered Extension (DBE) can be loaded with the following entry:

Load "dbe"

SubSection also instructs the server to load the module called ModuleName. The module name given should be the module's extension name, not the module file name. The extension name is case-sensitive, and does not include the "lib" prefix, or the ".so.1" or ".1" suffix. The difference between Load and SubSection is that in SubSection the listed Options are passed to the module when it is loaded.

Modules are searched for in each directory specified in the ModulePath search path (or the default ModulePath if one is not specified in the Files section) and in the drivers, input, extensions, fonts, and hpux subdirectories of each directory in the ModulePath.

Noload instructs the server to not load the module called ModuleName.

InputDevice section

An InputDevice section is considered active if there is a reference to it in the active ServerLayout section. There may be multiple InputDevice sections. There will normally be at least two: one for the core (primary) keyboard, and one for the core pointer. InputDevice sections have the following format:

```
Section "InputDevice"
          Identifier    "InputDeviceID"
           Driver         "DriverName"
          [Option ...]
     . . .
EndSection
```

The Identifier entry specifies the unique name for this input device and must match an InputDeviceID in the active ServerLayout section in order to be active.

The Driver entry specifies the name of the driver to use for this input device.

InputDevice sections recognize some driver-independent Options, which are described here. See the individual input driver manual pages for a description of the device-specific options that can be entered here.

Table 2-2 Options for InputDevice section

Option	Value	Description
CorePointer	NA	When this is set, the input device is installed as the core (primary) pointer device. There must be no more than one core pointer. If this option is not set here, or in the ServerLayout section, or from the -pointer command line option, then the first input device that is capable of being used as a core pointer will be selected as the core pointer. Source: XF86Config man page.
CoreKeyboard	NA	When this is set, the input device is to be installed as the core (primary) keyboard device. There must be no more than one core keyboard. If this option is not set here, or in the ServerLayout section, then the first input device that is capable of being used as a core keyboard will be selected as the core keyboard. Source: XF86Config man page.
AlwaysCore SendCoreEvents	boolean	Both of these options are equivalent, and when enabled cause the input device to always report core events. This can be used, for example, to allow additional pointer devices to generate core pointer events (such as moving the cursor, etc). Source: XF86Config man page.
HistorySize	integer	Sets the motion history size. Default: 0. Source: XF86Config man page.

The following two examples show an InputDevice section for a keyboard and mouse:

```
Section "InputDevice"
         Identifier      "Keyboard0"
         Driver        "keyboard"
EndSection
Section "InputDevice"
         Identifier      "Mouse0"
         Driver         "mouse"
         Option         "Protocol"    "PS/2"
EndSection
```

Screen section

The configuration file may have multiple Screen sections. There must be at least one, for the "screen" being used. A "screen" binds a graphics device (Device section) and a monitor (Monitor section) together. A Screen section is considered "active" if it is referenced by an active ServerLayout section. If neither of these is present, the first Screen section found in the configuration file is considered the active one. Screen sections have the following format:

```
Section "Screen"
       Identifier "ScreenID"
       Device     "DeviceID"
       Monitor    "MonitorID"
```

```
        DefaultDepth     <Depth>
        Option ...
        .
        .
        SubSection “Display”
                .
                .
        EndSubSection
        .
EndSection
```

The Identifier entry specifies the unique name for this screen. The Identifier generally must match a ScreenID listed in the active ServerLayout section. The Screen section provides information specific to the whole screen, including screen-specific Options. In multi-screen configurations, there will be multiple active Screen sections, one for each head.

The Device keyword specifies which Device section is used for this screen. This is what binds a specific graphics card to a screen. The DeviceID must match the Identifier of a Device section in the configuration file.

The Monitor keyword specifies which Monitor section is used for this screen. This is what binds a specific monitor to the screen. The MonitorID must match the Identifier of a Monitor section in the configuration file.

The DefaultDepth keyword specifies which color depth the server should use by default. The -depth command line option can be used to override this. If neither is specified, the default depth is driver specific, but in most cases is 8.

Various Option flags may be specified in the Screen section. Some are driver specific and are described in the driver documentation. Driver-independent options are described here.

Table 2-3 Options for Screen section

Entry	Entry Position	Description
Accel	NA	Enables XAA (X Acceleration Architecture), a mechanism that makes video cards' 2D hardware acceleration available to the Xserver. This option is on by default. There are many options to disable specific accelerated operations. Note that disabling an operation will have no effect if the operation is not accelerated (whether due to lack of support in the hardware or in the driver). Source: XF86Config man page.

Each Screen section must contain one or more Display subsections. Those subsections provide depth configuration information, and the one chosen depends on the depth that is being used for the screen. The Display subsection format is described in the section below.

Display subsection

The Screen sections include one or more Display subsections. One Display subsection may be provided for each depth that the server supports. The size of the Xserver's virtual screen may also be specified. The virtual screen allows you to have a "root window" larger than what can be displayed on the monitor. (e.g. the monitor may be a 800x600 display, but have a 1280x1024 virtual size). The Virtual keyword is used to specify this size. Note that many of the new accelerated graphics drivers use non-displayed memory for caching. It is not desirable to use all available memory for the virtual display, as this leaves none for caching, and this can decrease server performance. Display subsections have the following format:

```
SubSection "Display"
    Depth <depth>
    Visual <visual>
    Modes "ModeName"
    ViewPort <x0> <y0>
    Option ...
    ...
EndSubSection
```

The Depth entry specifies what color depth the Display subsection is to be used for. Only depths of 8 and 24 are supported.

The Modes entry specifies the list of video modes to use. Each ModeName specified must be in double quotes. They must correspond to those specified or referenced in the appropriate Monitor section. The server will delete modes from this list which don't satisfy various requirements. The first valid mode in this list will be the default display mode for startup. The list of valid modes is converted internally into a circular list. It is possible to switch to the next mode with Ctrl+Alt+Keypad-Plus and to the previous mode with Ctrl+Alt+Keypad-Minus. When this entry is omitted, the largest valid mode referenced by the appropriate Monitor section will be used.

The Visual entry is optional and sets the default root visual type.

The visual type available for the depth 8 is: PseudoColor

The visual type available for depth 24 is: TrueColor

The ViewPort optional entry sets the upper left corner of the initial display. This is only relevant when the virtual screen resolution is different from the resolution of the initial video mode. If this entry is not given, then the initial display will be centered in the virtual display area. Source: XF86Config man page.

Option flags may be specified in the Display subsections. These may include driver-specific options or driver-independent options. The former are described in the driver-specific documentation. Some of the latter are described above in the section about the Screen section, and they may also be included here. However, options set in the Display subsection may be "overridden" in the Screen section.

Monitor section

The configuration file may have multiple Monitor sections. The Monitor section provides information about the specifications of the monitor, monitor-specific Options, and information about the video modes to use with the monitor. There must be at least one Monitor section, for the monitor being used. A Monitor section is considered “active” if it is referenced by an active Screen section. Monitor sections have the following format:

```
Section “Monitor”
     Identifier     “MonitorID”
     VendorName     “Vname”
     ModelName      “Mname”
     HorizSync     horizsync-range
     VertRefresh    vertrefresh-range
     DisplaySize    width  height
     Option …
EndSection
```

The Identifier entry specifies the unique name for this monitor.

The VendorName is an optional entry and is used to specify the monitor's manufacturer.

The ModelName is an optional entry that is used to specify the monitor model.

HorizSync gives the range(s) of horizontal sync frequencies supported by the monitor. horizsync-range may be a comma separated list of either discrete values or ranges of values. A range of values is two values separated by a dash. By default the values are in units of kHz. They may be specified in MHz or Hz if MHz or Hz is added to the end of the line. The data given here is used by the X Server to determine if video modes are within the specifications of the monitor. This information should be available in the monitor's handbook. If this entry is omitted, a default range of 28-33 kHz is used. Source: XF86Config man page.

VertRefresh gives the range(s) of vertical refresh frequencies supported by the monitor. vertrefresh-range may be a comma separated list of either discrete values or ranges of values. A range of values is two values separated by a dash. By default the values are in units of Hz. They may be specified in MHz or kHz if MHz or kHz is added to the end of the line. The data given here is used by the X Server to determine if video modes are within the specifications of the monitor. This information should be available in the monitor's handbook. If this entry is omitted, a default range of 43-72Hz is used. Source: XF86Config man page.

DisplaySize is an optional entry which gives the width and height, in millimeters, of the picture area of the monitor. If given these values are used to calculate the horizontal and vertical pitch (DPI) of the screen. Source: XF86Config man page.

Device section

The configuration file may have multiple Device sections. There must be at least one, for the video card being used. Device sections have the following format:

```
Section “Device”
     Identifier      “DeviceID”
     Devicefile       “/dev/gvid”
      Option …
     . . .
EndSection
```

The Identifier entry specifies the unique name for this graphics device. It must match a DriverID in the active Screen section.

sample XF86Config file

```
# This is a sample XF86Config file. It can be cut from this document
# and placed in the /etc/X11/XF86Config file.
#
# The config file has a hierarchical “Section” structure along
# with some standalone “Sections.”
#
# The standalone sections are the Files, Module, and DRI Sections.
# There may only be one of each of these sections in the config file.
#
# The hierarchical section consists of the ServerLayout, InputDevice,
# Screen, Monitor, and Device sections. There may be multiple sections
# of each.
#
# Each screen section in turn specifies a Monitor and a Device
# section.
#
# Check the document “Graphics Administration Guide” for complete
# documentation of the config file organization and description of all
# options. An online version of the Graphics Administration Guide is
# located online at:
# http://docs.hp.com
# Search for “5991-7583”, the part number of this document.
# Comment/uncomment/modify as needed.
#
# The ServerLayout section specifies the input and output devices that are
# connected to the server. Multiple ServerLayout sections may be contained in
# the XF86Config file. However, the first one in the file is the active
# layout, unless otherwise specified by the -layout option from the command
# line. Check the "Graphics Administration Guide" (GAG) for other options that
# may be set here, or elsewhere, in the XF86Config file.
#
Section "ServerLayout"

 #
 # The ServerLayout ID. A required line.
 #
 Identifier     "Main Layout"

 #
 # The first field on the Screen line specifies the screen number. It is
 # optional. The second field is the Screen ID. It must match an entry in
 # a Screen section. Only Screens specified here will be active. The
 # remaining fields specify relative or absolute positions of the screen
 # relative to other screens. Check the GAG for full details on
 # specifying the Screen.
 #

 Screen      0  "Screen 0" 0 0

 #
```

```
 # Each InputDevice line specifies an InputDevice section ID name and
 # optionally some options that specify the way the device is to be used.
 # Typically there is a pointer device (mouse) and a keyboard. They
 # usually are specified with a CorePointer and CoreKeyboard option
 # respectively. Additional pointers and keyboards are specified with
 # the SendCoreEvents option. The options may also be specified in the
 # InputDevice section. It is not necessary to specify an InputDevice.
 #

 InputDevice    "Mouse0" "CorePointer"
 InputDevice    "Keyboard0" "CoreKeyboard"

 #
 # Uncomment this to force OGL indirect contexts to be rendered in
 # software.  Indirect rendering is done with the hardware driver by
 # default.  However, some features such as rendering to a glXPixmap
 # may not be available in all hardware drivers.
 #

 #Option       "AccelerateIndirectRendering" "false"

 # Uncomment the following line and update the time to turn on Xserver
 # screen blanking. The time is in minutes.

 #Option "blank time"   "10"

 # Uncomment the following lines to set the DPMS time periods. The
 # time is in minutes. The DPMS Monitor Option must be on for these
 # to have an effect (see the "Monitor" section).

 #Option "standby time" "20"
 #Option "suspend time" "30"
 #Option "off time"     "40"
EndSection

#
# The Files section is used to specify the location of various files
# to the X server. There may only be one Files section in the XF86Config
# file.
#
Section "Files"

 #
 # FontPaths. Specifies the font paths. You may want to
 # specify a different font path for the following reasons.
 # 1) An application delivers its own fonts.
 # 2) A font server is to be used instead of the default path.
 #
EndSection

#
# The Module section is used to inform the server which loadable libraries are
# to be loaded at run time. There may only be one Module section in the
# XF86Config file.
# See the GAG for more details.
#
Section "Module"
EndSection

#
# There may be multiple InputDevice sections. An InputDevice section is active
# only if it is specified by the active ServerLayout section. The Identifief
# is a required line and must be identical to an InputDevice line in the
# active ServerLayout in order for the device to be active. Normally there
# are two InputDevice sections in the XF86Config file. One for the pointer
```

```
# (mouse) and the other for the keyboard. The Driver line is required. It
# specifies which driver is to be loaded at run time. See the GAG for more
# details on what input devices are supported and which options may be selected.
#
Section "InputDevice"
 Identifier "Keyboard0"
 Driver     "keyboard"
EndSection

Section "InputDevice"
 Identifier "Mouse0"
 Driver     "mouse"
 Option     "Protocol" "PS/2"
EndSection

#
# There may be multiple Monitor sections. The purpose of this section is
# is to specify the range of operation of a monitor. For a Monitor to be
# in use the Identifier must match the Monitor line in an active Screen.
# HorizSync and VertRefresh are required fields. See the GAG for more
# options that may be set.
#
Section "Monitor"
 Identifier   "Monitor 0"
 HorizSync    30.0 - 110.0
 VertRefresh  50.0 - 75.0

 # DPMS is not enabled by default. Uncomment the following line to
# enable it.

  #Option      "DPMS"  "on"
EndSection

#
# There may be multiple Device sections. This section is used to specify
# parameters for the graphics device. The Identifier string must match
# the Device string in the active Screen section for this device to be
# in use. See GAG for more options that may be set for this particular device.
#
Section "Device"
 Identifier      "Console"
 Devicefile      "/dev/gvid"
EndSection

#
# There may be multiple Screen sections. The Identifier string must match
# the Device string in the active ServerLayout section for the Screen to
# be active.
#
Section "Screen"
 Identifier "Screen 0"
 Device     "Console"
 Monitor    "Monitor 0"

 #
 # Set the default depth.
 #
 DefaultDepth     24

 #
 # The subsection associates a buffer depth with a screen size.
 #
 SubSection "Display"
  Depth     24
  Modes    "1280x1024"
```

```
 EndSubSection
EndSection
```

extensions

double buffer extension (DBE)

DBE is an extension to the X Server that provides a double-buffering API. For more information about DBE and the API, consult the DBE man pages:

DBE

XdbeQueryExtension

XdbeGetVisualInfo

XdbeFreeVisualInfo

XdbeScreenVisualInfo

XdbeAllocateBackBufferName

XdbeDeallocateBackBufferName

XdbeSwapBuffers

XdbeBeginIdiom

XdbeEndIdiom

XdbeGetBackBufferAttributes

display power management signaling (DPMS)

Monitors constitute a large percentage of the power used by a workstation even when not actively in use (i.e. during screen blanking). In order to reduce the power consumption, the Video Electronic Standards Association (VESA) has defined a Display Power Management Signaling (DPMS) standard which can be used to greatly reduce the amount of power being used by a monitor during screen blanking.

The following table is a description of the states that are defined by VESA. The Power Savings column indicates (roughly) the level of power savings achieved in the given state. The Recovery Time is the amount of time that the screen takes to return from a power saving state to an ON state (by pressing a key or the moving the mouse).

Table 2-4 DPMS Levels

Level	State	Power Savings	Recovery Time
0	On	None	None
1	Standby	Minimal	Short
2	Suspend	Substantial	Longer
3	Off	Maximum	System Dependent

The actual amount of power saved and the recovery time for each of the states is monitor dependent and may vary widely. The customer can compensate for this by choosing an appropriate level for the monitor that is currently in use.

By default, the DPMS level used is 0 or On (i.e. no power savings). If you wish to use power saving, set the following XF86Config file entries before starting the server: standby time, suspend time, and off time. Also set the Option “DPMS” “on” in the monitor section.

DPMS should not be confused with screen blanking or screen saving. Screen blanking will merely cause the screen to go dark. Screen saving will display a changing pattern on the screen to avoid the burn-in of a static image. Neither of these will cause any power saving to occur.

The DPMS Extension lets individual users customize their personal DPMS settings to meet their work styles and any restrictions imposed by their employers. For example, an employer may decide that all monitors must save power after 30 minutes of idle time. The individual user may decide that 30 minutes is too long, and adjust the time downward to meet their work preference.

More information (including sample code) on the DPMS Extension entry points can be found online, via the man pages. The extension entry points are:

```
DPMS
DPMSQueryExtension
DPMSGetVersion
DPMSCapable
DPMSSetTimeouts
DPMSGetTimeouts
DPMSEnable
DPMSDisable
DPMSForceLevel
DPMSInfo
```

XFree86 provides four options that may be set in the ServerLayout section that may be used to support this functionality. The options are: blank time, standby time, suspend time, and off time. The following example sets these to 10, 20, 30, and 60 minutes respectively.

```
Section “ServerLayout”
    . . .
      Option       “blank time”          “10”
      Option       “standby time”        “20”
      Option       “suspend time”        “30”
     Option        “off time”            “60”
    . . .
EndSection
```

Table 2-5 DPMS Settings

Option	Value	Default	Description
BlankTime	time	10	Sets the inactivity timeout for the blanking phase of the screensaver. Time is in minutes. This is equivalent to the Xserver's `-s' flag, and the value can be changed at run-time with xset(1).
StandbyTime	time	20	Sets the inactivity timeout for the “standby” phase of DPMS mode. Time is in minutes, and the value can be changed at run-time with xset(1).
SuspendTime	time	30	Sets the inactivity timeout for the “suspend” phase of DPMS mode. Time is in minutes, and the value can be changed at run-time with xset(1).
OffTime	time	40	Sets the inactivity timeout for the “off” phase of DPMS mode. Time is in minutes, and the value can be changed at run-time with xset(1).

dynamic library loading

The path for each dynamically loaded module must be specified in the ModulePath in order for them to load. See the section “The XF86Config File: Files Section” for more details regarding the ModulePath.

Dynamically loaded modules are recorded by the X Server in the `/var/X11/Xserver/logs` directory. The log file reflects the display identifier for a given run. Only the last invocation against a given display identifier is retained. The log file contains the parsed contents of the XF86Config file and the full path name for all dynamically loaded modules for the given X Server invocation. Deferred loaded modules are recorded as they are referenced.

NOTE: Altering or removing files under `/usr/lib/X11/Xserver` may prevent the X Server from running.

features

cursor scaling

There are times when the standard X11 cursors are difficult to see on the screen. The effect is compounded on large displays. Two options are available in the X Server that instruct the X Server to scale all X11 cursors (both user-defined and built-in cursors) by a user-defined value.

Cursor Scaling is indicated with the following syntax in the XF86Config file:

```
Section "ServerLayout"
      . . .
     Option         "CursorScaleFactor"       "n"
     Option         "MaxCursorSize"           "Size"
      . . .
EndSection
```

Where n = 1, 2, 3, …

Where Size = 2, 4, 8, 16, 32, 64, …

For example, n=2 instructs the X Server to scale all cursors by "2x" so that a 16x16 cursor becomes a 32x32 cursor and a 9x9 cursor becomes an 18x18 cursor, etc.

If the scaled width or height of any cursor is greater than Size, the scale factor is reduced so that the net size of the cursor fits into a Size x Size rectangle. Size needs to be a power of two.

The default value for "n" is 1, or no scaling. The default value for "Size" is 64, or 64 x 64 maximum size.

virtual frame buffer (Xvfb)

Xvfb(1) is an X Server that does not require display hardware or input devices. It emulates a video frame buffer by using the system's virtual memory.

Xvfb may be used for: rendering with non-standard depths and screen configurations, software rendering, providing a way to run applications that don't need an X Server but for some reason insist on having one, etc.

Generally the user application must use functions such as XGetImage(3) in order to see what was rendered.

security

See Xf86(1) for information on configuring the Xf86 security policies, files and settings.

connecting to the network

The X Server supports client connections via a platform-dependent subset of the following transport types: TCP/IP and Unix Domain sockets.

granting access

Information on X Server authorization may be found in the Xf86(1) man pages.

signals

See Xf86(1) for information on how the X Server handles signals.

starting the X Server from the command line

Command line options for the X Server are described in Xf86(1).

mapping options from the previous hp X Server to the XF86 hp X Server

The purpose of this section is to provide the user who is familiar with the X* screens files or the HP X Server a method of setting the equivalent options in the XF86Config file in the current release of the XF86 HP X Server. Only those options that are currently implemented in the release are documented here.

defaultVisual option

The default visual in the Xf86 HP X server can be implicitly set by changing the depth of the framebuffer. Depth 8 implies the PseudoColor visual and Depth 24 implies the TrueColor visual. No other visuals are supported. The following example demonstrates how this would be done in the X*screens file and how it would be done in the XF86Config file. The example sets the default visual class to TrueColor.

X*screens File Example:

```
Screen /dev/crt
     DefaultVisual
         Class TrueColor
```

XF86Config File Example:

```
Section "Screen"
     . . .
     DefaultDepth 24
     . . .
EndSection
```

DPMS options

See the section on "Display Power Management Signaling" for more details on these options.

HPCursorScaleFactor <n>

See the section "Features: Cursor Scaling" for more details regarding this option.

input devices

keyboards

supported keyboard drivers

The supported keyboard driver is: keyboard

supported keyboard options

The following is a list of keyboard options supported by HP.

Table 2-6 Keyboard Options Supported by HP

Options	Value	Description
AutoRepeat	Integer	Set the keyboard auto repeat parameters. Not all platforms implement this.
Xleds	Integer …	Specify which keyboard LEDs can be user-controlled (for example, with xset(1)).

pointers

supported pointer drivers

The supported pointer driver is:

mouse

supported pointer options

The following is a list of pointer options supported by HP.

Table 2-7 Pointer Options Supported by HP

Options	Value	Description
Protocol	String	Values may only be “PS/2”
Device	String	Device file for mouse.

output devices

ATI Radeon™ family device-dependent information

The ATI Radeon family includes the following graphics devices:

- Radeon 7000 PCI card
- Radeon 7500 PCI card
- A6869A VGA/USB PCI card with RV100 graphics chip
- RV100 core graphics
- ES1000 core graphics

supported visuals

ATI Radeon family graphics devices support the following visuals:

Class	**Depth**
PseudoColor	8
TrueColor	24

There are two visuals available with the ATI Radeon family displays, although not at the same time. The X server can be either in depth 8 or depth 24 frame buffer mode. In depth 24 mode the available visual is TrueColor. This is the recommended visual. In depth 8 mode the available visual is PseudoColor. The frame buffer depth is configurable through SAM/SMH.

supported monitor configurations

ATI Radeon family graphics devices support monitor resolutions from 1024x768 to 1920x1200 with the exception of the ES1000 based core graphics which only supports resolutions from 1024x768 to 1600x1200.

The Radeon 7000 and Radeon 7500 have two video connectors: a VGA analog connector and a DVI-I digital/analog connector. These devices will thus support up to two independent monitors. When using a single monitor, the monitor can be attached to either connector.

Some systems using the ES1000 core graphics may have two VGA connectors, one in front of the machine and one in the back. These systems are, however, not able to support two independent monitors. The two video connectors will generate identical outputs with the same timing and resolution. Two connectors on these machines are provided only for the convenience of hookup.

The graphics driver will use the monitor's EDID information via the DDC interface if the monitor provides it. The EDID information tells the graphics device which timings the monitor supports.

If EDID information from the monitor is not available, the driver will provide the following default resolutions and timings. Use caution when selecting an "X default" timing. You should only use those that you know the monitor will support, otherwise a black or unreadable screen may result. The setmon or SAM/SMH utilities for changing monitor resolution will tell you if a particular timing was obtained via DDC and is thus supported by the monitor, or if it is an "X default" timing.

Table 2-8 Supported Monitor Options

Resolution (HxV)	Frequency (Hz)	Recommended usage
1024x768	60	Flat panel
1024x768	75	
1024x768	85	
1280x1024	60	Flat panel
1280x1024	75	
1280x1024	85	
1600x1200	60	Flat panel

Table 2-8 Supported Monitor Options *(continued)*

Resolution (HxV)	Frequency (Hz)	Recommended usage
1600x1200	75	
1600x1200	85	
1920x1200	60	Flat panel
1920x1200	75	
1920x1200	85	

Framebuffer depth and colormaps

Depth 24 is the recommended depth. The benefit is that with the TrueColor visual, applications have a large 16M color palette available. Because the TrueColor colormap is read only, it can be shared among multiple applications without any color flashing problems.

The depth 8 framebuffer configuration is mainly intended for backwards compatibility. Some older applications, or applications that want to use colormap tricks may insist on using the PseudoColor visual. The PseudoColor colormap is read/write and has 256 colors. Each application can have a private colormap if requested, or multiple applications can share the same default colormap.

An issue with the PseudoColor visual is that there is only one hardware colormap. If there are multiple applications running, each with its own private logical colormap, they have to share the single hardware colormap. Only one application, the one that has the focus, has the correct colors. Others do not. You may observe color “flashing” when you move focus between different windows. If PseudoColor applications can share the same default colormap without running out of colors, the color flashing does not occur.

ATI Radeon family configuration hints

hardware compatibility

The ATI Radeon family graphics are supported on HP Integrity entry class servers and HP 9000 entry class servers.

compatibility table with other display adapters

The following table illustrates differences between the HP Visualize-FXE graphics devices and the OEM graphics devices.

Feature	Visualize-FXE	Radeon Family
Overlay Planes	8	No
Overlay LUTs	2	No
Image Planes	8 and 24	8 or 24
Image LUTs	2	1
Visuals	PseudoColor, DirectColor, and TrueColor	PseudoColor or TrueColor
VGA video output	Yes	Yes
DVI-I video output	No	Yes (Radeon 7000 and Radeon 7500 only)
Maximum resolution	1600x1200	1920x1200 (1600x1200 on ES1000)

miscellaneous

fonts

The X Server can obtain fonts from directories or font servers. Setting up a font server or making a directory a font directory is beyond the scope of this document. The font path can be loaded via the –fp option from the command line or from the XF86Config file. The latter is the preferred method. The default font path is: /usr/lib/X11/fonts/misc. See section "XF86Config File: Files section" regarding the FontPath. The following font directories are delivered with the system and may be added to the font path. Applications may install their own fonts. The application font path can be added to the FontPath as necessary.

```
/usr/lib/X11/fonts/misc/
/usr/lib/X11/fonts/hp_kana8/
/usr/lib/X11/fonts/hp_roman8/75dpi/
/usr/lib/X11/fonts/iso_8859.1/100dpi/
/usr/lib/X11/fonts/iso_8859.1/75dpi/
/usr/lib/X11/fonts/hp_chinese_s/75dpi/
/usr/lib/X11/fonts/hp_chinese_t/75dpi/
/usr/lib/X11/fonts/hp_korean/75dpi/
/usr/lib/X11/fonts/hp_japanese/100dpi/
/usr/lib/X11/fonts/iso_8859.2/75dpi/
/usr/lib/X11/fonts/iso_8859.5/75dpi/
/usr/lib/X11/fonts/iso_8859.6/75dpi/
/usr/lib/X11/fonts/iso_8859.7/75dpi/
```

```
/usr/lib/X11/fonts/iso_8859.8/75dpi/
/usr/lib/X11/fonts/iso_8859.9/75dpi/
/usr/lib/X11/fonts/iso_8859.15/75dpi/
```

files

The X Server makes use of various files on the system during normal operation. This section lists the default location of the files and gives a brief description of what they do.

File	Description
/etc/X11/XF86Config	The configuration file. Xf86 uses this file to configure itself during initialization.
/etc/X11/rgb.*	The color database.
/etc/Xn.hosts	Initial access control list for display n.
/var/X11/Xserver/logs/Xf86.n.log	The log file, where n is the display number.

3 configuring X server on HP-UX (HP Visualize-FXE)

This chapter documents information specific to the HP X Server. It describes features that are unique to HP's X Server, provides information on how to configure the X Server and includes a list of supported configurations.

Information specific to a new release of the X Server, beyond the scope of the general information in this document, can be found in the HP-UX Release Notes located in `/usr/share/doc`.

NOTE: This chapter deals with configuration information for HP Visualize-FXE graphics cards ONLY. For configuration information for OEM graphics cards, see Chapter 2 in this document.

X Server configuration

Configuration of the X Server is supported through SAM (HP-UX 11.23 and earlier) or SMH (HP-UX 11.31 and later) via a utility titled "X Server Configuration." This utility resides under the "Display" panel.

There are several X*screens files used to configure the operation of the X Server. The SAM/SMH graphical user interface for X Server configuration is provided to simplify and facilitate ease of use. While it is still possible to modify these files manually (see below), using the SAM/SMH interface greatly simplifies the process for creating Multi-Display and Single Logical Screen configurations.

The SAM/SMH component has the following actions:

- Configure Print Server...
- Configure How X Starts...
- Modify Multi-Screen Layout...
- Single Logical Screen (SLS) ->
- Modify Default Visual...

- Describe Screen...
- Identify Screen
- Modify Screen Options...
- Modify Server Options...

- Add Screen to Configuration
- Remove Screen from Configuration

The first group of Actions menus can be thought of as “global” actions. They will typically be active regardless of what has been selected. If any of these menu items are not visible it is because they are not supported under the current configuration.

The Configure Print Server item allows you to manage print servers. From this menu item you can create, stop or remove print servers.

On systems that contain a mix of HP Visualize and OEM graphics cards, the Configure How X Starts item allows you to choose on which graphics devices the X Server should start. From this action, you can assign which of your configuration files to use as a display connection - the X* screens file for HP Visualize graphics cards or the XF86Config file with OEM graphics cards. Running independent X Servers on an HP Visualize graphics device and an OEM graphics device simultaneously is not supported.

If you have multiple HP Visualize graphics display devices you can configure the X server to have multiple independent screens. Use the Modify Multi-Screen Layout menu item to do this.

To configure a multi-screen layout, first select the devices you want to include. In the X Server Configuration panel click the mouse on the first device, and **[Ctrl]**-click on the others. All the devices you want to combine into a multi-screen configuration should now be highlighted. Then use the Modify Multi-Screen Layout menu item in the Actions menu to finish the configuration.

SLS is a mechanism for treating homogeneous multi-display configurations as a single logical screen. This allows the moving/spanning of windows across multiple physical monitors. The word homogeneous is included because SLS only works if the graphics devices included in the SLS Configuration are of the same type.

Enabling an SLS configuration is similar to enabling a multi-screen layout. First select the devices you want to combine into an SLS configuration. In the X Server Configuration panel click the mouse on the first device, and **[Ctrl]**-click on the others. All the devices you want to combine into an multi-screen configuration should now be highlighted. Then from the "Actions" menu, choose the Single Logical Screen (SLS) -> Create SLS... menu item to finish the configuration. The "X Server Configuration" window should now show a single icon denoting an SLS confguration.

The Modify Default Visual menu item lets you set the default visuals and framebuffer depth on a graphics device.

The second group of “Actions” menus can be thought of as screen actions. They will be activated depending on which screens have been chosen.

The Describe Screen and Identify Screen menu selections provide information about the device. Identify Screen flashes the monitor that is connected to the graphics device.

The Modify Screen Options item contains options that are specific to each graphics device. This list might be different for cards depending on the capabilities of each card.

Specific X server options can be set with the Modify Server Options menu item. See the item for information on specific options.

Grayed out screen icons represent screens that have not been configured for use by the X Server. You can select these grayed out icons and choose the Add Screen to Configuration menu item to add screens to the X*screens.

using setmon to configure the monitor

setmon is a configuration tool used to change the monitor settings for a monitor attached to a graphics device. This tool permits you to change the monitor's refresh rate and frame buffer resolution. To change the monitor setting, the setmon command can be executed directly or done through SAM/SMH.

The setmon executable is located at /opt/graphics/common/bin/setmon. Under SAM/SMH this component is located under the top-level “Display” panel, under “Monitor Configuration”.

NOTE: Changing the monitor type while the X Server is running will necessitate killing and restarting the X Server. In order to change the monitor settings, the X Server needs to be running on the device specified.

X*screens file

For manual changes, please refer to the configuration files in the /etc/X11/ directory. Three files of particular interest are the X0screens, X0devices, and X0pointerkeys files.

description of the X*screens configuration file

This file belongs in /etc/X11/X*screens, where “*” is the display number of the server. For example, the “X0screens” file is used when the $DISPLAY environment variable is set to hostname:0.screen and the server is invoked using the “:0” option.

The X*screens file is used to specify:

- Device-independent server options, and
- For each screen:
  - what device file to use (required),
  - the default visual,
  - monitor size, and
  - device-dependent screen options.

Note that all of the items above, except for device-independent server options, are specified on a per-screen basis.

The X Server supports up to four screens at a time. Specifying more than four screens will cause a server error message.

syntax guidelines

- Blank lines and comments (text following "#") are ignored. Entries can occupy more than a single line.
- All symbols in the file are recognized case-insensitive.

the X*screens file format

Items must appear in the X*screens file in the order that they are specified below.

```
 [ServerOptions
   <server_option>
   .
   .
   .
   <server_option>]
{Screen <device_name>} ||
{SingleLogicalScreen <nRows> <nCols>
   <device_name1> . . .<device_nameN>}
      [DefaultVisual
         [Class <visual_class>]
         [Depth <depth>]
         [Layer <layer>]
         [Transparent]]
       [MonitorSize <diagonal_length><units>]
       [MinimumMonitorPowerSaveLevel <level>]
       [ScreenOptions
              <screen_option>
                .
                .
                .                <screen_option>]
```

Brackets ("[" and "]") denote optional items. Italicized items in angle brackets ("<" and ">") denote values to be specified. The double vertical line ("||") denotes that one of the ored values (items surrounded by braces, "{" and "}") must be included.

The block from the "Screen <device_name>" line to the final "<screen_option>" line is referred to as a either a "Screen Entry" or as a "Single Logical Screen entry". As shown above, the X*screens format is composed of an optional block specifying device-independent server options followed by one or more either Screen or Single Logical Screen entries (maximum of four graphics devices).The minimum X*screens file is a line with the keyword "Screen" followed by a screen device file. For example: Screen /dev/crt

server options

For more information about server options, or about additional server options, look in an information file (for example, /usr/lib/X11/Xserver/info/screens/hp).

GraphicsSharedMemorySize <memory_size>

Specify the size of the graphics shared memory region. The size must be specified in bytes and must be in hexadecimal.

Default value: 0x580000

ImmediateLoadDles

The X Server delays loading of some X extensions until the first protocol request to the given extension is received. Specifying this server option forces all extensions to be loaded at X Server startup.

screen entries

The minimum screen entry is a line with the keyword “Screen” followed by a screen device file.

Optional specifications for default visual, monitor size, and device-dependent screen options may follow this minimal screen description line.

Default Visual

This optional part of the format specifies the default visual that the screen uses. Valid keywords following the “DefaultVisual” keyword are “Class”, “Depth”, “Layer”, and “Transparent”.

If no default visual is specified, then the standard default visual class, depth, layer, and transparency for the graphics device is used.

If there is an error in a specification, look in an information file for more details (for example, /usr/lib/X11/Xserver/info/screens/hp), in case it is newer than the document you’re now reading.

Class <PseudoColor> | <DirectColor>| <TrueColor>

Specify the class of the default visual.

Depth <depth_value>

Specify the depth of the default visual (for example 8 or 24).

Layer <Image> | <Overlay>

Specify the layer of the default visual.

Transparent

Specify that a visual with an application-accessible transparent entry in the default colormap be used.

MonitorSize <diagonal_length> Inches | MM

Specify the diagonal size of the monitor. After the “MonitorSize” keyword, you must specify the diagonal length of the monitor and then the units. Use this entry only if you are using a non-standard monitor.

ScreenOptions

Screen options are device-dependent options that are documented in a file in the X Server information directory (for example, /usr/lib/X11/Xserver/info/screens/hp).

sample X*screens files

Below are several sample X*screens files that illustrate the new format.

- This is the minimum legal X*screens file, the “Screen” keyword followed by the screen device. Since no other information is given, the X Server will assume default values for other options and settings.

  Screen /dev/crt

  Figure 3-1 Results of minimal legal X*screens file

  

- This sample X*screens file could be used on a system using HP VISUALIZE-FXE with a 17-inch monitor. In this example, the GraphicsSharedMemorySize is decreased to 1 Mbyte in order to reduce the swap space requirements of the system. Decreasing GraphicsSharedMemorySize is appropriate when you do not intend to run any 3D graphics applications.

  ```
  ServerOptions
    GraphicsSharedMemorySize 0x100000
  Screen /dev/crt
    MonitorSize 17 inches
  ```

  The display diagram would be the same as that of the “Results of Minimal Legal X*screens File” configuration, above.

- The overlay visual is selected as the default. There are 255 overlay colormap entries available on the HP VISUALIZE-FXE. The 256th entry is hard-wired to transparent. Having less than 256 colormap entries should not cause a problem for most applications, but for those applications that require 256 colormap entries, the CountTransparentInOverlayVisual screen option should be used as shown below. Note that any attempts to modify the 256th entry will have no effect on the colormap.

  ```
  Screen /dev/crt
    ScreenOptions
     CountTransparentInOverlayVisual
  ```

  The display diagram would be the same as that of the “Results of Minimal Legal X*screens File” configuration, above.

- The default visual on the HP VISUALIZE-FXE is the opaque overlay visual. All 256 colormap entries are opaque and allocable. If an application requires transparency in the default visual, the “Transparent” keyword can be used to select the transparent overlay visual as shown below.

```
Screen /dev/crt
DefaultVisual
Transparent
```

The display diagram would be the same as that of the "Results of Minimal Legal X*screens File" configuration, above.

miscellaneous topics

double buffer extensions

DBE is an extension to the X Server that provides a double-buffering Application Programming Interface (API). For more information about DBE and the API, consult the DBE man pages:

```
DBE
XdbeQueryExtension
XdbeGetVisualInfo
XdbeFreeVisualInfo
XdbeAllocateBackBufferName
XdbeDeallocateBackBufferName
XdbeSwapBuffers
XdbeBeginIdiom
XdbeEndIdiom
XdbeGetBackBufferAttributes
```

performing buffer swaps on vertical blank

For performance reasons, the default DBE behavior is to not synchronize buffer swaps with the monitor's vertical retrace period. In some instances, therefore, image tearing (seeing part of the old image and part of the new image on the display at the same time) could be visible while swapping large DBE windows. For those instances where tearing would occur and is undesirable, an optional X Server mode is available to allow for synchronization of buffer swaps with vertical retrace. To activate this optional X Server mode, set the following screen option in the X*screens File before the X Server is started:

SwapBuffersOnVBlank

determining swap performance

The DBE API does not allow users to determine if double-buffering in a visual is through software or hardware. However, the API does provide a way to determine relative swapping performance on a per-visual basis. The XdbeScreenVisualInfo() function returns information about the swapping performance levels for the double-buffering visuals on a display. A visual with a higher performance level is likely to have better double-buffer graphics performance than a visual with a lower performance level. Nothing can be deduced from any of the following: the magnitude of the difference of two performance levels, a performance level in isolation, or comparing performance levels from different servers.

For more information, refer to the DBE man page on XdbeScreenVisualInfo().

display power management signaling (DPMS)

Monitors constitute a large percentage of the power used by a workstation even when not actively in use (i.e., during screen blanking). In order to reduce the power consumption, the Video Electronic Standards Association (VESA) has defined a Display Power Management Signaling (DPMS) standard which can be used to greatly reduce the amount of power being used by a monitor during screen blanking.

The following table is a description of the states that are defined by VESA. The Power Savings column indicates (roughly) the level of power savings achieved in the given state. The Recovery Time is the amount of time that the screen takes to return to a usable state when the screen saver is turned off (by pressing a key or the moving the mouse).

Table 3-1 Power saving states defined by VESA

Level	State	Power Savings	Recovery Time
0	On	None	None
1	Stand-by	Minimal	Short
2	Suspend	Substantial	Longer
3	Off	Maximum	System Dependent

The actual amount of power saved and the recovery time for each of the states is monitor-dependent and may vary widely. The customer can compensate for this by choosing an appropriate level for the monitor that is currently in use.

By default, the DPMS level used is the 0 or On (i.e. no power savings).

DPMS should not be confused with screen blanking or screen saving. Screen blanking will merely cause the screen to go dark. Screen saving will display a changing pattern on the screen to avoid the burn-in of a static image. Neither of these will cause any power saving to occur.

shared memory extension (MIT_SHM)

The MIT shared memory extension provides both shared-memory XImages and shared-memory pixmaps based on the SYSV shared memory primitives.

Shared memory XImages are essentially a version of the XImage interface where the actual image data is stored in a shared memory segment, and thus need not be moved through the Xlib interprocess communication channel. For large images, use of this facility can result in increased performance.

Shared memory pixmaps are a similar concept implemented for the pixmap interface. Shared memory pixmaps are two-dimensional arrays of pixels in a format specified by the X Server, where the pixmap data is stored in the shared memory segment. In

all other respects, shared memory pixmaps behave the same as ordinary pixmaps and can be modified by the usual Xlib routines. In addition, it is possible to change the contents of these pixmaps directly without the use of Xlib routines merely by modifying the pixmap data.

hp Visualize-FXE device-dependent information

This section includes information on the HP VISUALIZE-FXE graphics devices.

The HP VISUALIZE-FXE has 8 overlay planes, 48 image planes a 24-bit z buffer and 4 hardware colormaps.

HP VISUALIZE-FXE graphics devices contain 2D hardware acceleration as well as 3D acceleration for lighting, shading and texture mapping.

supported visuals

HP VISUALIZE-FXE graphics devices support all of the following visuals:

- Class PseudoColor Depth 8 Layer Image
- Class PseudoColor Depth 8 Layer Overlay
- Class PseudoColor Depth8 Layer Overlay Transparent
- Class DirectColor Depth 24 Layer Image
- Class TrueColor Depth 24 Layer Image

The following visuals are enabled by default on the HP VISUALIZE-FXE:

- Class PseudoColor Depth 8 Layer Image

  supports DBE hardware double-buffering
- Class PseudoColor Depth 8 Layer Overlay

  supports DBE software double-buffering
- Class PseudoColor Depth 8 Layer Overlay Transparent supports DBE software double-buffering
- Class DirectColor Depth 24 Layer Image

  does not support DBE hardware or software double-buffering
- Class TrueColor Depth 24 Layer Image

  does not support DBE hardware or software double-buffering

NOTE: When running xdpyinfo or calling the XGetVisualInfo() Xlib function, some extra duplicate visuals may appear in the visual list. These extra visuals are created on behalf of the OpenGL extension to X (GLX). If necessary, the extra visuals can be disabled using the DisableGLxVisuals screen option. See the "Disabling the GLX Visuals" section for more information.

supported screen options

The following screen options are supported:

- CountTransparentInOverlayVisual
- ImageTextViaBitMap
- EnableIncludeInferiorsFix
- DisableGlxVisuals

hp VISUALIZE-FXE configuration hints

overlay visuals and overlay transparency

HP VISUALIZE-FXE devices have two visuals in the overlay planes, both depth-8 PseudoColor. The first (default) overlay visual has 256 entries per colormap and no transparency. The second overlay visual has 255 entries per colormap and supports transparency.

To allow applications to determine which visuals are in the overlay planes, both overlay visuals are listed in the SERVER_OVERLAY_VISUALS property attached to the root window. The default overlay visual has a transparent type of "0" (None), while the transparent overlay visual has a transparent type of "1" (TransparentPixel).

If you need an overlay colormap that supports transparency, create the colormap using the visual that has transparency in its SERVER_OVERLAY_VISUALS property.

disabling the GLX visuals

The HP VISUALIZE-FXE products support the OpenGL extension to X (GLX). If HP OpenGL is installed on an HP VISUALIZE-FXE system, then the GLX extension offers new entry points for obtaining more information about X visuals. As part of offering extended visual information, some extra X visuals appear in the X visual list. The extra visuals are simply duplicates of visuals that would normally appear in the X visual list. In case that the extra visuals cause problems with applications, a screen option can be used to disable them.

To disable the GLX visuals, add the DisableGlxVisualsScreen Option to the X*screens file.For example:

```
Screen /dev/crt/
    ScreenOption
       DisableGlxVisuals
```

hp VISUALIZE-FXE colormaps

HP VISUALIZE-FXE devices have a total of 4 hardware colormaps. 2 of the colormaps are dedicated to the overlay planes. The remaining 2 colormaps are dedicated to the image planes.

Of the two overlay colormaps, one is permanently reserved for the default colormap. The other overlay colormap is available to applications.

changing the monitor type

A configuration tool is available to change the monitor type on HP VISUALIZE-FXE devices. This tool permits users to change the monitor's refresh rate, frame buffer resolution, and frame buffer memory configuration (e.g., Stereo, Double Buffer), when the device supports multiple options. To change the monitor setting, the setmon command can be executed directly or done through the SAM/SMH system administration tool.

The setmon executable is located at /opt/graphics/common/bin/setmon. Under SAM/SMH this component is located under the top-level "Display" panel, under "Monitor Configuration".

NOTE: Changing the monitor type while the X Server is running will necessitate killing and restarting the X Server.

4 X Server configuration details

This chapter discusses several details concerning the configuration of X hosts, mouse and keyboard.

making an X*.hosts file

The /etc/X0.hosts file is an ASCII text file containing the hostnames of each remote host permitted to access your local server.

- If you are running as a stand-alone system, you must have your system's name in this file.
- If you are part of a network, the other system names must be included.

The syntax is as follows:

```
<host>
<host>
<host>
```

For example, if you are hpaaaaa, and regularly ran clients on hpccccc, and hpdddddd, you would want the following lines.

```
hpaaaaa
hpccccc
hpdddddd
```

Note that aliases work as well as hostnames, provided they are valid, that is, commonly known across the network.

using an /etc/hosts file

This file need not be present if your system is configured to query a nameserver.

The /etc/hosts file is an ASCII text file containing a list of all the host names and internet addresses known to your system, including your own system.

If your system is not connected to a network, use the loopback address (127.0.0.1) and the hostname unknown: 127.0.0.1 unknown

For a local system to access a remote host:

- The address and hostname of the remote host must be listed in the local system's /etc/hosts file.
- The user must have a valid login (username and password) and home directory on the remote host.

customizing the mouse and keyboard

This section describes various customizations that can modify the default keyboard and mouse behavior.

changing mouse button actions

The xmodmap utility can be used to change mouse button mappings. The syntax for changing mouse button mappings with xmodmap is:

xmodmap {-e “pointer = {default | number [number...]}” | -pp}

where:

-e

Specifies a remapping expression.

default

Set mouse keys back to default bindings.

number

Specifies a list of button numbers to map the mouse keys to. The order of the numbers refers to the original button mapping.

pp

Print the current pointer mapping.

For example, to reverse the positions of buttons 1 and 3 for left-handed mapping:

xmodmap -e “pointer = 3 2 1" (2-button mouse)

modifying modifier key bindings with xmodmap

To change the meaning of a particular key for a particular X11 session, or to initialize the X Server with a completely different set of key mappings, use the xmodmap client.

The syntax for xmodmap is as follows: xmodmap <options> [<filename>] where <options> are:

-display <host>:<display>

Specifies the host, display number, and screen to use.

-help

Displays a brief description of xmodmap options.

-grammar

Displays a brief description of the syntax for modification expressions.

-verbose

Prints log information as xmodmap executes.

-quiet

Turns off verbose logging. This is the default.

-n

Lists changes to key mappings without actually making those changes.

-e <expression>

Specifies a remapping expression to be executed.

-pm, -p

Prints the current modifier map to the standard output. This is the default.

-pk

Prints the current keymap table to the standard output.

-pp

Print the current pointer map to the standard output.

- (dash)

Specifies that the standard input should be used for the input file.

<filename>

Specifies a particular key mapping file to be used.

specifying key remapping expressions

Whether you remap a single key "on the fly" with a command-line entry or install an entire new keyboard map file, you must use valid expressions in your specification, one expression for each remapping.

A valid expression is any one of the following:

Table 4-1 Expression Types

To do this. . .	Use this expression. . .
Assign a key symbol to a keycode	keycode <keycode> = <keysym>
Replace a key symbol expression with another	keysym <keysym> = <keysym>
Clear all keys associated with a modifier key	clear<modifier>
Add a key symbol to a modifier	add <modifier>= <keysym>
Remove a key symbol from a modifier	remove <modifier> = <keysym>

keycode

Refers to the numerical value that uniquely identifies each key on a keyboard. Values may be in decimal, octal, or hexadecimal.

keysym

Refers to the character symbol name associated with a keycode; for example, KP_Add.

<modifier >

Specifies one of the eight modifier names: Shift, Control, Lock, Mod1, Mod2, Mod3, Mod4, and Mod5.

On Hewlett-Packard keyboards, the lock modifier is set to the **Caps Lock** key. However, any of the modifiers can be associated with any valid key symbol. Additionally, you

can associate more than one key symbol with a modifier (such as Lock = Shift_R and Lock = Shift_L), and you can associate more than one modifier with a key symbol (for example, Control = Caps_Lock and Lock = Caps_Lock).

For example, on a PC-style keyboard, you can press **D**to print a lower case "d", **Shift D**to print a capital "D", **Alt D**to print something else, and **Shift Alt D**to print still something else.

The xmodmap client gives you the power to change the meaning of any key at any time or to install a whole new key map for your keyboard.

examples

Suppose you frequently press the **Caps Lock**key at the most inopportune moments. You could remove the **Caps Lock**key from the lock modifier, swap it for the **f1** key, then map the **f1**key to the lock modifier. Do this by creating a little swapper file that contains the following lines:

```
!This file swaps the [Caps] key with the [F1] key.
remove Lock = Caps_Lock
keysym Caps_Lock = F1
keysym F1 = Caps_Lock
add Lock = Caps_Lock
```

Note the use of the ! in the file to start a comment line. To put your "swapper" file into effect, enter the following on the command line:

```
xmodmap swapper
```

If you use such a swapper file, you should probably have an unswapper file. The following file enables you to swap back to the original keyboard mapping without having to exit X11:

```
!This file unswaps the [F1] key with the [Caps Lock] key.
remove Lock = Caps_Lock
keycode 88 = F1
keycode 55 = Caps_Lock
add Lock = Caps_Lock
```

Note the use of the hexadecimal values to reinitialize the keycodes to the proper key symbols. You put your “unswapper” file into effect by entering the following command line:

xmodmap unswapper

On a larger scale, you can change your current keyboard to a Dvorak keyboard by creating a file with the appropriate keyboard mappings.

xmodmap .keymap

printing a key map

The -pk option prints a list of the key mappings for the current keyboard.

xmodmap -pk

The list contains the keycode and up to four 2-part columns. The first column contains unmodified key values, the second column contains shifted key values, the third column contains meta (**Extend Char/Alt**) key values, and the fourth column contains shifted meta key values. Each column is in two parts: hexadecimal key symbol value, and key symbol name.

stopping the X Window system

After stopping all application programs, stop the window system by holding down the **Ctrl**and **Shift**keys, and then press the **Pause/Break** key. This stops the display server, and with it the window system.