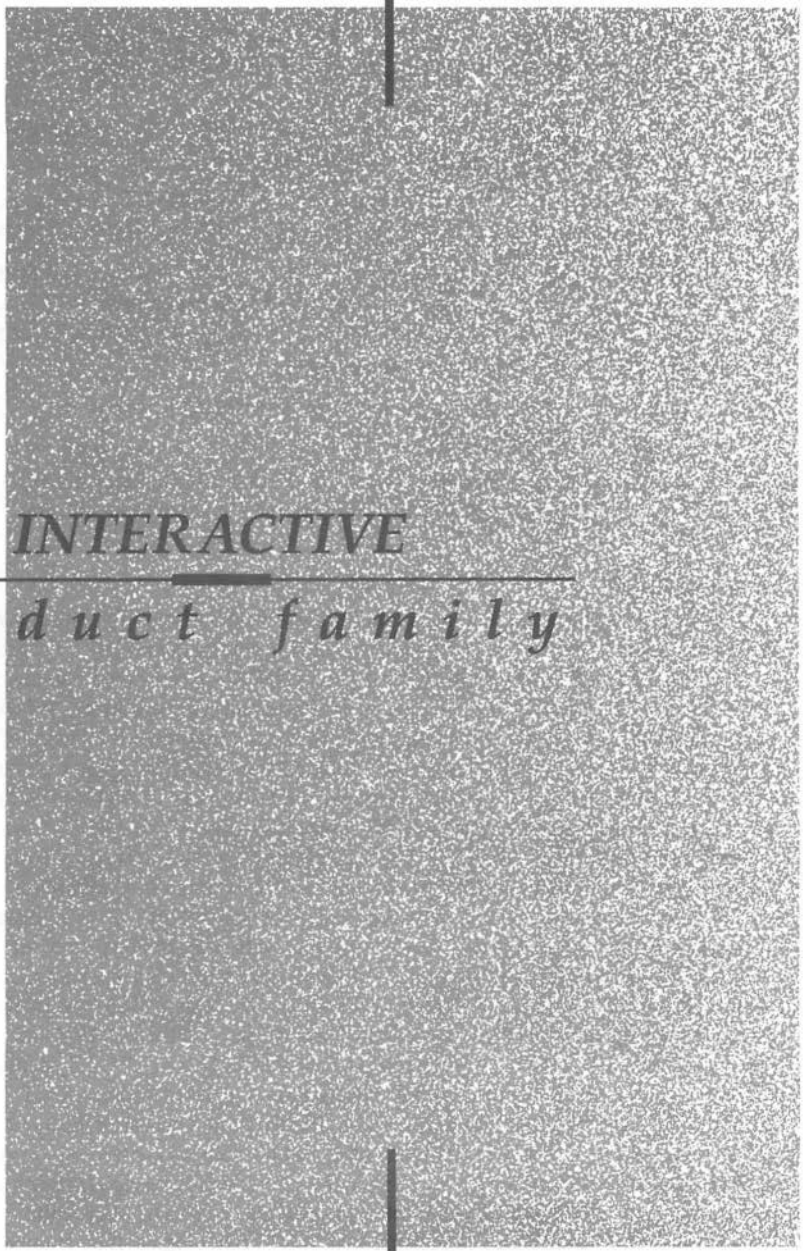
INTERACTIVE™ X11 Runtime System Guide



SunSoft
A Sun Microsystems, Inc. Business

INTERACTIVE X11
Version 2.0.1
– Update Package –

This *INTERACTIVE X11 Update Package* contains new and revised pages for the *INTERACTIVE X11 Runtime System Guide*, Version 2.0.1.

Add the following article to your *INTERACTIVE X11 Runtime System Guide*:

INTERACTIVE X11 Version 2.0.1 Release Notes

Add or replace the following manual entries in the "INTERACTIVE X11 Reference Manual:"

New Entries	**Revised Entries**
gv.sh(1)	*Xcvc*(1)
	Xgp(1)
	Xv256(1)
	Xvga(1)

INTERACTIVE X11
Runtime System Guide

CONTENTS

Introduction to INTERACTIVE X11

Welcome to the *INTERACTIVE X11 Runtime System Guide*. This guide contains the basic documentation you need to install, maintain, and use INTERACTIVE X11 Version 2.0. Whether you are an experienced programmer or a novice user, be sure to read the next few pages of this document. They will tell you what is contained in this guide and how to use it to your best advantage.

WHAT'S INCLUDED

The *INTERACTIVE X11 Runtime System Guide* includes:

- **INTERACTIVE X11 Release Notes**
  Provides a description of the current release of INTERACTIVE X11.
- **Using the INTERACTIVE Easy Windows Environment**
  Describes how to install the INTERACTIVE Easy Windows* Environment.
- **INTERACTIVE Motif Window Manager Release Notes**
  Provides a description of the current release of the INTERACTIVE Motif* Window Manager subset.
- **INTERACTIVE Motif Window Manager Installation Instructions**
  Describes the basic requirements and procedures that are necessary to install the current release of the INTERACTIVE Motif Window Manager subset.
- **Character Bitmap Distribution Format**
  Describes Adobe* System's character bitmap distribution format (BDF). Information on the form of a font bitmap description file is provided.
- **INTERACTIVE X11 Installation Instructions and Maintenance Procedures**
  Provides step-by-step instructions for installing INTERACTIVE X11 and discusses the component packages of INTERACTIVE X11. It describes how to perform the preliminary setup procedures that are required to use the system once it is installed. This document also discusses building new servers, removing servers, making default servers, modifying configuration

information, installing new X device drivers, and adding and deleting X users. It includes information on running the server and removing X packages from the system, and discusses the kernel facilities required to run X.

- **INTERACTIVE X11 Reference Manual**
  Includes INTERACTIVE's proprietary server and driver manual entries, as well as entries applicable to X11, Release 4 client programs.

- **Reader's Comment Form**
  Provides you with a way to tell us what you like or dislike about this guide and to send us your ideas for making it even better.

Other documentation supplied with the INTERACTIVE X11 Runtime System:

- *O'Reilly & Associates X Window System User's Guide, Motif Edition*
  Describes window system concepts and features, including display servers, client application programs, window managers, and the `xterm` terminal emulator.

WHERE TO BEGIN

The *INTERACTIVE X11 Runtime System Guide* includes documents for users at all levels of expertise. Depending on your experience, you may want to use this guide in a number of different ways. The following outline provides some suggested ways to use this guide:

- **If you are a beginner . . .**
  First, read this document to get to know what INTERACTIVE X11 is. Then, read section 1 of the *X Window System User's Guide*, which is supplied as a supplement to this guide.

- **If you are an experienced UNIX* System user . . .**
  Read the "INTERACTIVE X11 Release Notes" for information on INTERACTIVE X11 Version 2.0. For more detailed and technical information about the X Window System*, refer to the *X Window System User's Guide*.

- **If you are installing and maintaining the system . . .**
  Read and follow the steps outlined in "INTERACTIVE X11 Installation Instructions and Maintenance Procedures."

- **If you want the latest system information . . .**
  Read the "INTERACTIVE X11 Release Notes" which will provide you with up-to-the-minute information on Version 2.0 of INTERACTIVE X11.

The documentation included in this guide provides information about how to install, use, and maintain INTERACTIVE X11. This guide is intended for users who will be running the INTERACTIVE X11 Runtime System.

If you plan to develop X applications, refer to the *INTERACTIVE X11 Development System Guide*. This guide is supplied with the INTERACTIVE X11 Development System and includes:

- **Inter-Client Communication Conventions Manual**
  Provides conventions that allow clients to cooperate in the areas of selections, cut buffers, window management, session management, and resources. This document is reprinted from the X Consortium document of the same name.
- **INTERACTIVE TCP/IP Programmer's Supplement**
  Presents supplemental information about how to program the USL Transport Layer Interface (also referred to as TLI or the Transport Interface) and the Berkeley Software Distribution (BSD) socket interface.

Other documents supplied with the INTERACTIVE X11 Development System are the:

- *O'Reilly & Associates Xlib Programming Manual*
  Provides information about the X library, the C language programming interface of the X Window System. It includes a conceptual introduction, tutorial material, and programming examples.
- *O'Reilly & Associates Xlib Reference Manual*
  Contains the manual entries for the X library.
- *O'Reilly & Associates X Toolkit Intrinsics Programming Manual, Motif Edition*
  Describes how to use the X Toolkit routines.
- *O'Reilly & Associates X Toolkit Intrinsics Reference Manual*
  Contains reference pages for the X Toolkit functions.

OVERVIEW OF INTERACTIVE X11

INTERACTIVE X11 is a network-based graphics system. It is based on X11 Release 4 of the X Window System that was developed at MIT. X11 has been adopted as the industry-standard windowing system. INTERACTIVE X11 follows a client-server model. In a client-server model, the server manages the graphical output and the user input. The clients are application programs that perform specific tasks. Clients communicate with the server, accepting user input and sending graphical output commands to it. This division permits the clients and the display server to work together on the same system or to be separated across a network.

INTERACTIVE X11 features include:

- Network-transparent access to the display
- High-performance, high-level, device-independent graphics
- Hierarchical, resizable, overlapping windows

INTERACTIVE X11 is divided into two system packages: the Runtime System and the Development System.

RUNTIME SYSTEM

The INTERACTIVE X11 Runtime System contains the software necessary to execute X11 applications on INTERACTIVE UNIX System V/386 Release 3.2, and the documentation required by users and system administrators.

Network Transparency

INTERACTIVE X11 is a complete implementation of X11 built on the INTERACTIVE UNIX System. It uses STREAMS-based Interprocess Communication (IPC) mechanisms to support a networked client-to-server link that operates across networks. INTERACTIVE X11 uses INTERACTIVE TCP/IP (Transmission Control Protocol/Internet Protocol) to communicate with X clients and servers on any machine in a network. INTERACTIVE X11 can also be installed on a system without networking software, in which case it automatically uses a local link between clients and the X server.

Peripheral Support

The supported graphical displays include:

- A variety of EGA/VGA boards that are register-compatible with IBM* EGA/VGA video controllers
- 256-color VGA boards
- 8514/A Graphics Processor (and register-compatible) boards
- The TIGA* 34010/34020 server supports the following boards:
  - COMPAQ* AG1024
  - Desktop Computing AGA 1024*
  - IMAgraph* TI-1210 series
  - MegaScan FDP-6120
  - Number Nine Computer Corporation* PEPPER* PRO1024ISA* and PEPPER PRO1280*
  - Renaissance Rendition* II
  - Spectre* SP200*
  - Texas Instruments TMS34010 Software Development Board
- Cornerstone Technology Incorporated* – supporting 150dpi monitors, DualPage*, and SinglePage XL*
- Hercules* monochrome graphics card
- Moniterm 21/91 Viking server
- Pixelworks Clipper Graphics Series from Pixelworks, Inc.
- Sigma Designs LaserView* Plus
- Bell Technologies Workstation Graphics Engine (Blit Express)

The supported mice include:

- LOGITECH* Three-Button Serial and Bus Mice, and MouseMan* Serial and Bus Mice
- Microsoft* Two-Button Serial and Bus Mice
- MSC Technologies (formerly Mouse Systems Corporation) OMNIMOUSE*, Serial, and Bus Mice

- Micro Channel* Architecture (PS/2*) and COMPAQ on-board mice and compatibles

The supported keyboards include:

- 101-key IBM AT*-style keyboard
- 84-key IBM PC-style keyboard

Performance

For boards without hardware graphics-drawing capabilities, INTERACTIVE has optimized the MIT frame-buffer code to enhance performance. For higher-end displays, INTERACTIVE takes full advantage of the hardware features provided in intelligent graphics controllers. INTERACTIVE also works with controller manufacturers to assist them in tuning their microcode to optimize performance when using X11.

DEVELOPMENT SYSTEM

The INTERACTIVE X11 Development System consists of the software and programmer's documentation necessary to create X11 applications. It is intended primarily for application developers and sophisticated end users.

Berkeley Facilities

The INTERACTIVE X11 Development System includes a Berkeley 4.3 (4.3BSD) compatible socket library to permit easy porting of existing Berkeley-based X applications.

FEATURES AND ENHANCEMENTS

INTERACTIVE X11 contains many performance and functional enhancements. Some of the features are:

- Optimized input code
- Local connections use STREAMS pipes, eliminating the need for INTERACTIVE TCP/IP for local connections
- Pseudo-color support – the user can set the color desired, where applicable

- Fonts can be downloaded to the board on displays that provide this feature – downloading fonts increases performance dramatically
- Graphics enhancements include:
  - Multiple clipping rectangle support
  - Rectangles as a special type of polygon support
  - Line segment support
  - Odd-even and winding fill rule support
  - X11R4 integer arc code resulting in speedups of up to 1000 times faster
  - X11R4 integer-based wide line code
- Shared `Xlib`, which reduces the sizes of the clients and permits faster loading of clients
- Text-based MS-DOS* (DOS) applications can be run using the `xpcterm` client developed and supplied by INTERACTIVE
- `vpix` and X co-exist with all the supported mouse devices on the EGA and VGA displays – for DOS graphics applications, `vpix` can be run on one virtual terminal and X on another virtual terminal, allowing the user to VT flip between them

DOCUMENTATION REFERENCES

Throughout this guide, the following full documentation titles will be referenced in shortened versions as follows:

Full Title	*Shortened Version*
INTERACTIVE UNIX System V/386 Release 3.2 Operating System Guide	INTERACTIVE UNIX Operating System Guide
INTERACTIVE UNIX System V/386 Release 3.2 Guide for New Users	INTERACTIVE UNIX Guide for New Users
INTERACTIVE UNIX System V/386 Release 3.2 User's/System Administrator's Reference Manual	INTERACTIVE UNIX System User's/System Administrator's Reference Manual
INTERACTIVE Software Development System Guide and Programmer's Reference Manual	INTERACTIVE SDS Guide and Programmer's Reference Manual

FOR MORE INFORMATION

INTERACTIVE X11 is a part of the INTERACTIVE Product Family, and it is supported by a complete set of documentation. For a complete listing of all INTERACTIVE UNIX System-related documentation, refer to the "Documentation Roadmap" in the *INTERACTIVE UNIX Operating System Guide*.

INTERACTIVE X11
Version 2.0.1
Release Notes
August 1992

1. INTRODUCTION

INTERACTIVE X11 Version 2.0.1 for the INTERACTIVE UNIX® Operating System incorporates a number of improvements into the INTERACTIVE X11 product. These release notes supplement the Version 2.0 release notes and describe only the differences between Version 2.0 and Version 2.0.1.

2. NEW FEATURES IN VERSION 2.0.1

There are several new features that are part of this release, including:

- Support for the Super VGA Protected Mode Interface (SVPMI)
- Support for several new VGA boards
- A new version of `xman`
- Enhancements to `xterm` and `xpcterm`

These enhancements are described below.

2.1 SVPMI Support

The Video Electronics Standards Association (VESA) has published a document describing a text file format that can be used to illustrate the characteristics of a Super VGA board. The capacity for interpreting such a file has been included as part of INTERACTIVE X11 for the following:

— VGA

— v256

— gp (8514)

This makes it possible to support future Super VGA boards without requiring another release of INTERACTIVE X11. A menu item (SVPMI) has been added to the `xconfig` software. When selected, it displays all of the video adapters supported through Super VGA `pmi` files resident on the system. These files have the format *name*`.pmi` and are located in

`/usr/lib/X11/vesa/`*server*, where *server* is one of the three servers listed above. Third-party `pmi` files must be installed in this directory. See also *gv.sh*(1).

2.2 Support for New VGA Boards

With this release, `pmi` files are supplied for the following boards:

For the VGA server:

ATI® VGA Wonder+
Genoa 6000 VGA
Genoa 7000 VGA
Orchid Prodesigner II VGA
Western Digital 90C11 VGA
Western Digital 90C30 VGA

For the v256 server:

ATI VGA Wonder+
COMPAQ® 486C
Genoa GVGA
Genoa 7000 VGA
Orchid Prodesigner II VGA
Western Digital WD90C11 VGA
Western Digital WD90C30 VGA

For the 8514 server:

ATI 8514A

2.3 New Version of `xman`

This release contains a new version of the `xman` client that supports manual entries that are packed (extension `.z`) as well as entries that are compressed (extension `.Z`).

2.4 Changes to `xterm/xpcterm`

`xterm` and `xpcterm` have been enhanced to support the programming of keys with all modifiers, including CTRL. Use `xmodmap` to change the meaning of a key. The same software change also causes the CAPS-LOCK key to work as expected.

INTERACTIVE X11
Version 2.0
Release Notes

CONTENTS

INTERACTIVE X11
Version 2.0
Release Notes
October 1991

1. INTRODUCTION

INTERACTIVE X11 Version 2.0 for the INTERACTIVE UNIX* Operating System is based on Version 11, Release 4 of the X Window System* from MIT. Version 2.0 is a complete release of INTERACTIVE X11. It introduces a number of improvements, including reduced memory usage and performance enhancements. This document describes the new features, hardware requirements, special installation considerations, and potential problems you may encounter with this release. This version runs on the INTERACTIVE UNIX System Version 2.2 or later.

2. DOCUMENTATION NOTES

X11 manual entries that are referenced in the text but not included in this manual can be found in the O'Reilly & Associates *X Window System User's Guide, Motif Edition*.

The following entries included in the O'Reilly & Associates *X Window System User's Guide* are not applicable to INTERACTIVE X11:

xcol(1)
xmh(1)

3. NEW FEATURES IN VERSION 2.0

The server-specific manual entries are those Section 1 entries in the "INTERACTIVE X11 Reference Manual" in which the entry names begin with an uppercase X. Refer to these entries to see which boards are supported and the specific resolutions that are supported for each.

Additional functionality that is part of this release:

- The `Xvga` and `Xv256` servers now support the Microlabs Ultimate VGA and VGA Solution.

- The `Xgp` server now supports boards from Adex and ATI*.
- The `Xvga`, `Xv256`, and `Xgp` servers may be configured using an external text file based on the proposed Video Electronics Standards Association (VESA) Super VGA Protected Mode Interface (SVPMI).
- The `Xcvc`, `Xlvp`, `Xhrc`, `Xgp`, `Xvga`, and `Xv256` servers have been significantly improved. They now support cursors and fill patterns of arbitrary size and offer improved performance.
- All servers now indicate the correct visual type based on their capabilities. Grayscale and monochrome displays will no longer be reported as color displays. In addition, servers supporting dynamic (changeable) colormaps can simulate static colormaps. This allows some applications to work better, because the server will return the closest color available rather than give an error if the colormap is full.

  Refer to *X*(1), *Xconfig*(5), and *Xcolors*(5) for more information on these features.
- Screen Blanking
  On displays that don't support screen blanking in hardware, the server will simulate blanking the display by drawing a black rectangle to blank the display and generating an exposure event to redisplay the screen.
- Several new clients have been added to this release including:

`xditview`	`troff` previewing program
`xman`	manual entry browser
`xfig`	drawing program (in contributed subset)
`xmail`	windowing mail interface (in contributed subset)

- When used with an INTERACTIVE display server, `xpcterm` now emits the correct scan codes regardless of the type of keyboard used. Applications that take advantage of the scan code mode of `xpcterm`, such as the VP/ix* Environment, will now function correctly when used with non-U.S. keyboards. A command-line option, `-noisc`, is provided for use with display servers that are not supplied with INTERACTVE X11.

4. HARDWARE REQUIREMENTS

4.1 Displays Supported

The displays supported in this release are:

- TIGA* 34010/34020 boards:
  - COMPAQ* AG1024 (`Xcpqag`)
  - Desktop Computing AGA 1024 (`Xdcaga`)
  - IMAgraph* TI1210 (`Xigsp`)
  - MegaScan FDP-6120 (`Xmsfdp`) (supported in dual-headed mode only)
  - Number Nine PEPPER* PRO1280* (`Xnnp`)
  - Number Nine PEPPER PRO1024ISA (`Xnnpi`)
  - Renaissance Rendition II (`Xrren2`)
  - Spectre* SP200* (`Xsp200`)
  - Texas Instruments TMS34010 Software Development Board (`Xtisdb`)
- Moniterm 21/91 Viking board (`Xviking`)
- 256-color VGA boards (`Xv256`)
- 8514/A Graphics Processor (and register-compatible) boards (`Xgp`)
- Bell Technologies Workstation Graphics Engine (Blit Express) (`Xwge`)
- Cornerstone Technology Incorporated* boards (`Xcvc`)
- EGA and VGA. All boards that are register-compatible with the IBM* EGA/VGA standards, including the SunRiver* Fiber Optic Workstations, will run on the `Xvga` server. Many boards are also supported in their high-resolution modes. Refer to *Xvga*(1) and *Xconfig*(5) for a list of the boards and the board resolutions that are currently supported.
- Hercules* monochrome graphics card and compatibles (`Xhrc`)

- Sigma Designs LaserView* Plus (`Xlvp`)
- Pixelworks Clipper Graphics Series from Pixelworks, Inc. This includes the Clipper, the Micro Clipper, and the Ultra Clipper (`Xpw`).

4.2 Mice Supported

The mice supported in this release are:

- LOGITECH*: three-button keyboard, serial, and bus mice; two-button keyboard and serial mice; MouseMan* serial and bus mice.
- Microsoft*: two-button keyboard, serial, and bus mice.
- MSC Technologies, Inc.: OMNIMOUSE*; three-button optical serial and bus mice; two-button optical serial and bus mice.
- PS/2* on-board mouse and compatibles (including the COMPAQ built-in mouse).

Xconfig(5) provides additional information on configuring mice.

If you need to temporarily run X11 without a mouse, you can use the `nullmouse` option for the mouse. However, no pointer-specific functions will be available under this configuration.

4.3 Digitizing Tablets Supported

INTERACTIVE X11 Version 2.0 supports the following tablets and compatibles:

- Summagraphics* SummaSketch* Plus, models MM961 and MM1201.
- Kurta* IS/ONE* tablets in MM961/1201 mode (switches C5 up, C6 down, C7 down, and C8 up).

Xconfig(5) provides additional information on tablets.

4.4 Recommended Minimum Hardware Configuration

The recommended minimum hardware configuration is:

- A minimum of 4 megabytes (MB) of real memory is required (6 MB are recommended). If you are running more than one server, an additional 2 MB per server is recommended.

- A minimum of 12 MB of page (swap) space on your fixed disk is recommended. If you are running more than one server, an additional 5 MB per server is recommended.
- You will need 10 to 12 MB of disk space on the `/usr` file system to load the Runtime System package. To load the entire system (Runtime System, Contributed Software, Development System, and Server Kit), you will need 20 to 22 MB of disk space on the `/usr` file system.

5. INSTALLING THE RELEASE

INTERACTIVE X11 Version 2.0 will install on the INTERACTIVE UNIX Operating System Version 2.2 or later. You can overlay INTERACTIVE X11 Version 2.0 on a previous release of X11 if one is currently installed.

You should not be in the X11 environment at the time you install (overlay) Version 2.0, because this will cause certain files that must be updated to be inaccessible.

5.1 Removing Previous Versions of INTERACTIVE X11

It is recommended that you do not remove the previous release, but overlay Version 2.0 on top instead. If you need to remove the previous release of INTERACTIVE X11, do *not* remove the System subpackage of the Runtime System. Removing this package may cause you to have to reinstall parts of your system. In lieu of removing the package, you can disable the kernel drivers that you no longer need using the `CONFIGURE KERNEL` option in `kconfig` or the `xdrivermgmt` option of `sysadm`.

5.2 Changes to the Runtime System Package

- In Version 2.0, the System and Servers subset contains only the 16 and 256 color VGA servers. Additional servers are contained in separate subsets. The installation and configuration procedures for the servers have been substantially revised. Refer to the "INTERACTIVE X11 Installation Instructions and Maintenance Procedures" for additional information.
- The color database has changed in X11 Release 4. This may result in slightly different colors appearing on screen than in previous releases. It also has caused the default behavior of Motif* to change, resulting in a different appearance of inactive window borders.

5.3 Installing INTERACTIVE X11 Version 2.0

You should refer to the "INTERACTIVE X11 Installation Instructions and Maintenance Procedures" for instructions on installing this release. In addition, the following information should be noted.

- To install INTERACTIVE X11 Version 2.0, you will need at least 0.5 MB of free space in the `/tmp` file system (or in the `root (/)` file system, if `/tmp` is not a separate file system). This is over and above the requirements specified in section 3, "HARDWARE REQUIREMENTS."
- When installing the INTERACTIVE X11 Version 2.0 Runtime System, any existing copies of INTERACTIVE X11 input drivers, INTERACTIVE X11 display drivers, fonts (miscellaneous, 75dpi, and 100dpi), microcodes, and X11-specific kernel drivers will be deleted. You should save any modifications or additions, particularly those made in the font directories, before installation.
- When installing the INTERACTIVE X11 Version 2.0 Development System, any existing copies of X11 `include` files and libraries will be deleted. You should make copies of any third-party `include` files installed in the directory `/usr/include/X11`.
- The following files will be updated when you install INTERACTIVE X11 Version 2.0 on top of a previous release of INTERACTIVE X11. If you have made changes to any of these files, you should save the modified files before installing INTERACTIVE X11 Version 2.0:
  - `/usr/lib/X11/Xconfig`
  - `/etc/X?.hosts`
  - `/usr/lib/X11/app-defaults/*`
  - `/usr/lib/X11/<resource_files>`
  - `/usr/lib/X11/fonts/*`
  - `/usr/lib/X11/xdm/*`
  - `/usr/include/X11/*`
  - X libraries in `/usr/lib`
- The format of the entries in the file `/usr/lib/X11/Xconfig` has changed for the *Xvga,*

Xv256, and *Xgp* servers. This change is reflected in the `sysadm` menus. It is recommended that you look at the sample entries provided or run the `xmgmt` option of `sysadm` to configure your system after installation.

- If you had any fonts previously installed in the system, copy them into the appropriate directories after completing the installation and run `mkfontdir` in each of those font directories. Refer to *mkfontdir*(1) for more information.
- The program `/usr/bin/X11/X` is now linked to the client `/usr/bin/X11/xfront` as part of the installation. You should not link `/usr/bin/X11/X` to a specific server. The correct server will be executed based on the server configuration information and command line arguments used. Refer to *xfront*(1) for more information.
- In order to configure and build a server and turn on the associated driver, you must run the `xmgmt` option of `sysadm`. The system will ask you if you want to use this option following the installation of the INTERACTIVE X11 System and Servers subset as well as following the installation of any additional servers.
- Due to hardware considerations, in certain cases, the configuration information you choose for a mouse might not be related to the actual brand name of the mouse you have. For examples and additional information, refer to the "INTERACTIVE X11 Installation Instructions and Maintenance Procedures" and to *Xconfig*(5).

  In order to support the mouse supplied with the ATI VGA Wonder* board, configure it as a LOGITECH two-button bus mouse. The ATI VGA mouse is *not* a Microsoft mouse as the documentation supplied by ATI states.
- The kernel tunables `NQUEUE` and `NSTREAM` are updated as part of the installation of the System and Servers subset in the INTERACTIVE X11 Runtime System package. However, it may be necessary for you to increase the settings of these variables if you run out of "STREAMS resources." To do this, increase the settings in the file `/etc/conf/cf.d/mtune` and then build a kernel.

  If you are installing only the Clients subset (and not installing the System and Servers subset), then you must update the tunables manually and build a kernel.

5.4 Obsolete Files and Directories

The following files and directories, supplied in Releases 1.0 and 1.1 of INTERACTIVE X11, are now obsolete and will be removed automatically if you are updating to X11 Version 2.0:

- `/usr/lib/X11/Xservers/mouse`
- `/usr/lib/X11/Xservers/keyboard`
- `/usr/options/x1.name`
- `/usr/options/x2.name`
- `/usr/options/xr.name`

6. CONTRIBUTED SOFTWARE

The Contributed Software is provided on an "as-is" basis. This software is unsupported, and it may not run on all configurations.

7. DEVELOPMENT SYSTEM

A new library, `libXmalloc`, which is substantially faster than the standard `malloc`, `realloc`, and `free` functions available in `libc` and `libmalloc`, has been provided.

Prior releases of INTERACTIVE X11 required client programs to be linked with the `inet` library. X11 Version 2.0 does not have this requirement, however, programs must now be linked with the `nsl_s` library. This will occur automatically when using a *makefile* generated from an *Imakefile*. To manually include this library, add `-lnsl_s` to the end of the final link line of the program. The Runtime version of `nsl_s` will be installed as part of the core base system. The library for compilation (`libnsl_s.a`) is in the STREAMS Facilities subset.

8. OPERATING SYSTEM

You may have to consider the following operating system related issues when installing and running INTERACTIVE X11:

- The keyboard mouse (COMPAQ built-in mouse) driver, the Microsoft bus mouse driver, and the LOGITECH bus mouse driver are in the Additional Drivers subset. If you are using one of these mice, install the appropriate driver from the Additional Drivers subset before you install X11.

- If you are installing INTERACTIVE X11 on the INTERACTIVE UNIX System Version 2.2 or 2.2.1 and you have more than 100 drivers installed, your system can malfunction. To prevent this, install the Kernel Configuration Update `SSU.8`. This step will not be necessary for most users.

9. NETWORKING

The following information applies to networking:

- If you have not installed INTERACTIVE TCP/IP (Transmission Control Protocol/Internet Protocol) on your system, you will get the message:

  ```
  NOTE: TCP connections are not available.
  ```

  This message states that you cannot have clients that access X across the network. X is completely functional for local connections. You will get this message if TCP/IP is not installed or if you are running *init* at level 2 (see *init*(1M) in the *INTERACTIVE UNIX System User's/System Administrator's Reference Manual*).

- This version of INTERACTIVE X11 requires INTERACTIVE TCP/IP to run TCP connections across the network. There are no such limitations for local connections.

- If you installed the INTERACTIVE TCP/IP extension after INTERACTIVE X11, you *must* reinstall INTERACTIVE TCP/IP after removing the INTERACTIVE X11 Runtime or Development systems.

- If you are running the name server on a slow machine, it may take a long time to resolve names in the `/etc/X0.hosts` file if you have several entries. This can cause `xinit` to time out before it can connect to a server.

10. VIRTUAL TERMINALS

The following information applies to virtual terminals:

- Currently, the server uses a new VT every time you run X. Hence, every X server needs a free VT that does not have an active `getty` on it. To change the number of active VTs on your system, select the `virtterm` option of the TTY MANAGEMENT (`sysadm ttymgmt`) menu.

- The server for Pixelworks displays does not implement VT flipping in single-headed mode.
- The following information pertains to VT flipping on boards supported by the TIGA 34010/34020 server.

  VT flipping is supported on the following boards:

  - COMPAQ AG1024 (`Xcpqag`)
  - Number Nine PEPPER PRO1024ISA (`Xnnpi`)
  - Renaissance Rendition II (`Xrren2`)
  - Spectre SP200 (`Xsp200`)
  - Texas Instruments TMS34010 Software Development Board (`Xtisdb`)

  VT flipping is not supported on the following boards:

  - Desktop Computing AGA 1024 (`Xdcaga`)
  - IMAgraph TI1210 (`Xigsp`)
  - MegaScan FDP-6120 (`Xmsfdp`) (supported in dual-headed mode only)
  - Number Nine PEPPER PRO1280 (`Xnnp`)
- When using INTERACTIVE X11 and the VP/ix Environment (on the console or on another VT) together and sharing a serial mouse, make sure you use the `COM1MOUSE` setting for the serial mouse in your VP/ix configuration file. If you use the `COM1` setting, VP/ix does not close the mouse device, and the X server cannot access the mouse device when you VT flip to X.

11. X SERVER

The following information applies to the X server:

- At start-up, the X server may complain that the LOGITECH serial mouse is not responding. This may also happen during VT flipping. The code in INTERACTIVE X11 that allows the mouse to coexist with VP/ix attempts to query the mouse for its current parameters and may become out of sync when the mouse is moved at start-up or VT flip time. The workaround is to not move the mouse during these times.

- Since NUM-LOCK and CAPS-LOCK are modifier keys, they affect the mouse buttons if either is on. You must configure your window manager to invoke the same functions regardless of whether NUM-LOCK or CAPS-LOCK are on or not.
- Panning may not work on all TIGA servers depending upon your hardware environment. Refer to the following Runtime System manual entries for specific references to panning support for your board:

  Xcpqag(1)
  Xdcaga(1)
  Xigsp(1)
  Xmsfdp(1)
  Xnnp(1)
  Xnnpi(1)
  Xrren2(1)
  Xsp200(1)
  Xtisdb(1)

- The Cornerstone (`Xcvc`) server in its default configuration may not recognize some of the newer board models. If server failure occurs and the console displays an error message such as `Unknown CVC board type; must specify resolution`, you must add the appropriate parameters to fields in the `Xconfig` file (see *Xcvc*(1) for information).
- Downloading a big bitmap may take a great deal of time on some servers, and the cursor freezes during the process. This time period can be up to 3 or 4 minutes, giving the illusion that the server has stopped operating.

12. CLIENTS

The `/usr/lib/X11` directory contains some X client resource default and example files:

- The `app-defaults` directory contains files for applications that require only a single file for default settings.
- Applications with more than one file, such as *xdm*(1), *xinit*(1), and *uwm*(1), require individual client directories.

The following information applies to specific client application programs:

- `xpcterm`
  - Note that the AT* terminal character set has special character glyphs in the range 0 through 31 and 127 through 255. Only the `ega` and `vga` fonts supplied with INTERACTIVE X11 support these special character glyphs. Programs that use these special character glyphs, such as those in the TEN/PLUS* Environment and the VP/ix Environment, should be run from an `xpcterm` window that uses one of these fonts. Programs that do not need these special character glyphs can be run using any other font.
  - `xpcterm` also emulates the AT keyboard. Programs that perform better when using an AT-style keyboard should be run under `xpcterm` rather than `xterm`. Further, when running `vpix` in an `xpcterm` window, the `TERM` shell variable should be set to `xpcterm` for it to function correctly. This should be the default `TERM` variable setting under `xpcterm`.
  - VP/ix Version 1.1.1 cannot handle the SIGWINCH signal that is sent when a window is resized. This has been corrected in VP/ix Version 1.2.0. If you are still running VP/ix Version 1.1.1, beginning with release 1.3 of INTERACTIVE X11, `xpcterm` provides the option of not sending the SIGWINCH signal to a process such as VP/ix. Refer to the information on the menu items accessible via the second button menu (accessed via CTRL second button).
- `xpr`
  - When using `xpr` with the HP* LaserJet*, the printer must have at least 1.5 MB of memory in order to process a full-page image.
- `xterm`
  - The `xterm` program emulates a `vs100` (VT102*-style) terminal. Set your `TERM` variable to `xterm` or `vs100` if you have a 66-line `xterm` window. Set your `TERM` variable to `xterms`, `vs100s`, or `vt100` if you have a 24-line `xterm` window.

- Any unrecognized escape sequence will cause `xterm` to not process input or output until a `soft reset` is done via the `xterm` menu.

- `xdm`
  - If you are running `xdm` to an X terminal that does not support the XDMCP protocol, you will notice the sessions on the X terminal being terminated after a short period of time. This happens because `xdm` pings the X terminal, and the X terminal does not respond. To prevent this from occuring, add one of the following lines to the appropriate `xdm` configuration file (the default file is `/usr/lib/X11/xdm/xdm-config`):

    ```
    DisplayManager.HOST_SERVER.pingInterval: 0
    ```

    or:

    ```
    DisplayManager*pingInterval: 0
    ```

    where `HOST` is the host name of the X terminal, and `SERVER` is the server number, usually `0`.

Some client programs that ran successfully in previous releases of X may terminate unexpectedly when used with X 2.0. This may be avoided by starting the server with the `-bc` option or by running `xset bc`.

13. MISCELLANEOUS

Messages that start with the word `WARNING` are messages that give you information. If you receive a `WARNING` message, it does not mean that something is wrong with your system.

If your EGA or VGA display adapter has auto-switching, turn it off. The auto-switching feature tries to determine the type of adapter the software is expecting. Some manufacturers call this the *VGA Implementation Type*.

In UNIX System V/386 Release 3.2, the `poll` system call is only capable of polling on STREAMS devices. INTERACTIVE has enhanced the operating system so that it supports polling on pipes and terminal (tty) devices. Application developers can use the `spipe` library call to set up a STREAMS pipe connection.

Using the INTERACTIVE Easy Windows Environment

CONTENTS

Using the INTERACTIVE Easy Windows* Environment

1. INTRODUCTION

Welcome to the world of the INTERACTIVE Easy Windows Environment. When you have finished installing this package, you will have an easy-to-use graphical interface to the INTERACTIVE UNIX* Operating System on the console. You will be able to access the power of the INTERACTIVE UNIX System through a simple, understandable desktop manager, without having to integrate and configure your system by hand. INTERACTIVE makes installation and configuration almost automatic.

Advanced Users – Getting Started

The INTERACTIVE Easy Windows Environment can be used only on the console. The following software must be installed in this order, configured, and running correctly, before you install the *Easy Windows Environment* diskette:

INTERACTIVE X11 Runtime System
- *System and Servers* diskettes
- *Clients* diskettes
- The appropriate server diskette, if a non-VGA display is to be used

In addition, in order to obtain maximum functionality, the following packages may be installed:

INTERACTIVE Motif* Window Manager
INTERACTIVE Looking Glass* Desktop Manager

After the software listed above is installed and running, insert the diskette labelled *Easy Windows Environment* and type `sysadm installpkg`.

2. INSTALLATION REQUIREMENTS

The INTERACTIVE Easy Windows Environment can be used only on the console. To install and run the INTERACTIVE Easy Windows Environment, you must first install the following software on your system, in this order:

- INTERACTIVE X11 Runtime System
  System and Servers diskettes
  Clients diskettes
  The appropriate server diskette, if a non-VGA display is to be used

In addition, the following packages may be installed at any point to obtain full functionality:

- INTERACTIVE Motif Window Manager
- INTERACTIVE Looking Glass Desktop Manager

☛ These packages must be *configured and running correctly* to successfully install and use the INTERACTIVE Easy Windows Environment.

You will use the same `sysadm installpkg` command that is used to install other INTERACTIVE software.

- To install the INTERACTIVE X11 Runtime System, read the "INTERACTIVE X11 Installation Instructions and Maintenance Procedures" in this guide. If you are installing the INTERACTIVE X11 Runtime System on your machine for the first time, build a new kernel when prompted, but do not reboot the machine at that time. You will reboot at a later time in the Easy Windows Environment installation.
- To install the INTERACTIVE Motif Window Manager, read the "INTERACTIVE Motif Window Manager Release Notes" and the "INTERACTIVE Motif Window Manager Installation Instructions" in this guide.
- To install the INTERACTIVE Looking Glass Desktop Manager, read the *INTERACTIVE Looking Glass Professional Release Notes*.

☛ Note that if you have previously installed and configured any of these software packages and you install the INTERACTIVE Easy Windows Environment, the system will ask if it should *overwrite the configuration files* in `/usr/lib/X11/xdm`. You may want to back up or change the name of any of these configuration files you want to save.

3. KNOWN PROBLEMS AND WORKAROUNDS

For users with 16 color X Windows Servers, the combination of running Looking Glass and Motif requests more colors for their use than the 16 colors that are available. As a result, the first client appears in full color (if used with Easy Windows this is the Motif Window Manager), and the next client (Looking Glass) appears in monochrome. Two workarounds exist:

- Reconfigure `mwm` to use fewer colors (typically by using the sample `Mwm` configuration files provided in the `MWM` subset).
- Upgrade the display subsystem to one with greater color capacity (in this case, to a 256 color adapter).

4. INSTALLING THE INTERACTIVE EASY WINDOWS ENVIRONMENT

After you have installed the software listed in the previous section, you need to install the diskette labelled *Easy Windows Environment*.

1. Insert the diskette into the drive and type `sysadm installpkg`. On systems with two diskette drives, the system then prompts you to enter the number of the drive from which you want to install. Your screen will look similar to this:

```
        This system has two diskette drives.

Enter the drive number you wish to install
from ((default) 0, 1):
```

2. Type `0` if you plan to install from the `A:` drive, `1` if you plan to install from the `B:` drive. The system then prompts you for the density of the diskettes you are using for the installation. Your screen will look similar to this:

```
Enter density of the diskettes you are installing from:

1) 1.2MB   (5 1/4" high density)
2) 1.44MB  (3 1/2" high density)
3) 360KB   (5 1/4" low density)
4) 720KB   (3 1/2" low density)

Please enter #(1-4), default 1:
```

3. Type in the number corresponding to the type of diskettes you are using for the installation. The system asks you to insert the first diskette of the subset into the diskette drive. Your screen will look similar to this:

```
Confirm

Please insert the floppy disk.

If the program installation requires more than one floppy
disk, be sure to insert the disks in the proper order,
starting with disk number 1.
After the first floppy disk, instructions will be provided
for inserting the remaining floppy disks.

Strike ENTER when ready
or ESC to stop
```

4. Press ENTER. The system asks you to confirm that you want to begin the installation of that specific subset:

```
Installation is in progress -- do not remove the floppy disk.
Install INTERACTIVE Easy Windows Environment -
Version 2.0? (y):
```

5. Press **ENTER** to start the installation process. The system displays the subset file names that are being loaded onto your machine. Your screen will look similar to this:

```
Installing INTERACTIVE Easy Windows Environment -
Version 2.0.
Copyright (c) 1990-1991 Interactive Systems Corp
All Rights Reserved
The following files are being installed:
                .
                .
                .
762 blocks

In order to configure the Easy Windows Environment,
you will need to run the command 'sysadm easysetup'.
Do you want to do that now? [y, n]
```

6. Type `y` and press **ENTER**. Your screen will look similar to this:

```
Running Sysadm...

Running subcommand 'easysetup' from menu 'easywindowmgmt',
EASY WINDOWS ENVIRONMENT MANAGEMENT MENU

This script will allow you to enable or disable automatic
startup of INTERACTIVE Easy Windows sessions for the
console device.


Available operations are:
        a) add entries for automatic startup of
           Easy Windows sessions
        d) delete entries for automatic session startup
        q) quit

enter selection:
```

7. Type `a` and press **ENTER**. Your screen will look similar to this:

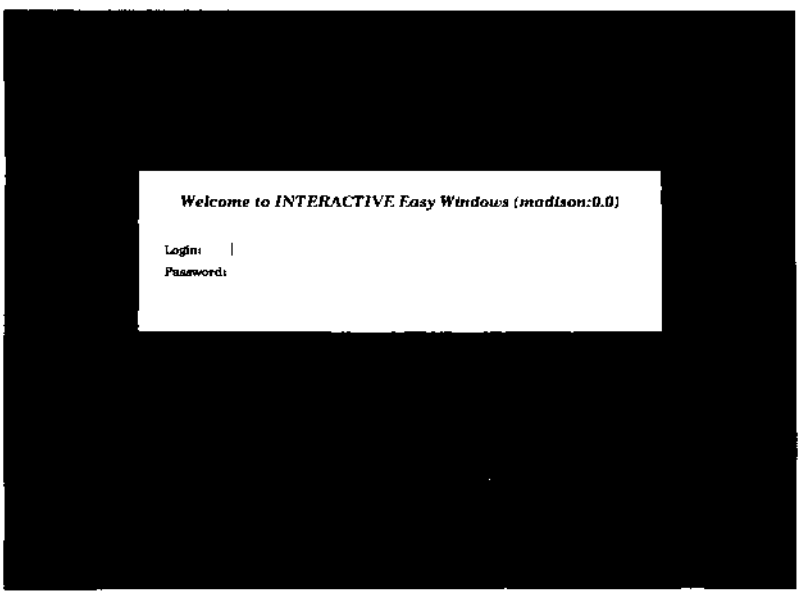
```
Which display number do you want to use [0]?
```

8. You must choose a display that is configured to use `/dev/console`. Press **ENTER** to accept the default or type in the number of the display you want to use and press **ENTER**. Your screen will look similar to this:

```
Inittab has been updated.

Easy Windows sessions on the console have been enabled.
You should now reboot the system for the change to take
effect.

Installation of Easy Windows Environment - Version 2.0
is complete.


Confirm

Please insert the floppy disk.

If the program installation requires more than one floppy
disk, be sure to insert the disks in the proper order,
starting with disk number 1.
After the first floppy disk, instructions will be provided
for inserting the remaining floppy disks.

Strike ENTER when ready
or ESC to stop.
```

9. Press ESC to return to the system prompt. You must now use the `shutdown` command to shut down and reboot your machine. (For information about `shutdown`, refer to section 3, "SHUTTING DOWN AND BRINGING UP THE SYSTEM" in the "System Administration for New Users of the INTERACTIVE UNIX Operating System" in the *INTERACTIVE UNIX System Guide for New Users*.

10. When you log in again, your screen will look similar to this:



11. Log in as usual. (If you need to abort the Easy Windows Environment installation for some reason, you can press ESC at this point to do so.) Your screen will look similar to this:

```
Starting xdm in 30 seconds.  Press ESC to abort to a
console getty or press x to start xdm immediately.
```

12. Type **x** to start the Easy Windows Environment right away. Your screen will look similar to this:

```
This is the first time you have logged in under the
INTERACTIVE Easy Windows Environment.  You now have
the option of installing the Easy Windows
configuration files into your home directory, which
will automate startup every time you log into the
Easy Windows Environment.

(Remember to move the cursor into the window before typing
your answers.)

Do you want to continue [y]?
```

13. Put the cursor in the window and press ENTER to continue. If you want to stop the installation for some reason, type **n** (meaning "no"), press ENTER, and skip to section 6.2, "Stopping Installation." If you choose to continue, your screen will look similar to this:

```
This procedure will allow you to install the standard Easy
Windows configuration files.  For each file, if you already
have a file of the same name, you will be asked if you want
to replace it with the Easy Windows version of the file.
If you choose to do so, the procedure will save your old
file with the extension ".old" appended to it.  If it
cannot do this, it will warn you and allow you overwrite
or skip the file.

Install the X resources file (.Xdefaults) [y]?
```

14. If you are unfamiliar with these configuration files, you should accept the default and press ENTER to install the file. The system confirms your choice for each file and displays the location where it is being installed. For example:

```
Installed the X resources file as /usr/jane/.Xdefaults
```

15. You are asked about each of the following configuration files in turn. Press ENTER each time you want to install the new file; type **n** (meaning "no") and press ENTER each time you do *not* want the new file to be installed.

```
Install the Motif Window Manager configuration file
    (.mwmrc) [y]?

Install the xdm Xsession file (.xsession) [y]?

Do you want to install the standard MWM resource file for
    your display [y]?

Do you want to have an icon box [n]?

Install the MWM low-res color resource file (Mwm) [y]?

Do you want to install the standard Looking Glass files for
    your display [y]?
```

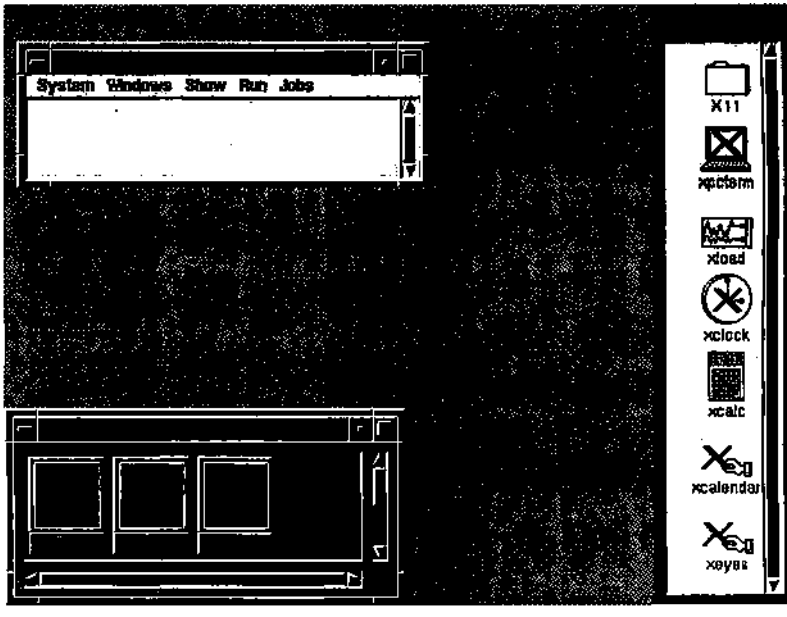
After you press ENTER to accept the default, your screen will look similar to this:

```
Installed the Looking Glass configuration file as
   /usr/jane/lg/lg_config
Installed the Looking Glass user preferences file as
   /usr/jane/lg/lg_pref
Installed the Looking Glass Desktop file as
   /usr/jane/lg/lg_desktop
Installed the Looking Glass color choices file as
   /usr/jane/lg/lg_colors
Installed the Looking Glass directory history file as
   /usr/jane/lg/lg_dirhist
```

The system then lists the files it has added or changed during the installation. Your screen will look similar to this:

```
The following files have been installed or updated for the
INTERACTIVE Easy Windows Environment:

        /usr/jane/.Xdefaults
        /usr/jane/.mwmrc
        /usr/jane/.xsession
        /usr/jane/Mwm
        /usr/jane/lg/lg_config
        /usr/jane/lg/lg_pref
        /usr/jane/lg/lg_desktop
        /usr/jane/lg/lg_colors
        /usr/jane/lg/lg_dirhist

Press [ENTER] to start your Easy Windows session:
```

16. Press ENTER. The INTERACTIVE Easy Windows Environment installation is complete and your screen will look similar to this:



Note that your screen may look somewhat different, depending upon the resolution of your display.

If you want to change the way your Looking Glass desktop looks, refer to the *INTERACTIVE Looking Glass User's Guide*.

5. USING THE EASY WINDOWS ENVIRONMENT

For information about using the Motif and Looking Glass environments, refer to the *INTERACTIVE Looking Glass User's Guide* and the *X Window System User's Guide, Motif Edition*.

5.1 Using the Icon Box

If, while installing, you chose to have an icon box, your icons will be stored in a scrollable box at the bottom of your screen. The icon box is described in the *X Window System User's Guide, Motif Edition*.

5.2 Exiting the Easy Windows Environment

To preserve the Looking Glass environment when exiting Easy Windows, you must first quit Looking Glass using the System menu in the Looking Glass window. Then, move the cursor outside any windows onto the background and hold down the left mouse button to access the Easy Windows Menu. Select `Quit` from the menu. You will be returned to the following message:

```
Starting xdm in 30 seconds.  Press ESC to abort to a
console getty or press x to start xdm immediately.
```

Press ESC to obtain an INTERACTIVE UNIX System login prompt.

6. TROUBLESHOOTING

- The Easy Windows Environment can only be run from the console. It will not work properly on a terminal.
- An insufficient amount of memory (6 MB or less) may affect the functionality of the Easy Windows Environment. For example, changing virtuals terminals rapidly several times in a row on such a machine may kill your Easy Windows Environment session.
- You must exit the Easy Windows Environment before using the `shutdown` command to shut down your system.
- If the xclock comes up in Eastern Standard Time and you want to change it to another time zone, you must edit the `/etc/default/login` file. For example, if the file entry is `EST5EDT` and you are in the Pacific time zone, you should change the entry to `PST8PDT`. The `8` signifies the number of hours your time zone is off from Greenwich Mean Time.
- If something is wrong with your INTERACTIVE X11 environment, it may cause the X11 display manager (`xdm`) to respawn every 30 seconds. For example, if you have a serial mouse that is unplugged from the system and you start up the Easy Windows Environment, your screen may black out every 30 seconds and display the startup message. To stop this, do the following:

  1. To break out of the Easy Windows Environment session, press ESC when prompted.
  2. Log in as `root` (ignore the blackouts; the system will respond in spite of them). Changing the `init` level by typing `init 2` will stop the blackouts altogether if you can do so without inconveniencing other users on your system.
  3. Type `sysadm easysetup` and select option `d` to delete your automatic startup files.
  4. Type `shutdown` and then reboot the machine.
  5. Check your INTERACTIVE X11 environment carefully to make sure it is functioning properly. If you cannot find a problem there, check your Motif and Looking Glass environments.

7. NOTES FOR ADVANCED USERS

7.1 What the Easy Windows Environment Provides

The INTERACTIVE Easy Windows Environment modifies the `/etc/inittab` file to start an X-based login procedure. Once the X-based login screen appears and you have logged in, the Easy Windows Environment scripts automatically bring up the X11, Motif, and Looking Glass software. Either new versions of the following files are installed or modifications are made to your existing files:

- **`$HOME/.Xdefaults`**
  Sets colors and various options for X11 clients.
- **`$HOME/.Xresources`**
  Sets colors and message strings printed at login.
- **`$HOME/.mwmrc`**
  Sets up the Easy Windows Menu. If you accept the Easy Windows version of `.mwmrc`, the default button bindings for `mwmrc` are used as the default in your `.Xdefaults` file.
- **`$HOME/.xsession`**
  Sets up the initial environment and starts clients for each user; uses `xpcterm` rather than `xterm`.
- **`$HOME/Mwm`**
  Motif Window Manager resource configuration file that specifies colors and options.
- **`$HOME/lg/lg_config`**
  Used to configure the Looking Glass Desktop environment.
- **`$HOME/lg/lg_pref`**
  Used to configure the Looking Glass Desktop environment.
- **`$HOME/lg/lg_desktop`**
  Used to configure the Looking Glass Desktop environment.
- **`$HOME/lg/lg_colors`**
  Used to configure the Looking Glass Desktop environment.
- **`$HOME/lg/lg_dirhist`**
  Used to configure the Looking Glass Desktop environment.

The Easy Windows Environment also adds `/usr/bin/X11` to your `PATH` variable in `.cshrc`, `.login`, or `.profile`, if any

of those files are present in your directory. If none is present, it creates a `.profile` file.

7.2 Stopping Installation

If you choose to stop the installation after logging in because you want to continue to use your current system configuration rather than the Easy Windows Environment, then depending on your current X11 configuration, one of several actions can happen.

If you are using an `xdm` start up protocol, the system starts your INTERACTIVE X11 Runtime System as usual.

If you are using an `xinit` protocol, the system asks the following:

```
Do you want to run your existing .xinitrc when you log in [y]?
```

If you press ENTER, your `.xinitrc` file is copied to `$HOME/.xsession` and the INTERACTIVE X11 Runtime System is started. Note that your `.profile` will not be in effect. You will have to edit your `.xsession` file manually to set up the `PATH` variable and any other variables that should be in effect immediately after you log in. If you type `n` and press ENTER, the system will display a typical `xpcterm` window. Refer to *xdm*(1) for more information.

8. UPDATING YOUR EASY WINDOWS ENVIRONMENT

If you install a new release of the INTERACTIVE X11 Runtime System, you will need to reinstall the *Easy Windows Environment* diskette.

9. REMOVING THE EASY WINDOWS ENVIRONMENT

To remove the INTERACTIVE Easy Windows Environment, do the following:

1. Exit the Easy Windows Environment.
2. Run `sysadm easysetup` and select `d` (delete) to return the `inittab` file to its previous state.
3. Run the `shutdown` command and then reboot your system. (`xdm` will no longer be running.)
4. Run `sysadm removepkg` and remove the Easy Windows Environment.

You may want to restore any configuration files you saved or merge them with the new versions, which will remain in `/usr/lib/X11/xdm`.

INTERACTIVE Motif*
Window Manager
Version 1.1.1
Release Notes
October 1991

1. INTRODUCTION

The INTERACTIVE Motif Window Manager is based on OSF/Motif* Revision 1.1.1 from the Open Software Foundation* (OSF*). It contains the Motif Window Manager (MWM) which provides a standard graphical user interface (GUI) that incorporates the behavior of the Presentation Manager as well as an X Window System* intrinsics-based toolkit. The INTERACTIVE Motif Window Manager is a licensed version of Motif that has been optimized for use with INTERACTIVE X11.

2. DOCUMENTATION

The "INTERACTIVE Motif Window Manager Installation Instructions" included with this release describe how to install the INTERACTIVE Motif Window Manager.

An *OSF/Motif User's Guide* is available for persons wanting to use the Motif Window Manager. This document can be obtained at your local bookstore or by contacting Prentice Hall.

In the United States:

Prentice Hall
Englewood Cliffs, New Jersey 07632
Telephone: (201) 767-5937

Internationally:

Simon and Schuster International Group
66 Wood Lane End
Hemel Hempstead, HP2 4RG, UK

3. INCOMPATIBILITY WITH PREVIOUS RELEASES OF THE MOTIF WINDOW MANAGER

Due to changes in the default colormap for X11R4, the Motif Window Manager's default behavior for drawing borders on inactive windows has changed. This will result in a slightly different appearance of inactive windows.

4. CAVEAT

The Motif menus might not pop up when a mouse button is clicked if NUM-LOCK is activated. The keyboard accelerators might not work if NUM-LOCK is activated. Please deactivate NUM-LOCK in these cases when accessing Motif menus and keyboard accelerators.

INTERACTIVE Motif Window Manager Installation Instructions

CONTENTS

INTERACTIVE Motif*
Window Manager
Installation Instructions

1. OVERVIEW

This document describes the basic requirements and procedures that are necessary to install the INTERACTIVE Motif Window Manager.

All menus and system messages in this document are applicable for installation on an INTERACTIVE UNIX* Operating System Version 2.2 or later. If you are installing the INTERACTIVE Motif Window Manager on an earlier version of the INTERACTIVE UNIX System, your screen displays may appear different from those shown.

The INTERACTIVE Motif Window Manager is installed on your fixed disk using the `sysadm` utility. It is comprised of the INTERACTIVE Motif Window Manager (MWM) and associated resource files. It uses about 720 kb of disk space and requires that the INTERACTIVE X11 Runtime System package already be installed.

2. INSTALLING THE INTERACTIVE MOTIF WINDOW MANAGER

1. To begin the installation, use the system administration command, `sysadm`, or log in as `sysadm` to access the Main menu. Your screen will look similar to this:

```
                        SYSTEM ADMINISTRATION

    1 diskmgmt        disk management menu
    2 filemgmt        file management menu
    3 machinemgmt     machine management menu
    4 packagemgmt     package management menu
    5 softwaremgmt    software management menu
    6 syssetup        system setup menu
    7 ttymgmt         tty management menu
    8 usermgmt        user management menu

  Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  q to QUIT:
```

2. Type 5 to access the Software Management menu. Your screen will then look similar to this:

```
                    SOFTWARE MANAGEMENT

  1 installpkg      install new software package onto built-in disk
  2 listpkg         list packages already installed
  3 removepkg       remove previously installed package from built-in disk
  4 runpkg          run software package without installing it

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 1, `installpkg`.

   The system will then prompt you for the density of the diskettes you will use for the installation. Your screen will look similar to this:

```
Enter density of the diskettes you are installing from:
  1) 1.2  MB  ( 5 1/4" high density )
  2) 1.44 MB  ( 3 1/2" high density )
  3) 360  Kb  ( 5 1/4" low density )
  4) 720  Kb  ( 3 1/2" low density )

Please enter #(1-4), default 1:
```

4. Type the number that corresponds to the type of diskettes you are using for the installation. The system will then ask you to insert the first diskette of the subset into the diskette drive. Your screen will look similar to this:

```
Confirm

Please insert the floppy disk

If the program installation requires more than one floppy
disk, be sure to insert the disks in the proper order,
starting with disk number 1.
After the first floppy disk, instructions will be provided
for inserting the remaining floppy disks.

Strike ENTER when ready
or ESC to stop
```

5. Insert the *Motif Window Manager* diskette into the diskette drive. The system asks you to confirm that this is the package you want to install. Press **ENTER** to start the installation process. The system will display the name of each subset file on your terminal screen. (The file names listed below may appear in a different order during your installation.) The screen will look similar to this:

```
Install the INTERACTIVE X11: INTERACTIVE Motif Window
Manager-Version 1.1.1 package? (y):
Installing the INTERACTIVE X11: INTERACTIVE Motif Window
Manager-Version 1.1.1.
Copyright (c) 1989-1991 Interactive Systems Corporation
All Rights Reserved

Derived from sources distributed by the Open Software
Foundation, Inc.
(c) Copyright 1989  Open Software Foundation, Inc.
With additional copyrights from the following sources:
(c) Copyright 1989  DIGITAL EQUIPMENT CORPORATION
(c) Copyright 1987, 1988, 1989 HEWLETT-PACKARD COMPANY
(c) Copyright 1988  MASSACHUSETTS INSTITUTE OF TECHNOLOGY
(c) Copyright 1988  MICROSOFT CORPORATION

The following files are being installed:
/usr/bin/X11/mwm
/usr/lib/X11/app-defaults/Mwm
/usr/lib/X11/examples/Mwm.2bw
/usr/lib/X11/examples/Mwm.4bl
/usr/lib/X11/examples/Mwm.4br
/usr/lib/X11/examples/Mwm.4cb
/usr/lib/X11/examples/Mwm.4dt
/usr/lib/X11/examples/Mwm.4g
/usr/lib/X11/examples/Mwm.4lg
/usr/lib/X11/examples/Mwm.4lsb
/usr/lib/X11/examples/Mwm.4mvr
/usr/lib/X11/examples/Mwm.4pg
/usr/lib/X11/examples/Mwm.4sb
/usr/lib/X11/examples/Mwm.4w
/usr/lib/X11/examples/Mwm.bl
/usr/lib/X11/examples/Mwm.br
/usr/lib/X11/examples/Mwm.cb
/usr/lib/X11/examples/Mwm.dt
/usr/lib/X11/examples/Mwm.lsb
/usr/lib/X11/examples/Mwm.mvr
/usr/lib/X11/examples/Mwm.pg
/usr/lib/X11/examples/Mwm.sb
/usr/lib/X11/system.mwmrc
/usr/options/xm.name
1417 blocks
```

6. The INTERACTIVE Motif Window Manager is now installed on your fixed disk.

3. DISTRIBUTED FILES

The INTERACTIVE Motif Window Manager is distributed on one diskette. The contents of the diskette are copied to the `/usr/bin/X11` and `/usr/lib/X11` directories on your fixed disk. This section briefly describes many of the files distributed with the INTERACTIVE Motif Window Manager.

- `/usr/bin/X11/mwm`
  This file is the INTERACTIVE Motif Window Manager. By default it uses the following files for its configuration information:

`/usr/lib/X11/system.mwmrc`
This file contains the default definitions of the menu system, the button bindings, and the key bindings in the INTERACTIVE Motif Window Manager.

`/usr/lib/X11/app-defaults/Mwm`
This file contains the default color, font, and resource definitions for the INTERACTIVE Motif Window Manager.

- `/usr/lib/X11/examples/Mwm.*`
These files contain additional example color and resource definitions for the INTERACTIVE Motif Window Manager:
  - The file `Mwm.2bw` is the default two-color binding.
  - The files `Mwm.4*` show a few of the possible four- or five-color bindings.
  - The remaining files show the other eight- and nine-color combinations for the INTERACTIVE Motif Window Manager.
  - The files `Mwm.cb` and `Mwm.bl` are generally considered the two most frequently used INTERACTIVE Motif Window Manager default color combinations.

Character Bitmap Distribution Format

CONTENTS

Character Bitmap Distribution Format

1. INTRODUCTION

This document describes the Character Bitmap Distribution Format (BDF), Version 2.1. BDF is an X Consortium standard for font interchange, intended to be easily understood by both users and computers.

2. FILE FORMAT

Character bitmap information is distributed in an USASCII encoded, user-readable form. Each file is encoded in the printable characters (octal 40 through 176) of USASCII plus a carriage return and a line-feed character. Each file consists of a sequence of variable-length lines. Each line is terminated by a carriage-return (octal 015) and a line-feed character (octal 012) or by just a line-feed character.

The information about a particular family and typeface at one size and orientation is contained in one file. The file begins with information pertaining to the typeface as a whole, followed by the information and bitmaps for the individual characters.

A font bitmap description file has the following general form, where each item is contained on a separate line of text in the file. Items on a line are separated by spaces.

1. The word `STARTFONT` followed by a version number indicating the exact file format used. The version number described here is `2.1`.
2. One or more lines beginning with the word `COMMENT`. These lines may be ignored by any program reading the file.
3. The word `FONT` followed by the full name of the font. Note that the name continues all the way to the end of the line, and it may contain spaces.
4. The word `SIZE` followed by the *point size* of the characters, the *x resolution*, and the *y resolution* of the device for which these characters are intended.

5. The word `FONTBOUNDINGBOX` followed by the width in x, height in y, and the x and y displacement of the lower left-hand corner from the *origin*. (See the examples in section 4.)

6. Optionally the word `STARTPROPERTIES` followed by the number of properties (p) that follow.

7. Then come p lines consisting of a word for the *property name* followed by either an integer or string surrounded by double-quotes (octal 042). Internal double-quote characters are indicated by using two in a row.

8. Properties named `FONT_ASCENT`, `FONT_DESCENT`, and `DEFAULT_CHAR` should be provided to define the logical font-ascent and font-descent and the default-char for the font. These properties will be removed from the actual font properties in the binary form produced by a compiler. If these properties are not provided, a compiler may reject the font or may compute (arbitrary) values for these properties.

9. The property section, if it exists, is terminated by `ENDPROPERTIES`.

10. The word `CHARS` followed by the number of character segments (c) that follow.

11. Then come c character segments of the form:

    a. The word `STARTCHAR` followed by up to 14 characters (no blanks) of the descriptive *name* of the glyph.

    b. The word `ENCODING` followed by a positive integer representing the Adobe* Standard Encoding value. If the character is not a member of the Adobe Standard Encoding, `ENCODING` is followed by `-1` and an optional integer specifying the glyph index; if the glyph index is not specified, a compiler will typically ignore the character segment.

    c. The word `SWIDTH` followed by the *scalable width* in x and y of character. Scalable widths are in units of 1/1000th of the size of the character. If the size of the character is p points, the width information must be scaled by $p/1000$ to get the width of the character in printer's points. This width information should be considered as a vector indicating the position of the next character's origin relative to the origin of the current

character. To convert the scalable width to the width in device pixels, multiply `SWIDTH` times $p/1000$ times $r/72$, where r is the device resolution in pixels per inch. The result is a real number giving the ideal print width in device pixels. The actual device width must of course be an integral number of device pixels; it is given in the next entry. The `SWIDTH` y value should always be zero (`0`) for a standard X font.

d. The word `DWIDTH` followed by the width in x and y of the character in device units. Like the `SWIDTH`, this width information is a vector indicating the position of the next character's origin relative to the origin of the current character. The `DWIDTH` y value should always be zero for a standard X font.

e. The word `BBX` followed by the width in x (*BBw*), height in y (*BBh*), and x and y displacement (*BBox, BBoy*) of the lower left-hand corner from the *origin* of the character.

f. The optional word `ATTRIBUTES` followed by the attributes as 4 *hex-encoded* characters. The interpretation of these attributes is not defined in this document.

g. The word `BITMAP`.

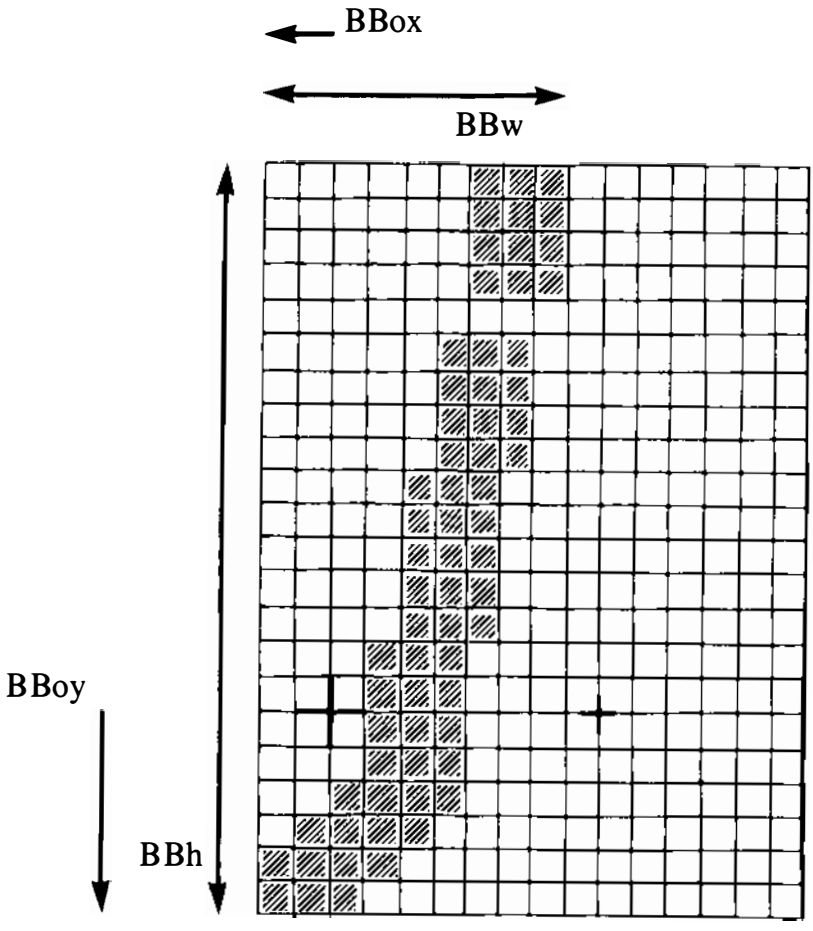
h. *h* lines of *hex-encoded bitmap,* padded on the right with zeros to the nearest byte, that is, a multiple of 8.

i. The word `ENDCHAR`.

12. The file is terminated with the word `ENDFONT`.

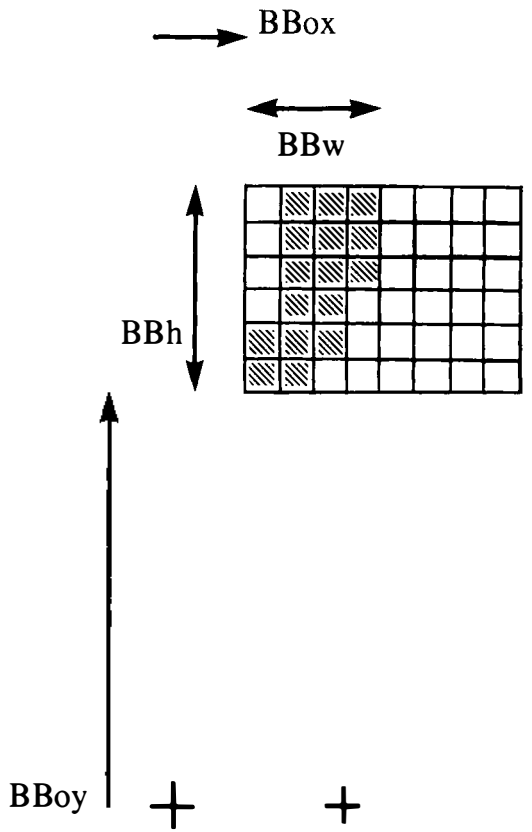
3. METRIC INFORMATION

Figures 1 and 2 best illustrate the bitmap format and character metric information.



BBw = 9, BBh = 22, BBox = -2, BBoy = -6
Rounded character width = 8 0
"+" = character origin and width

Figure 1. An Example of a Descender



BBh = 6, BBw = 4, BBox = +2, BBoy = +12
Rounded character width = 5 0

Figure 2. An Example With the Origin Outside the Bounding Box

4. AN EXAMPLE FILE

Figure 3 is an abbreviated example of a bitmap file containing the specification of two characters (the `j` and `quoteright` in 4).

```
STARTFONT @value(formatversion)
COMMENT This is a sample font in 2.1 format.
FONT Helvetica-Bold
SIZE 8 200 200
FONTBOUNDINGBOX 9 24 -2 -6
STARTPROPERTIES 2
MinSpace 4
Copyright "Copyright (c) 1987 Adobe Systems, Inc."
ENDPROPERTIES
CHARS 2
STARTCHAR j
ENCODING 106
SWIDTH 355 0
DWIDTH 8 0
BBX 9 22 -2 -6
BITMAP
0380
0380
0380
0380
0000
0700
0700
0700
0700
0E00
0E00
0E00
0E00
0E00
1C00
1C00
1C00
1C00
2C00
7800
F000
E000
ENDCHAR
STARTCHAR quoteright
ENCODING 39
SWIDTH 223 0
DWIDTH 5 0
BBX 4 5 2 12
ATTRIBUTES 01C0
BITMAP
70
70
60
E0
C0
ENDCHAR
ENDFONT
```

Figure 3. A Short Example File

INTERACTIVE X11 Installation Instructions and Maintenance Procedures

CONTENTS

INTERACTIVE X11 Installation Instructions and Maintenance Procedures

1. INTRODUCTION

This document describes the basic requirements and procedures that are necessary to install INTERACTIVE X11.

All menus and system messages in this document are applicable for installation on the INTERACTIVE UNIX* Operating System, Version 2.2 or later.

Once you have completed the basic system installation, refer to section 5 in this document to learn how to configure user profiles, install and configure servers, and tailor the system to match your requirements.

1.1 Overview of This Document

This document is divided into eight major sections:

1. INTRODUCTION
   This section provides a general overview of this document.
2. GETTING STARTED
   This section discusses hardware and software considerations, and provides an overview of the system administration menu (`sysadm`).
3. SYSTEM PACKAGES
   This section describes the INTERACTIVE X11 packages.
4. INSTALLING INTERACTIVE X11
   This section explains how to install the system. It also describes how to perform the preliminary setup procedures that are required to use the system once it is installed.

5. CONFIGURATION AND MANAGEMENT PROCEDURES
   This section discusses building new servers, removing servers, making default servers, modifying configuration information, installing new X device drivers, adding X users, and deleting X users.
6. RUNNING INTERACTIVE X11
   This section explains how to run X11.
7. REMOVING X11 SYSTEM PACKAGES
   This section describes how to remove X11 system packages from the system.
8. KERNEL CONSIDERATIONS
   This section provides suggestions for making sure your kernel has the required facilities to run X.

1.2 What Will I Learn From This Document?

This document provides step-by-step instructions designed to help you install and maintain INTERACTIVE X11. This document describes:

- The component system packages of INTERACTIVE X11.
- The basic installation and setup procedures (most are performed automatically by the system).
- The configuration and maintenance procedures for managing INTERACTIVE X11 after it is installed.
- How to remove system packages from your system.
- The kernel considerations for running X11.

2. GETTING STARTED

Before you install INTERACTIVE X11 on your computer, you must complete the following tasks:

1. Read the "INTERACTIVE X11 Release Notes" in this guide for any additional information required to install the release.
2. Assemble and set up your hardware.
3. Ensure that the INTERACTIVE UNIX System, the Kernel Configuration subset, and the STREAMS Facilities subset are installed on your system.
4. If you plan to use the VP/ix* Environment, it should be installed prior to installing INTERACTIVE X11.
5. Running X11 across the network currently requires INTERACTIVE TCP/IP (Transmission Control Protocol/Internet Protocol). If you want to use the INTERACTIVE TCP/IP facilities, this extension must be installed on your system. Note that this does *not* apply to local connections. Currently the `socket` library as supplied runs only with INTERACTIVE TCP/IP. The InterLan* networking software is not supported under X11.
6. If you are installing the INTERACTIVE X11 Development System, the INTERACTIVE Software Development System extension must be installed on your system.

After verifying these steps, you will be ready to install INTERACTIVE X11.

2.1 Hardware Requirements

To determine the INTERACTIVE X11-specific hardware information you need to install and run X, refer to the "INTERACTIVE X11 Release Notes."

Compare these requirements with your hardware configuration. Make sure that you have available at least the minimum configuration. Then follow the manufacturer's instructions to determine how to assemble and attach all components for operation.

2.2 Installation Using the System Administration Menu

The system administration menu (`sysadm`) can be used to perform the installation. You must use the console terminal to log in as `sysadm`, and you will be asked for the password assigned to `sysadm`.

3. SYSTEM PACKAGES

INTERACTIVE X11 is installed as an extension on the INTERACTIVE UNIX System. The following subsections briefly describe the system packages that comprise this extension.

3.1 Runtime System

The Runtime System allows the user to invoke and utilize the X environment. It consists of the Clients, the System and Servers (which includes the VGA server, kernel driver, system administration scripts, and server configuration facilities), and the Display Servers I and II (which includes servers other than the VGA servers). This system package is intended for users who want to use the X environment, but who do not plan to develop X client application programs (clients). The user cannot develop or build X clients with this package because the required libraries and `include` files are not present. The Runtime System must be present before any other X11 system packages are installed.

3.2 Development System

The Development System must be installed after the Runtime System package. It contains the necessary libraries and `include` files to develop X clients. This system package is designed for developers of INTERACTIVE X11 applications.

3.3 Contributed Software

The Contributed Software is included with the Development System. It contains a number of client programs and fonts from the user-contributed part of the MIT tape and can be installed any time after the Runtime System is installed. The installation of this system package is optional and it is provided on an as-is basis. This software is unsupported, and it may not run on all configurations.

Documentation for the Contributed Software is supplied on-line in the `/usr/man/mann` directory.

3.4 Server Kit

The Server Kit must be installed after the Development System package. It is intended for developers who would like to develop an INTERACTIVE X11 server. The Server Kit is only available to qualified developers. Please contact the INTERACTIVE X11 Product Manager or your sales representative for further information.

4. INSTALLING INTERACTIVE X11

Follow all of the steps given below. You must use the console terminal to log in as `sysadm`, and you will be asked for the password assigned to `sysadm`. Experienced UNIX System users can log in to the console terminal as `root` and perform these same tasks from the command line.

INTERACTIVE X11 Version 2.0 will install on the INTERACTIVE UNIX Operating System, Version 2.2 or later.

You can overlay INTERACTIVE X11 Version 2.0 on a previous release of X11 if one is currently installed.

You should not be in the X11 environment at the time you install (overlay) Version 2.0, because this will cause certain files that must be updated to be inaccessible.

4.1 Removing Previous Releases of X11

It is recommended that you do not remove previous X11 releases, but overlay Version 2.0 on top instead. Refer to the release notes for information on saving any existing files that you may have modified.

4.2 Installing the Runtime System

To install the Runtime System package software on an INTERACTIVE UNIX System:

1. Use the `sysadm` command or log in as `sysadm` and access the Main menu. Your screen will look similar to this:

```
                    SYSTEM ADMINISTRATION

  1 diskmgmt        disk management menu
  2 filemgmt        file management menu
  3 machinemgmt     machine management menu
  4 packagemgmt     package management menu
  5 softwaremgmt    software management menu
  6 syssetup        system setup menu
  7 ttymgmt         tty management menu
  8 usermgmt        user management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  q to QUIT:
```

2. Type `q` at any time to quit the current operation. If a `?` appears as a choice, type `?` for help.

Select option 5 to access the Software Management (`softwaremgmt`) menu. Your screen will look similar to this:

```
                    SOFTWARE MANAGEMENT

  1 installpkg      install new software package onto built-in disk
  2 listpkg         list packages already installed
  3 removepkg       remove previously installed package from built-in disk
  4 runpkg          run software package without installing it

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 1, `installpkg`.

   The system will then prompt you for the density of the diskettes you will use for the installation. Your screen will look similar to this:

```
Enter density of the diskettes you are installing from:
  1) 1.2  MB  ( 5 1/4" high density )
  2) 1.44 MB  ( 3 1/2" high density )
  3) 360  Kb  ( 5 1/4" low density )
  4) 720  Kb  ( 3 1/2" low density )

Please enter #(1-4), default 1:
```

4. Type the number that corresponds to the type of diskettes you are using for the installation, or press ENTER to accept the default. The system will then ask you to insert the first diskette of the Runtime System into the diskette drive. Your screen will look similar to this:

```
Confirm

Please insert the floppy disk

If the program installation requires more than one floppy
disk, be sure to insert the disks in the proper order,
starting with disk number 1.
After the first floppy disk, instructions will be provided
for inserting the remaining floppy disks.

Strike ENTER when ready
or ESC to stop
```

5. If you wish to stop the installation process and return to the system prompt, press ESC. To proceed with the installation, insert the first *Clients* diskette into the drive and press ENTER. The following message will be displayed:

```
Installation is in progress -- do not remove the floppy disk.
Install the INTERACTIVE X11: Runtime System - Clients -
Version 2.0 package?
(y/n):
```

6. Type **y** to proceed with the installation of the Clients subsystem. (You can type **n** to stop the installation.) If you type **y**, your screen will look similar to this:

```
Installing the INTERACTIVE X11: Runtime System - Clients -
Version 2.0.
Copyright (c) 1988-1991 Interactive Systems Corporation
All Rights Reserved

Derived from X11R4 sources distributed by MIT
Copyright 1985-1989 by the Massachusetts Institute of
Technology and others.  Refer to /usr/lib/X11/LABEL for
a complete list.

The following files are being installed:
```

7. A list of the files being copied onto your system from the first diskette will accompany the above message. When all files from the first diskette have been copied, the system will prompt you to insert the second diskette:

```
Floppy diskette number 1 is complete
Remove floppy and insert floppy number 2
Type <enter> when ready:
```

8. Insert the second diskette into the diskette drive and press **ENTER**. A list of the files being copied onto your system from the second diskette will be displayed on your screen.

9. Continue inserting diskettes until all of the diskettes have been copied. When the last diskette has been copied, the following message will be displayed:

```
Installation of INTERACTIVE X11: Runtime System - Clients -
Version 2.0 is complete.
```

10. Insert the first *System and Servers* diskette and press **ENTER**. Your screen will look similar to this:

```
Installing INTERACTIVE X11: Runtime System -
System and Servers - Version 2.0
The following files are being installed:
```

11. Continue inserting the *System and Servers* diskettes until they have all been copied. After the files from the diskettes have been installed, your screen will look similar to this:

```
Installation of a basic X11 system is complete.
Do you plan to use a
standard VGA display or a 256-color VGA display?
[y, n, q]
```

12. If you plan to run a VGA server, you can respond `y` and configure your server at this time by skipping to step 16.

13. If you plan to run a server other than a VGA, then you must now install the appropriate server from the *Display Servers I and II* diskettes. To do this, repeat steps 5 through 9, this time using the *Display Servers I and II* diskettes.

    If you used *Display Servers I*, your screen will look similar to this:

```
Installation is in progress -- do not remove the floppy disk.
This Package Contains the following:

 1. INTERACTIVE X11: Cornerstone Server
 2. INTERACTIVE X11: IBM 8514/A Server
 3. INTERACTIVE X11: Hercules Server
 4. INTERACTIVE X11: Sigma Designs Laserview Server
 5. INTERACTIVE X11: Pixelworks Server
 6. INTERACTIVE X11: Moniterm Viking 21/91 Server
 7. INTERACTIVE X11: Bell Tech WGE Blit Server

Enter a list of numbers separated by spaces for those modules
you want to install, or enter "n" for NONE, or enter "all" to
install ALL modules:
```

14. Select the server or servers you want to use. If, for example, you enter `3` to install the Hercules* server, your screen will look similar to this:

```
Installing INTERACTIVE X11: Hercules Server - Version 2.0
The following files are being installed:

Installation of the "INTERACTIVE X11: Hercules Server -
Version 2.0" subset is now complete.  In order to use X11
with this display, the server must be built and the
driver must be configured into the kernel.
Do you want to do this now?  [y, n, q]
```

15. At this point you must determine whether you would like to configure a server. Before you can use X11, you will need to create your display server configuration entries. If servers are configured that require drivers, the system will also configure the appropriate driver at this time. To configure a server, type `y`.

    If you press `n`, the system will display:

```
This server will be unavailable for use until you execute
"sysadm configmgmt", "sysadm servermgmt",
"sysadm xdrivermgmt", and "kconfig" to configure the server,
build the server, configure the driver, and build the kernel,
respectively.
```

16. If you typed `y`, the system will automatically start up the `configmgmt` utility for display `0`. Display `0` is usually

considered the default console display for X. The `configmgmt` utility will present a series of menus and questions regarding the hardware configuration for this display. In most cases, the *default* selection, `[shown in brackets]`, will be the correct choice. If you need additional information to make your selection, you should refer to *xconfig*(1), *Xconfig*(5), and the server-specific manual entries in this guide for additional configuration information. (The server-specific manual entries are those Section 1 entries in which the entry names begin with an uppercase X.) You can refer to section 5.3 for additional information on listing, adding, and deleting configuration entries.

Your screen will look similar to this:

```
Enter the operation you want to perform:

 1 list
 2 add
 3 delete

[default: q]:
```

17. Type 2 to add servers. Your screen will look similar to this:

```
Enter display number ('l' for list, 'q' to quit) [0]:
Display "0" doesn't exist, create it? [y]:
```

18. Press ENTER to choose display 0 and create it:

```
Create display "0":
Select resource to add:
        1       display
        2       keyboard
        3       mouse
        4       tablet
(select 'l' to list, 'q' to quit)

Enter selection [display]:
```

19. When a menu has a default selection, that choice is shown in brackets in the line at the bottom of the screen that says `Enter selection [`*selection*`]`. If you want to select the default resource `display` in the above screen, press ENTER. (You could also type either the option number (`1`) or the selection name (`display`).) If you want to select a resource other than `display`, you may type either the option number (`2`, `3`, or `4`) or the selection name (`keyboard`, `mouse`, or `tablet`) of that resource. The X configuration procedures will automatically cycle through all of the resource options.

When configuring your resources, note that your display and the information requested will vary according to the display type you select.

☛ Note that you cannot include both a mouse and a tablet in the same configuration.

If you have installed the *Display Servers I* diskette and you select the `display` option, your screen will look similar to this:

```
Select display type:
 1  CVC      Cornerstone Displays (and compatibles)
 2  8514     IBM 8514/A Display Adapter and compatibles
 3  HRC      Hercules monochrome display
 4  LVP      Sigma Designs LaserView
 5  V256     VGA and compatibles (256 Colors)
 6  VGA      VGA and compatibles
 7  EGA      EGA and compatibles
 8  VIKING   Moniterm Viking
(select 'q' to quit)

Enter selection [VGA]:
```

☛ If you have also installed the *Display Servers II* diskette on your system, your menu will include choices in addition to those listed above that are available for installation with this release.

If you are overlaying X11 Version 2.0 on top of Release 1.3, your menu will include additional choices not listed above. However, only the choices shown above are available for installation with this release.

20. Type the option number or the name that corresponds to the type of display you are using. For example, to select the default, VGA, press ENTER, or type 6 or VGA. If you select the VGA display type, your screen will look similar to this:

```
Select VGA board Manufacturer:
 1   ATI
 2   CompuAdd
 3   Dell
 4   Genoa
 5   IBM (or other basic VGA)
 6   Microlabs
 7   Orchid
 8   Paradise
 9   Quadram
 10  Sigma
 11  STB
 12  SunRiver
 13  Tatung
 14  Techmar
 15  Tseng
 16  Trident
 17  Video7 (Headland)

Enter selection [IBM]:
```

21. Type the option number or the name that corresponds to the manufacturer of the VGA display you are using. For example, to select the default, `IBM`, press ENTER, or type `5` or `IBM`. If you select the `IBM` display type, your screen will look similar to this:

```
Select VGA display type:
 1  VGA      Basic 640x480 VGA
 2  VGAPAN   Panning VGA

Enter selection [1]:
```

22. Enter the VGA display type you are using. For example, to select the default, `VGA`, press ENTER or type `1`. If you select the `Basic 640x480 VGA` display type, your screen will look similar to this:

```
Select number of color planes to use:
        1       1 plane  (2 colors)
        2       2 planes (4 colors)
        3       4 planes (16 colors)

Enter selection [3]:
```

23. Type the option number that corresponds to the number of planes you are using. To select the default number of planes (option `3`), press ENTER. Your screen will look similar to this:

```
Enter screen size (in inches) in the form WIDTHxHEIGHT [11x8]:
```

24. Type the size of the screen in whole inches, or press ENTER to select the default screen size (`11x8`). Your screen will look similar to this:

```
Enter full pathname to screen device [/dev/console]:
```

25. Enter the path name of the screen device, or press ENTER to select the default path name (`/dev/console`).

    If you are using a SunRiver* display, SunRiver workstations are specified in the configuration file by indicating the SunRiver device name in place of `/dev/console`, that is, `/dev/st00` for the first SunRiver display, `/dev/st01` for the second, etc. Refer to the *Xvga*(1) manual entry in this guide for additional information on SunRiver device names. For information on the SunRiver keyboard display driver and SunRiver serial ports, refer to *skd*(7) and *sasy*(7) in the *INTERACTIVE UNIX System User's/System Administrator's Reference Manual*.

    Your screen will look similar to this:

```
Select resource to add:
        1       display
        2       keyboard
        3       mouse
        4       tablet
(select '1' to list, 'q' to quit)

Enter selection [keyboard]:
```

26. The display resource has now been configured. The next resource, `keyboard`, has been automatically selected as the next default selection. If you made a mistake in your display selection and wish to edit the display resource again, type 1 or `display`; otherwise press ENTER to select the default option (`keyboard`). If you select the `keyboard` option, your screen will look similar to this:

```
1   AT     Standard PC/AT keyboard
2   WY60   Wyse 60 terminal in scancode mode
(select 'q' to quit)

Enter selection [AT]:
```

27. Type 1 or press ENTER to select the `AT` keyboard type. Your screen will look similar to this:

```
Enter keyboard type:
        1       101 keys (with separate arrow keys)
        2       84 keys  (escape key above number pad)

Enter selection [101]:
```

28. Type the option that corresponds to the number of keys on your IBM* AT*-style keyboard. You can press ENTER to accept the default (101). If you do not know whether you

have an 84- or 101-key keyboard, you can use the following scheme to determine which type you have:

- If your keyboard has arrow keys that are separate from the ones on the numeric keypad, then you have a 101-key keyboard.
- If the arrow keys on your keyboard are located on the numeric keypad only, then you have an 84-key keyboard.

29. After you have made your selection, your screen will look similar to this:

```
Enter keyboard device name (from the manual entry)
[/dev/console:/dev/vt%02d]:
```

30. Press ENTER to select the default keyboard device name, unless you are using a SunRiver display.

    If you are using a SunRiver display, the keyboard device name for the first workstation is `/dev/st00:/dev/st0%d`. A second SunRiver workstation would have `/dev/st10:/dev/st1%d` specified as the device name, etc. Refer to *Xvga*(1) in this guide for additional information on SunRiver device names. For information on the SunRiver keyboard display driver and SunRiver serial ports, refer to *skd*(7) and *sasy*(7) in the *INTERACTIVE UNIX System User's/System Administrator's Reference Manual*.

    After selecting the keyboard device name, your screen will look similar to this:

```
Select resource to add:
        1       display
        2       keyboard
        3       mouse
        4       tablet
(select '1' to list, 'q' to quit)

Enter selection [mouse]:
```

31. The keyboard resource has now been configured. The next resource, `mouse`, has been automatically selected as the next default selection. If you made a mistake in your keyboard selection and wish to edit the keyboard resource again, type 2 or `keyboard`; otherwise press ENTER to select the default option (`mouse`).

☛ Due to hardware considerations, in certain cases, the configuration information you choose for a mouse might not be related to the actual brand name of the mouse you have. For example, the LOGITECH* two-button serial mouse emulates a Microsoft* serial mouse. Hence the configuration type you would choose for the LOGITECH two-button serial mouse would be `MS-S` (the Microsoft serial mouse type).

If you install a COMPAQ* built-in mouse driver on some AT systems that do not have a built-in mouse, your system may not boot. Only install the built-in mouse driver if your machine has a built-in mouse.

Refer to *Xconfig*(5) for additional information.

(If you will be using a tablet instead of a mouse, type `tablet` and skip to step 36 to add the tablet resource to your configuration.) If you select the `mouse` option, your screen will look similar to this:

```
Select mouse type:
        1       LOGI-S     Logitech Serial Mouse
        2       LOGI-B     Logitech Bus Mouse
        3       MS-S       Microsoft Serial Mouse
        4       MS-B       Microsoft Bus Mouse
        5       MSC-S      Mouse Systems 3-button Serial Mouse
        6       MSC-B      Mouse Systems 3-button Bus Mouse
        7       OMNI       MSC OmniMouse
        8       PS/2       IBM PS/2 on-board mouse
        9       COMPAQ     Compaq on-board mouse
(select 'q' to quit)

Enter selection [LOGI-S]:
```

32. Type the option number or the name of the mouse you plan to use, or press ENTER to select the default (`LOGI-S Logitech Serial Mouse`). If you select the LOGITECH Serial Mouse, your screen will look similar to this:

```
Select Logitech Mouse baud rate:
        1       1200 baud
        2       2400 baud
        3       4800 baud
        4       9600 baud

Enter selection [1200]:
```

33. Type the option number or baud rate that corresponds to the baud rate you plan to use, or press ENTER to select the default baud rate (`1200`). Your screen will look similar to this:

```
Select number of buttons on mouse:
        1       1 button
        2       2 buttons
        3       3 buttons

Enter selection [3]:
```

34. Type the option number that corresponds to the number of buttons on your mouse, or press ENTER to select the default (3). Your screen will look similar to this:

```
Enter mouse device name [/dev/tty00]:
```

35. Type the mouse device name, or press ENTER to select the default (`/dev/tty00`) (serial port 0, i.e., COM1).

    To select serial port 1, i.e., COM2, enter `/dev/tty01` as the mouse device name.

    If you are using the SunRiver mouse devices, the names are `/dev/ser00` for the first workstation, `/dev/ser10` for the second workstation, etc. For information on SunRiver serial ports, refer to *sasy*(7) in the *INTERACTIVE UNIX System User's/System Administrator's Reference Manual*.

    For information on other serial ports, refer to the section "HARDWARE COMPATIBILITY AND CONFIGURATION" in the "INTERACTIVE UNIX Operating System Maintenance Procedures" in the *INTERACTIVE UNIX Operating System Guide*.

    Your screen will look similar to this:

```
Select resource to add:
        1       display
        2       keyboard
        3       mouse
        4       tablet
(select 'l' to list, 'q' to quit)

Enter selection [tablet]:
```

36. The mouse resource has now been configured. The next resource, `tablet`, has been automatically selected as the next default selection. If you made a mistake in your mouse selection and wish to edit the mouse resource again, type 3 or `mouse`; otherwise press ENTER to select the default option (`tablet`). If you select the `tablet` option, your screen will look similar to this:

```
Select tablet type:
    1    SUMMA    Summagraphics SummaSketch Plus tablet
(select 'q' to quit)

Enter selection [SUMMA]:
```

37. Type 1 or press ENTER to select the Summagraphics* SummaSketch* Plus tablet or a compatible. You will then see a series of displays for configuring the tablet you have chosen. First, you will be asked to select the tablet model and its orientation. Your screen will look similar to this:

```
Select tablet model and orientation:
  1  MM961 (6x9" tablet) oriented vertically
  2  MM961 (6x9" tablet) oriented horizontally
  3  MM1201 (12x12" tablet) [in any position]

Enter selection [3]:
```

38. Type the number that corresponds to your selection, or press ENTER to accept the default.

    You will then be asked to select the type of pointer you will be using. Your screen will look similar to this:

```
Select type of pointer installed:
  1  4-button cursor (puck)
  2  2-button stylus (pen)

Enter selection [1]:
```

39. Type the number that corresponds to your selection, or press ENTER to accept the default. You will then be asked to select the input mode. Your screen will look similar to this:

```
Select input mode:
  1  absolute mode (conventional tablet)
  2  relative mode (like a mouse)

Enter selection [1]:
```

40. Type the number that corresponds to your selection, or press ENTER to accept the default. You will then be asked to select the lines of resolution. Your screen will look similar to this:

```
Select lines of resolution:
  1  1000 lines per inch
  2  500  lpi
  3  400  lpi
  4  200  lpi
  5  100  lpi
  6  40 lines per mm (1016 lpi)
  7  20 lpmm (508 lpi)
  8  10 lpmm (254 lpi)

Enter selection [1000]:
```

41. Type the number that corresponds to your selection, or press ENTER to accept the default. You will then be asked to select the report rate divisor. Your screen will look similar to this:

```
Select report rate divisor:
  1  maximum throughput (fast systems only)
  2  throughput / 2
  3  throughput / 8
  4  throughput / 32

Enter selection [3]:
```

42. Type the number that corresponds to your selection, or press ENTER to accept the default. You will then be asked to enter the tablet device name. Your screen will look similar to this:

```
Enter tablet device name [/dev/tty00]:
```

43. You have now configured all of the resources for the display. Press ENTER to see a listing of all of the options you have selected. Here is a sample listing that assumes you have configured a mouse instead of a tablet; your screen will look similar to this:

```
Display "0" is now:
Resource Type    Info              Display Device
display  VGA     "640x480 16 11x8" 0 /dev/console
keyboard AT      101               0   /dev/console:/dev/vt%02d
mouse    LOGI-S  "1200 3"          0   /dev/tty00

Save these changes? [y]:
```

44. Press ENTER to save the configuration you have selected. Your screen will look similar to this:

```
Enter the operation you want to perform:

    1 list
    2 add
    3 delete

[default: q]:
```

45. Press ENTER. Your screen will look similar to this:

```
Finished configuring servers? [y]
```

46. Press ENTER if you have finished configuring the servers. Your screen will look similar to this:

```
Building /usr/bin/X11/Xvga ...
**** Server 'Xvga' has been created ****

The kernel must be built before the X11 Windowing
System may be used.
Do you want to build a kernel with kconfig
at this time? [y]
```

47. The installation of the INTERACTIVE X11 Runtime System is now complete.

    Before you can use X, you will need to build and install a new kernel that includes the X device drivers that were added when you configured the server. You should build the kernel now unless you are going to be installing another package that will also require you to rebuild the kernel. (You will not need to rebuild your kernel after installing the INTERACTIVE X11 Development System.) Press ENTER or type **y** to rebuild the system kernel:

```
Building kernel to include INTERACTIVE X11 Package

        unix.I made.

executing //etc/conf/bin/idbuild -k I -r /
Finished building unix system

Installing a new kernel requires a system re-boot.

Do you wish to install and boot the new kernel now [y/n]?
```

48. To install and boot the new kernel at this time, type **y**. Your screen will look similar to this:

```
This procedure will execute a shutdown to reboot the
new kernel unix.I.
Enter y to continue, n to terminate:
```

49. To execute a system shutdown to reboot the new kernel, type **y**. (Refer to *shutdown*(1M) in the *INTERACTIVE UNIX System User's/System Administrator's Reference Manual* and to the "INTERACTIVE UNIX Operating System Installation Instructions" in the *INTERACTIVE UNIX Operating System Guide* for more information.) You will then see the output from the `shutdown` program on your screen, and your system will be shut down. The final message will say:

```
The system is down.
Press any key to re-boot.
```

50. Before pressing any key to reboot, you must remove the INTERACTIVE X11 Version 2.0 Runtime System diskette. After rebooting, you will see a login prompt.

4.3 Installing the Development System

The Development System must be installed if you plan to develop X clients. You do not need this system package if you are only using the X environment. You should install this system package after the Runtime System has been installed.

To install the Development System software on an INTERACTIVE UNIX System:

1. Use the `sysadm` command or log in as `sysadm` and access the Main menu. Your screen will look similar to this:

```
                    SYSTEM ADMINISTRATION

  1 diskmgmt        disk management menu
  2 filemgmt        file management menu
  3 machinemgmt     machine management menu
  4 packagemgmt     package management menu
  5 softwaremgmt    software management menu
  6 syssetup        system setup menu
  7 ttymgmt         tty management menu
  8 usermgmt        user management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  q to QUIT:
```

2. Select option 5 to access the Software Management menu. Your screen will look similar to this:

```
                    SOFTWARE MANAGEMENT

  1 installpkg      install new software package onto built-in disk
  2 listpkg         list packages already installed
  3 removepkg       remove previously installed package from built-in disk
  4 runpkg          run software package without installing it

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 1, `installpkg`.

   The system will then prompt you for the density of the diskettes you will use for the installation. Your screen will look similar to this:

```
Enter density of the diskettes you are installing from:
  1) 1.2  MB  ( 5 1/4" high density )
  2) 1.44 MB  ( 3 1/2" high density )
  3) 360  Kb  ( 5 1/4" low density )
  4) 720  Kb  ( 3 1/2" low density )

Please enter #(1-4), default 1:
```

4. Type the number that corresponds to the type of diskettes you are using for the installation. The system will then ask you to insert the first diskette of the subsystem into the diskette drive. Your screen will look similar to this:

```
Confirm

Please insert the floppy disk

If the program installation requires more than one floppy
disk, be sure to insert the disks in the proper order,
starting with disk number 1.
After the first floppy disk, instructions will be provided
for inserting the remaining floppy disks.

Strike ENTER when ready
or ESC to stop
```

5. If you wish to stop the installation process and return to the system prompt, press **ESC**. To proceed with the installation, insert the first *Development System* diskette into the drive and press **ENTER**. The following message will be displayed:

```
Installation is in progress -- do not remove the floppy disk.
Install the INTERACTIVE X11: Development System -
Version 2.0 package? (y):
```

6. Type `y` to proceed with the installation of the Development System package. (You can type `n` to stop the installation.) If you type `y`, your screen will look similar to this:

```
Installing the INTERACTIVE X11: Development System -
Version 2.0.
Copyright (c) 1988-1991 Interactive Systems Corporation
All Rights Reserved

Derived from X11R4 sources distributed by MIT
Copyright 1985-1989 by the Massachusetts Institute of
Technology and others.  Refer to /usr/lib/X11/LABEL for
a complete list.

The following files are being installed:
```

7. A list of the files being copied onto your system from the first diskette will accompany the above message. When all of the files from the first diskette have been copied, the system will prompt you to insert any additional diskettes.

8. When all of the diskettes have been copied, the installation is complete. The following message will display:

```
Floppy diskette number 1 is complete

Confirm

Please insert the floppy disk.

If the program installation requires more than one floppy
disk, be sure to insert the disks in the proper order,
starting with disk number 1.
After the first floppy disk, instructions will be provided
for inserting the remaining floppy disks.

Strike ENTER when ready
or ESC to stop.
```

9. The above message indicates that the installation of the INTERACTIVE X11 Development System has completed successfully. Press ESC to stop. Your screen will look similar to this:

```
Press the ENTER key to see the softwaremgmt menu [?, ^, q]:
```

10. Press ENTER to return to the Software Management menu, or type `q` to exit `sysadm`.

4.4 Installing the Contributed Software

The Contributed Software is a single subsystem. You can install it at any time after you have installed the Runtime System. This system package contains "demo" clients and a large number of fonts. The fonts are in the directory `/usr/lib/X11/fonts`.

☛ Refer to *xset*(1) for information on adding new font directories to your font path. One example for adding a font to your font path is given at the end of this section.

To install the Contributed Software on an INTERACTIVE UNIX System:

1. Use the `sysadm` command or log in as `sysadm` and access the Main menu. Your screen will look similar to this:

```
                    SYSTEM ADMINISTRATION

   1 diskmgmt        disk management menu
   2 filemgmt        file management menu
   3 machinemgmt     machine management menu
   4 packagemgmt     package management menu
   5 softwaremgmt    software management menu
   6 syssetup        system setup menu
   7 ttymgmt         tty management menu
   8 usermgmt        user management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  q to QUIT:
```

2. Select option 5 to access the Software Management menu. Your screen will look similar to this:

```
                    SOFTWARE MANAGEMENT

   1 installpkg      install new software package onto built-in disk
   2 listpkg         list packages already installed
   3 removepkg       remove previously installed package from built-in disk
   4 runpkg          run software package without installing it

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 1, `installpkg`.

   The system will then prompt you for the density of the diskettes you will use for the installation. Your screen will look similar to this:

```
Enter density of the diskettes you are installing from:
   1) 1.2  MB  ( 5 1/4" high density )
   2) 1.44 MB  ( 3 1/2" high density )
   3) 360  Kb  ( 5 1/4" low density )
   4) 720  Kb  ( 3 1/2" low density )

Please enter #(1-4), default 1:
```

4. Type the number that corresponds to the type of diskettes you are using for the installation. The system will then ask you to insert the first diskette of the subsystem into the diskette drive. Your screen will look similar to this:

```
Confirm

Please insert the floppy disk

If the program installation requires more than one floppy
disk, be sure to insert the disks in the proper order,
starting with disk number 1.
After the first floppy disk, instructions will be provided
for inserting the remaining floppy disks.

Strike ENTER when ready
or ESC to stop
```

5. If you wish to stop the installation process and return to the system prompt, press ESC. To proceed with the installation, insert the first *Contributed Software* diskette into the drive and press ENTER. The following message will be displayed:

```
Installation is in progress -- do not remove the floppy disk.
Install the INTERACTIVE X11: Contributed Software -
Version 2.0 package? (y):
```

6. Type **y** to proceed with the installation of the Contributed Software package. (You can type **n** to stop the installation.) If you type **y**, your screen will look similar to this:

```
Installing the INTERACTIVE X11: Contributed Software -
Version 2.0.
Copyright (c) 1988-1991 Interactive Systems Corporation
All Rights Reserved

Derived from X11R4 sources distributed by MIT
Copyright 1985-1989 by the Massachusetts Institute of
Technology and others.  Refer to /usr/lib/X11/LABEL for
a complete list.

The following files are being installed:
```

7. A list of the files being copied onto your system from the first diskette will accompany the above message. When all of the files from the first diskette have been copied, the system will prompt you to insert the second diskette:

```
Floppy diskette number 1 is complete
Remove floppy and insert floppy number 2
Type <enter> when ready:
```

8. Insert the second diskette in the diskette drive and press ENTER. A list of the files being copied onto your system from the second diskette will be displayed on your screen.

9. Continue inserting diskettes until all of the diskettes have been copied. When the last diskette has been copied, the following message will display:

```
Floppy diskette number 3 is complete
Installation of the INTERACTIVE X11: Contributed
Software - Version 2.0 is complete.

Confirm

Please insert the floppy disk.

If the program installation requires more than one floppy
disk, be sure to insert the disks in the proper order,
starting with disk number 1.
After the first floppy disk, instructions will be provided
for inserting the remaining floppy disks.

Strike ENTER when ready
or ESC to stop.
```

10. The above message indicates that the installation of the Contributed Software has completed successfully. Press ESC to stop. Your screen will look similar to this:

```
Press the ENTER key to see the softwaremgmt menu [?, ^, q]:
```

11. Press ENTER to return to the Software Management menu, or type `q` to exit `sysadm`.

☛ You can add fonts distributed with the Contributed Software to your font path. For example, to add the `oldx11` font to your font path, exit the `sysadm` program. At the command line (`$` prompt), type:

```
xset fp+ /usr/lib/X11/fonts/oldx11
xset fp rehash
```

5. CONFIGURATION AND MANAGEMENT PROCEDURES

For additional information on X11 configuration procedures, refer to the "Display Names" section of *X*(1).

5.1 Per-User X11 Configuration

The `addxuser` and `delxuser` functions of `sysadm` allow you to automatically configure a user's environment for INTERACTIVE X11.

5.1.1 Adding X11 Users

To enable a user to use X:

1. Use the `sysadm` command or log in as `sysadm` and access the Main menu. Your screen will look similar to this:

```
                        SYSTEM ADMINISTRATION

   1 diskmgmt        disk management menu
   2 filemgmt        file management menu
   3 machinemgmt     machine management menu
   4 packagemgmt     package management menu
   5 softwaremgmt    software management menu
   6 syssetup        system setup menu
   7 ttymgmt         tty management menu
   8 usermgmt        user management menu

 Enter a number, a name, the initial part of a name, or
 ? or <number>? for HELP,  q to QUIT:
```

2. Select option 4 to access the Package Management menu. Your screen will look similar to this:

```
                        PACKAGE MANAGEMENT

  1 lpmgmt          add line printer
  2 tcpipmgmt       extended networking utilities menu
  3 xmgmt           X Window System utilities management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 3 to access the X Window System Management (`xmgmt`) menu. Your screen will look similar to this:

```
                    X WINDOW SYSTEM MANAGEMENT

  1 addxuser        enable users to use the X Window System
  2 configmgmt      manage display configuration entries (list, add, delete)
  3 delxuser        disable users from using the X Window System
  4 hostmgmt        manage /etc/X?.hosts entries (list, add, delete)
  5 servermgmt      manage X11 Servers (list, add, delete)
  6 xdrivermgmt     manage X11 kernel drivers (list, add, remove)

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the default.
```

4. Select option 1 to enable a user to use the X Window System*. Your screen will look similar to this:

```
Do you wish to set up a new X user? [y, n, ?, q]
```

5. Type y to set up a new X user. Your screen will look similar to this:

```
Enter the User's login ID :
```

6. Type the login name of the user you wish to add. Your screen will look similar to this:

```
Enter the window manager you want to use:
  1 mwm
  2 twm
  3 uwm

[default=1]
```

7. Enter the number that corresponds to the type of window manager the new user will use, or press ENTER to accept the default, the Motif* Window Manager mwm.

   The display below shows a sample entry for a user with the login name rws who is using the Motif Window Manager. The name rws will be replaced with the login name of the user you want to add. Your screen will look similar to this:

```
Installing /usr/rws/.mwmrc
Installing /usr/rws/.xdefaults
Modifying /usr/rws/.profile

User rws can now use the X Window System

Do you wish to set up a new X user? [y, n, ?, q]
```

8. Type y to set up additional users, or type q to quit. If you type q, your screen will look similar to this:

```
Press the ENTER key to see the xmgmt menu [?, ^, q]:
```

9. Press ENTER to return to the X Window System Management menu, or type `q` to exit `sysadm`.

5.1.2 Deleting X11 Users

To disable a user's ability to run X:

1. Use the `sysadm` command or log in as `sysadm` and access the Main menu. Your screen will look similar to this:

```
                        SYSTEM ADMINISTRATION

   1 diskmgmt        disk management menu
   2 filemgmt        file management menu
   3 machinemgmt     machine management menu
   4 packagemgmt     package management menu
   5 softwaremgmt    software management menu
   6 syssetup        system setup menu
   7 ttymgmt         tty management menu
   8 usermgmt        user management menu

 Enter a number, a name, the initial part of a name, or
 ? or <number>? for HELP,  q to QUIT:
```

2. Select option 4 to access the Package Management menu. Your screen will look similar to this:

```
                        PACKAGE MANAGEMENT

  1 lpmgmt          add line printer
  2 tcpipmgmt       extended networking utilities menu
  3 xmgmt           X Window System utilities management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 3 to access the X Window System Management menu. Your screen will look similar to this:

```
                    X WINDOW SYSTEM MANAGEMENT

  1 addxuser        enable users to use the X Window System
  2 configmgmt      manage display configuration entries (list, add, delete)
  3 delxuser        disable users from using the X Window System
  4 hostmgmt        manage /etc/X?.hosts entries (list, add, delete)
  5 servermgmt      manage X11 Servers (list, add, delete)
  6 xdrivermgmt     manage X11 kernel drivers (list, add, remove)

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the default.
```

4. Select option 3 to disable a user's ability to use the X Window System. Your screen will look similar to this:

   ```
   Do you wish to disable an X user's ability to run X?
   [y, n, ?, q]
   ```

5. Type `y` to disable an X user's ability to run X. Your screen will look similar to this:

   ```
   Enter the User's login ID :
   ```

6. Now type the login name of the user who will no longer be running X. The display below shows a sample entry for the login name `rws`. The name `rws` will be replaced with the login name of the user you want to delete. Your screen will look similar to this:

   ```
   rws's /usr/rws/.Xdefaults and .uwmrc files will now be removed.
   Proceed? [y, n, ?, q]
   ```

7. The system asks you to confirm that you want to disable this user's ability to run X. If you type `y`, your screen will look similar to this:

   ```
   User rws can no longer use the X Window System

   Do you wish to disable an X user's ability to run X?
   [y, n, ?, q]
   ```

8. Type `y` to disable another user's ability to run X, or type `q` to quit. If you type `q`, your screen will look similar to this:

   ```
   Press the ENTER key to see the xmgmt menu [?, ^, q]:
   ```

9. Press ENTER to return to the X Window System Management menu, or type `q` to exit `sysadm`.

5.2 X Host Management

If you are running INTERACTIVE TCP/IP, X Host Management allows you to manage the list of X hosts that are allowed to connect across the net to your display server. Refer to *xhost*(1) for additional information.

To access the X Host Management menu:

1. Use the `sysadm` command or log in as `sysadm` and access the Main menu. Your screen will look similar to this:

```
                        SYSTEM ADMINISTRATION

   1 diskmgmt        disk management menu
   2 filemgmt        file management menu
   3 machinemgmt     machine management menu
   4 packagemgmt     package management menu
   5 softwaremgmt    software management menu
   6 syssetup        system setup menu
   7 ttymgmt         tty management menu
   8 usermgmt        user management menu

 Enter a number, a name, the initial part of a name, or
 ? or <number>? for HELP,  q to QUIT:
```

2. Select option 4 to access the Package Management menu. Your screen will look similar to this:

```
                        PACKAGE MANAGEMENT

  1 lpmgmt          add line printer
  2 tcpipmgmt       extended networking utilities menu
  3 xmgmt           X Window System utilities management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 3 to access the X Window System Management menu. Your screen will look similar to this:

```
                    X WINDOW SYSTEM MANAGEMENT

  1 addxuser        enable users to use the X Window System
  2 configmgmt      manage display configuration entries (list, add, delete)
  3 delxuser        disable users from using the X Window System
  4 hostmgmt        manage /etc/X?.hosts entries (list, add, delete)
  5 servermgmt      manage X11 Servers (list, add, delete)
  6 xdrivermgmt     manage X11 kernel drivers (list, add, remove)

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the default.
```

4. Select option 4 to manage `/etc/X?.hosts` entries. Your screen will look similar to this:

```
This procedure is used to list, add, and delete entries in
the X Window System '/etc/X0.hosts' file.

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the
default.

Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

5.2.1 Listing Remote Hosts

To list the remote hosts that are currently allowed access to a display server:

1. First access the `hostmgmt` menu item of `sysadm` as shown in steps 1-4 under section 5.2. Your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

2. Type 1 to see a listing of host entries for a display. Your screen will look similar to this:

```
Enter the number of the display [0]:
```

3. Type `0` (the default display server) or the number of the display for which you want a list. The display below shows a sample entry for display number `0`. A host named `expo.lcs.mit.edu` is listed. The display number and host name shown will be replaced with the number and list of host names for the display you choose to list. Your screen will look similar to this:

```
This is the current list of hosts allowed access to the X
Window System on display 0:
expo.lcs.mit.edu

Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

4. Type `q` to quit or `1` to list another display. If you type `q`, your screen will look similar to this:

```
Press the ENTER key to see the xmgmt menu [?, ^, q]:
```

5. Press ENTER to return to the X Window System Management menu, or type `q` to exit `sysadm`.

5.2.2 Adding Remote Hosts

To enable a remote host to have network access to your display server:

1. First access the `hostmgmt` menu item (option `4`) of `sysadm` as shown in steps 1-4 under section 5.2. Your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

2. Type `2` to add a host entry. Your screen will look similar to this:

```
This procedure is used to create entries in the
X Window System '/etc/X0.hosts' file.
This file contains information which X uses to determine
which remote hosts are allowed to access the display server.

Enter the number of the display [0]:
```

3. Type 0 (the default display server), or type the number of the display for which you wish to add a host entry. Your screen will look similar to this:

```
Enter the name of the host you want to add:
```

4. Type the name of the host you want to have access to this display server. The display below shows a sample entry for a host named `expo.lcs.mit.edu`. The name `expo.lcs.mit.edu` will be replaced with the name of the host you want to add. Your screen will look similar to this:

```
Here is the entry for machine 'expo.lcs.mit.edu.'

SYSTEM NAME:     expo.lcs.mit.edu

Should this be entered into the '/etc/X0.hosts' file?
[y, n, q]
```

5. The system asks you to confirm that you want to add this entry. If you type `y` to add the entry, your screen will look similar to this:

```
* host 'expo.lcs.mit.edu' has been added to '/etc/X0.hosts' *

Add another entry to the /etc/X0.hosts file? [y, n, q]
```

6. Type `y` to add another entry, or type `q` to quit adding entries for this display. If you type `q`, your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

7. Type `q` to quit, or type `1` to add a host to another display. If you type `q`, your screen will look similar to this:

```
Press the ENTER key to see the xmgmt menu [?, ^, q]:
```

8. Press ENTER to return to the X Window System Management menu, or type `q` to exit `sysadm`.

5.2.3 Deleting Remote Hosts

To remove a remote host's entry in the file `/etc/X0.hosts`, so that the host will no longer have network access to your server:

1. First access the `hostmgmt` menu item of `sysadm` as shown in steps 1-4 under section 5.2. Your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

2. Type 3 to delete a host entry. Your screen will look similar to this:

```
This procedure is used to remove entries in the
X Window System '/etc/X0.hosts' file.
This file contains information which X uses to determine which
remote hosts are allowed to access the display server.

Enter the number of the display [0]:
```

3. Type 0 (the default display server) or the number of the display for which you want to delete a host entry. The display below shows a sample entry for a host named `expo.lcs.mit.edu`. The name `expo.lcs.mit.edu` will be replaced with the name of the host you want to delete. Your screen will look similar to this:

```
This is the current list of host entries known to the X Window
System:

expo.lcs.mit.edu

Which host entry do you want to delete? [q]
```

4. Type the name of the host entry you wish to delete. Your screen will look similar to this:

```
Here is the file entry for machine 'expo.lcs.mit.edu'.

SYSTEM NAME:     expo.lcs.mit.edu
Do you want to delete the host entry named 'expo.lcs.mit.edu'?
[y, n, ?, q]
```

5. The system asks you to confirm that you want to delete this entry. If you type y to delete the entry, your screen will look similar to this:

```
* entry 'expo.lcs.mit.edu' has been deleted from
'/etc/X0.hosts' *
Any other deletions? [y, n, q]
```

6. Type `y` to delete another entry, or type `q` to quit deleting entries for this display. If you type `q`, your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

7. Type `q` to quit or `3` to delete a host from another display. If you type `q`, your screen will look similar to this:

```
Press the ENTER key to see the xmgmt menu [?, ^, q]:
```

8. Press ENTER to return to the X Window System Management menu, or type `q` to exit `sysadm`.

5.3 Display Configuration

The Display Configuration menu of `sysadm` allows you to interactively create, revise, list, and delete X11 display configuration information stored in the system configuration file. You should refer to *xconfig*(1), *Xconfig*(5), and the server-specific manual entries in this guide for additional configuration information. (The server-specific manual entries are those Section 1 entries in which the entry names begin with an uppercase X.)

To list, add, or delete X display configuration information:

1. Use the `sysadm` command or log in as `sysadm` and access the Main menu. Your screen will look similar to this:

```
                    SYSTEM ADMINISTRATION

  1 diskmgmt        disk management menu
  2 filemgmt        file management menu
  3 machinemgmt     machine management menu
  4 packagemgmt     package management menu
  5 softwaremgmt    software management menu
  6 syssetup        system setup menu
  7 ttymgmt         tty management menu
  8 usermgmt        user management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  q to QUIT:
```

2. Select option 4 to access the Package Management menu. Your screen will look similar to this:

```
                    PACKAGE MANAGEMENT

  1 lpmgmt          add line printer
  2 tcpipmgmt       extended networking utilities menu
  3 xmgmt           X Window System utilities management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 3 to access the X Window System Management menu. Your screen will look similar to this:

```
                  X WINDOW SYSTEM MANAGEMENT

  1 addxuser        enable users to use the X Window System
  2 configmgmt      manage display configuration entries (list, add, delete)
  3 delxuser        disable users from using the X Window System
  4 hostmgmt        manage /etc/X?.hosts entries (list, add, delete)
  5 servermgmt      manage X11 Servers (list, add, delete)
  6 xdrivermgmt     manage X11 kernel drivers (list, add, remove)

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the default.
```

4. Select option 2 to manage display configuration entries. Your screen will look similar to this:

```
This procedure is used to list, add, and delete entries in the
X Window System Utilities '/usr/lib/X11/Xconfig' file.

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the
default.

Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

5.3.1 Listing a Display Configuration

To list a display configuration entry:

1. First access the `configmgmt` menu item of `sysadm` as shown in steps 1-4 under section 5.3. Your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

2. Type `1` to list the display configuration. Your screen will look similar to this:

```
Enter display number to list ('l' for list, 'q' to quit) [0]:
```

3. Type the number of a display you want to list. Your screen will look similar to this:

```
Values for display "0":
Resource Type    Info              Display Device
display  VGA     "640x480 16 11x8" 0 /dev/console
keyboard AT      101               0   /dev/console:/dev/vt%02d
mouse    LOGI-S "1200 3"          0   /dev/tty00

Enter display number to list ('l' for list, 'q' to quit) [q]:
```

4. To list another entry, type a display number. If you do not wish to list another entry, type `q` to quit. If you type `q`, your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

5. Type `q` to quit. Your screen will look similar to this:

```
Press the ENTER key to see the xmgmt menu [?, ^, q]:
```

6. Press **ENTER** to return to the X Window System Management menu, or type `q` to exit `sysadm`.

5.3.2 Adding or Modifying a Display Configuration

To add or modify a display configuration entry:

1. First access the `configmgmt` menu item of `sysadm` as shown in steps 1-4 under section 5.3. Your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

2. Type **2** to add a display configuration. Your screen will look similar to this:

```
Enter display number to add ('l' for list, 'q' to quit) [0]:
```

3. Type the number of the display you want to add. For example, type **1**. Your screen will look similar to this:

```
Display "1" doesn't exist, create it? [y]:
```

4. If there is no configuration information for this screen in the system configuration file, the system will ask you to confirm that you want to create it. If the configuration information already exists, you may modify any of the resources for the screen. If you type **y**, your screen will look similar to this:

```
Create display "1":
Select resource to add:
        1       display
        2       keyboard
        3       mouse
        4       tablet
(select 'l' to list, 'q' to quit)

Enter selection [display]:
```

5. The information in this section presents an alternate configuration using the same steps that you used when you initially configured your system (see steps 15-47 in section 4). The configuration shown, adding an IBM 8514/A display adapter, could be used, for example, on a virtual terminal using VT flip. (For additional information on virtual terminals, refer to the section "USING VIRTUAL TERMINALS" in the "INTERACTIVE UNIX Operating System Primer" in the *INTERACTIVE UNIX System Guide for New Users*.) Note that only one IBM 8514/A adapter can be run at a time.

When a menu has a default selection, that choice is shown in brackets in the line at the bottom of the screen that says `Enter selection [`*selection*`]`. If you want to select the default resource (`display`), press **ENTER**. If you want to select a resource other than the `display` resource, you may type either the option number (`2`, `3`, or `4`) or the selection name (`keyboard`, `mouse`, or `tablet`). The X configuration procedures automatically cycle between the various resource options.

When configuring your resources, note that your display and the information requested will vary according to the display type you select.

☛ Note that you cannot include both a mouse and a tablet in the same configuration.

If you installed the *Display Servers I* diskette and you select `display`, your screen will look similar to this:

```
Select display type:
 1  CVC      Cornerstone Displays (and compatibles)
 2  8514     IBM 8514/A Display Adapter and compatibles
 3  HRC      Hercules monochrome display
 4  LVP      Sigma Designs LaserView
 5  V256     VGA and compatibles (256 Colors)
 6  VGA      VGA and compatibles
 7  EGA      EGA and compatibles
 8  VIKING   Moniterm Viking
(select 'q' to quit)

Enter selection [VGA]:
```

☛ If you are overlaying Version 2.0 on top of Release 1.3, your menu will include additional choices not listed above. However, only the choices shown above are available for installation with this release.

6. Type the option number or the name that corresponds to the type of display you are using. For example, to select the IBM 8514/A display adapter, type `2` or type `8514`. If you select the `8514` display type, your screen will look similar to this:

```
Select display adapter:
 1   IBM
 2   Matrox
 3   ADEX

Enter selection [IBM]:
```

7. Type the option number or the name that corresponds to the display adapter you are using. For example, to select the

IBM, adapter, type 1 or IBM, or press ENTER to accept the default. If you select the IBM display adapter, your screen will look similar to this:

```
Select screen resolution:
 1   640x480
 2   1024x768

Enter selection [1024x768]:
```

8. To select the default, 1024x768, type 2 or press ENTER. Your screen will look similar to this:

```
Select number of colors:
  1   16 colors
  2   256 colors

Enter selection [256]:
```

9. To select 16 colors, type 1. To select the default, 256 colors, type 2 or press ENTER. Your screen will look similar to this:

```
Enter screen size (in inches) in the form WIDTHxHEIGHT [11x8]:
```

10. Type the screen size you will be using, or press ENTER to select the default.

    Your screen will look similar to this:

```
Select resource to add:
        1       display
        2       keyboard
        3       mouse
        4       tablet
(select 'l' to list, 'q' to quit)

Enter selection [keyboard]:
```

11. The display resource has now been configured. The next resource, keyboard, has been automatically selected as the default selection. If you made a mistake in your display selection and wish to edit the display resource again, type 1 or display; otherwise press ENTER to select the default option (keyboard). If you select the keyboard option, your screen will look similar to this:

```
Select keyboard type:
 1   AT     Standard PC/AT keyboard
 2   WY60   Wyse 60 terminal in scancode mode
(select 'q' to quit)

Enter selection [AT]:
```

12. Type 1 or press ENTER to select the AT keyboard type. Your screen will look similar to this:

```
Enter keyboard type:
        1       101 keys (with separate arrow keys)
        2       84 keys  (escape key above number pad)

Enter selection [101]:
```

13. Type the option that corresponds to the number of keys on your AT keyboard. You can press ENTER to accept the default (101). If you do not know whether you have an 84- or 101-key keyboard, you can use the following scheme to determine which type you have:

    - If your keyboard has arrow keys that are separate from the ones on the numeric keypad, then you have a 101-key keyboard.
    - If the arrow keys on your keyboard are located on the numeric keypad only, then you have an 84-key keyboard.

14. After you have made your selection, your screen will look similar to this:

```
Enter keyboard device name (from the manual entry)
[/dev/console:/dev/vt%02d]:
```

15. Press ENTER to select the default keyboard device name unless you are using a SunRiver display.

    If you are using a SunRiver display, the keyboard device name for the first workstation is `/dev/st00:/dev/st0%d`. A second SunRiver workstation would have `/dev/st10:/dev/st1%d` specified as the device name, etc. Refer to *Xvga*(1) in this guide for additional information. For information on the SunRiver keyboard display driver and SunRiver serial ports, refer to *skd*(7) and *sasy*(7) in the *INTERACTIVE UNIX System User's/System Administrator's Reference Manual*.

    After selecting the keyboard device name, your screen will look similar to this:

```
Select resource to add:
        1       display
        2       keyboard
        3       mouse
        4       tablet
(select 'l' to list, 'q' to quit)

Enter selection [mouse]:
```

16. The keyboard resource has now been configured. The next resource, `mouse`, has been automatically selected as the

default selection. If you made a mistake in your keyboard selection and wish to edit the keyboard resource again, type 2 or `keyboard`; otherwise press ENTER to select the default option (`mouse`). (If you will be using a tablet instead of a mouse, type `tablet` and skip to step 21 to add the tablet resource to your configuration.) If you select the `mouse` option, your screen will look similar to this:

```
Select mouse type:
        1       LOGI-S      Logitech Serial Mouse
        2       LOGI-B      Logitech Bus Mouse
        3       MS-S        Microsoft Serial Mouse
        4       MS-B        Microsoft Bus Mouse
        5       MSC-S       Mouse Systems Serial Mouse
        6       MSC-B       Mouse Systems Bus Mouse
        7       OMNI        MSC OmniMouse
        8       PS/2        IBM PS/2 on-board mouse
        9       COMPAQ      Compaq on-board mouse
(select 'q' to quit)

Enter selection [LOGI-S]:
```

17. Type the option number or the name of the mouse you plan to use, or press ENTER to select the default (`LOGI-S`). If you select the LOGITECH Serial Mouse, your screen will look similar to this:

```
Select Logitech Mouse baud rate:
        1       1200 baud
        2       2400 baud
        3       4800 baud
        4       9600 baud

Enter selection [1200]:
```

18. Type the option number or the baud rate you plan to use, or press ENTER to select the default (`1200`). Your screen will look similar to this:

```
Select number of buttons on mouse:
        1       1 button
        2       2 buttons
        3       3 buttons

Enter selection [3]:
```

19. Type the option number that corresponds to the number of buttons on your mouse, or press ENTER to select the default (`3`). Your screen will look similar to this:

```
Enter mouse device name [/dev/tty00]:
```

20. Type the mouse device name, or press ENTER to select the default (`/dev/tty00`) (serial port 0, that is, COM1).

To select serial port 1, i.e., COM2, enter `/dev/tty01` as the mouse device name. (If you are using the SunRiver mouse devices, the names are `/dev/ser00` for the first workstation, `/dev/ser10` for the second workstation, etc. For information on SunRiver serial ports, refer to *sasy*(7) in the *INTERACTIVE UNIX System User's/System Administrator's Reference Manual*.)

Your screen will look similar to this:

```
Select resource to add:
        1       display
        2       keyboard
        3       mouse
        4       tablet
(select '1' to list, 'q' to quit)

Enter selection [tablet]:
```

21. The mouse resource has now been configured. The next resource, `tablet`, has been automatically selected as the default selection. If you made a mistake in your mouse selection and wish to edit the mouse resource again, type `3` or `mouse`; otherwise press **ENTER** to select the default option (`tablet`). If you select the `tablet` option, your screen will look similar to this:

```
Select tablet type:
    1    SUMMA    Summagraphics SummaSketch Plus tablet
(select 'q' to quit)

Enter selection [SUMMA]:
```

22. Type `1` or press **ENTER** to select the Summagraphics SummaSketch Plus tablet or a compatible. You will then see a series of displays for configuring the tablet you have chosen. First, you will be asked to select the tablet model and its orientation. Your screen will look similar to this:

```
Select tablet model and orientation:
  1  MM961 (6x9" tablet) oriented vertically
  2  MM961 (6x9" tablet) oriented horizontally
  3  MM1201 (12x12" tablet) [in any position]

Enter selection [3]:
```

23. Type the number that corresponds to your selection, or press **ENTER** to accept the default.

You will then be asked to select the type of pointer you will be using. Your screen will look similar to this:

```
Select type of pointer installed:
  1  4-button cursor (puck)
  2  2-button stylus (pen)

Enter selection [1]:
```

24. Type the number that corresponds to your selection, or press ENTER to accept the default. You will then be asked to select the input mode. Your screen will look similar to this:

```
Select input mode:
  1  absolute mode (conventional tablet)
  2  relative mode (like a mouse)

Enter selection [1]:
```

25. Type the number that corresponds to your selection, or press ENTER to accept the default. You will then be asked to select the lines of resolution. Your screen will look similar to this:

```
Select lines of resolution:
  1  1000 lines per inch
  2  500  lpi
  3  400  lpi
  4  200  lpi
  5  100  lpi
  6  40 lines per mm (1016 lpi)
  7  20 lpmm (508 lpi)
  8  10 lpmm (254 lpi)

Enter selection [1000]:
```

26. Type the number that corresponds to your selection, or press ENTER to accept the default. You will then be asked to select the report rate divisor. Your screen will look similar to this:

```
Select report rate divisor:
  1  maximum throughput (fast systems only)
  2  throughput / 2
  3  throughput / 8
  4  throughput / 32

Enter selection [3]:
```

27. Type the number that corresponds to your selection, or press ENTER to accept the default. You will then be asked to enter the tablet device name. Your screen will look similar to this:

```
Enter tablet device name [/dev/tty00]:
```

28. You have now configured all of the resources for the display. Press ENTER to see a listing of all of the options you have selected. Here is a sample listing that assumes you have

configured a mouse instead of a tablet; your screen will look similar to this:

```
Display "1" is now:
Resource Type    Info              Display Device
display  8514    "IBM 43 1024x768 256 11X8" 1 /dev/console
keyboard AT      101               1     /dev/console:/dev/vt%02d
mouse    LOGI-S "1200 3"           1    /dev/tty00

Save these changes? [y]:
```

29. Press ENTER to save the configuration you have selected. Your screen will look similar to this:

```
Enter display number ('l' for list, 'q' to quit) [q]:
```

30. To add another entry, type a display number; or type `q` to quit. If you type `q`, your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

31. Type `q` to quit. Your screen will look similar to this:

```
Press the ENTER key to see the xmgmt menu [?, ^, q]:
```

32. Press ENTER to return to the X Window System Management menu, or type `q` to exit `sysadm`.

5.3.3 Deleting a Display Configuration

To delete a display configuration entry:

1. First access the `configmgmt` menu item of `sysadm` as shown in steps 1-4 under section 5.3. Your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

2. Type `3` to delete a display configuration. Your screen will look similar to this:

```
Enter display number to delete ('l' for list, 'q' to quit) [0]:
```

3. Type the number of the display you want to delete. If you type `1`, your screen will look similar to this:

```
Values for display "1":
Resource Type    Info              Display Device
display  8514    "IBM 43 1024x768 256 11X8" 1 /dev/console
keyboard AT      101               1     /dev/console:/dev/vt%02d
mouse    LOGI-S "1200 3"           1    /dev/tty00

Delete display "1"? [y]:
```

4. The system asks you to confirm that you want to delete this entry. Press ENTER or type **y** to delete the entry; type **n** if you do not want to delete this entry. If you type **y**, your screen will look similar to this:

```
Display "1" has been deleted.
Enter display number to delete ('l' for list, 'q' to quit) [q]:
```

5. To delete another entry, type a display number; or type **q** to quit. If you type **q**, your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

6. Type **q** to quit. Your screen will look similar to this:

```
Press the ENTER key to see the xmgmt menu [?, ^, q]:
```

7. Press ENTER to return to the X Window System Management menu, or type **q** to exit **sysadm**.

5.4 Server Management

To list, add, or delete an X server:

1. Use the **sysadm** command or log in as **sysadm** and access the Main menu. Your screen will look similar to this:

```
                        SYSTEM ADMINISTRATION

   1 diskmgmt        disk management menu
   2 filemgmt        file management menu
   3 machinemgmt     machine management menu
   4 packagemgmt     package management menu
   5 softwaremgmt    software management menu
   6 syssetup        system setup menu
   7 ttymgmt         tty management menu
   8 usermgmt        user management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  q to QUIT:
```

2. Select option 4 to access the Package Management menu. Your screen will look similar to this:

```
                    PACKAGE MANAGEMENT

  1 lpmgmt          add line printer
  2 tcpipmgmt       extended networking utilities menu
  3 xmgmt           X Window System utilities management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 3 to access the X Window System Management menu. Your screen will look similar to this:

```
                  X WINDOW SYSTEM MANAGEMENT

  1 addxuser        enable users to use the X Window System
  2 configmgmt      manage display configuration entries (list, add, delete)
  3 delxuser        disable users from using the X Window System
  4 hostmgmt        manage /etc/X?.hosts entries (list, add, delete)
  5 servermgmt      manage X11 Servers (list, add, delete)
  6 xdrivermgmt     manage X11 kernel drivers (list, add, remove)

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the default.
```

4. Select option 5 to manage X11 Servers. Your screen will look similar to this:

```
This procedure is used to list, add, and delete X11 servers
for use.

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the
default.

Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

5.4.1 Listing Servers

To see a listing of the servers you have created on your system:

1. First access the `servermgmt` menu item of `sysadm` as outlined above (see section 5.4). Your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

2. Type `1` to see a list of servers you have created. Your screen will look similar to this:

```
This procedure is used to list X11 servers.

The following is a list of X11 servers:

Name      built Description
----      ----- -----------
Xcvc            Server for Cornerstone displays
Xgp             Server for IBM 8514/A and compatible displays
Xhrc            Server for Hercules Monochrome Graphics
                  displays
Xhrcdemo        Demonstration server for Hercules Monochrome
                  Graphics display
Xlvp            Server for Sigma Designs LaserView Plus
                  displays
Xv256           Server for 256 color VGA and compatible
                  displays
Xvga      yes   Server for VGA/EGA and compatible displays
Xviking         Server for Moniterm Viking 21/91 Display

Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

3. Type `q` to quit. Your screen will look similar to this:

```
Press the ENTER key to see the xmgmt menu [?, ^, q]:
```

4. Press ENTER to return to the X Window System Management menu, or type `q` to exit `sysadm`.

5.4.2 Adding Servers

To add a new server to your system:

1. First access the `servermgmt` menu item of `sysadm` as shown in steps 1-4 under section 5.4. Your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

2. Type `2` to add a new server to your system. Your screen will look similar to this:

```
This procedure is used to create new X11 servers.  A new X11
server must be created for each type of display.

The following is a list of the possible X11 servers to build:

Name      built Description
----      ----- -----------
Xcvc            Server for Cornerstone displays
Xgp             Server for IBM 8514/A and compatible displays
Xhrc            Server for Hercules Monochrome Graphics
                  displays
Xhrcdemo        Demonstration server for Hercules Monochrome
                  Graphics display
Xlvp            Server for Sigma Designs LaserView Plus
                  displays
Xv256           Server for 256 color VGA and compatible
                  displays
Xvga      yes   Server for VGA/EGA and compatible displays
Xviking         Server for Moniterm Viking 21/91 Display

Enter the name of the server you want to create or 'q' to quit
[q]:
```

3. Type the name of the server you want to create. The display below shows a sample entry for a server named `Xgp`. Replace `Xgp` with the name of the server you want to create. Your screen will look similar to this:

```
Should server 'Xgp' be created? [y, n, q]
```

4. The system asks you to confirm that you want to create this server. If you type `y`, your screen will look like this:

```
  **** server 'Xgp' has been created ****

Create another X11 server? [y, n, q]
```

5. Type `y` to create another server, or type `q` to quit. If you type `q`, your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

6. Type `q` to quit. Your screen will look similar to this:

```
Press the ENTER key to see the xmgmt menu [?, ^, q]:
```

7. Press ENTER to return to the X Window System Management menu, or type `q` to exit `sysadm`.

5.4.3 Deleting Servers

To delete a server from your system:

1. First access the `servermgmt` menu item of `sysadm` as shown in steps 1-4 under section 5.4. Your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

2. Type `3` to delete a server that was previously created from your system. Your screen will look similar to this:

```
This procedure is used to delete X11 servers.


The following is a list of the possible X11 servers to delete:

Xgp     Server for IBM 8514/A and compatible displays
Xvga    Server for VGA/EGA and compatible displays

Enter the name of the server you want to delete:
```

3. Type the name of the server you want to delete. The display below shows a sample entry for a server named `Xgp`. Replace `Xgp` with the name of the server you want to delete. Your screen will look similar to this:

```
Should server "Xgp" be deleted? [y, n, q]
```

4. The system asks you to confirm that you want to delete this server. If you type `y`, your screen will look like this:

```
  **** server 'Xgp' has been deleted ****

Delete another X11 server? [y, n, q]
```

5. Type `y` to delete another server, or type `q` to quit. If you type `q`, your screen will look similar to this:

```
Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

6. Type `q` to quit. Your screen will look similar to this:

```
Press the ENTER key to see the xmgmt menu [?, ^, q]:
```

7. Press ENTER to return to the X Window System Management menu, or type `q` to exit `sysadm`.

5.5 Configuring X11 Drivers

Some high-resolution displays require UNIX System kernel drivers in order to operate in high-resolution modes. `xdrivermgmt` is the `sysadm` interface used to configure these drivers into the kernel. Configuration is limited to turning the driver on or off in the kernel. In order to change specification settings, such as I/O address, DMA, and shared memory address, you must edit the appropriate file in `/etc/conf/sdevice.d` and edit the `/etc/conf/cf.d/mdevice` file. Consult your hardware manual and the appropriate INTERACTIVE X11 manual entries for the correct values for these settings. When a driver is configured, checking will be done automatically to ensure that conflicts in the above paramenters do not occur. If an error does occur, you will be notified of the error but you will still be permitted to configure the driver. Before the kernel can be built, however, these conflicts must be resolved.

It is easier to list, add, and delete driver entries using `sysadm`, but the `kconfig` program can also be used. For information on `kconfig`, refer to the "INTERACTIVE UNIX Operating System Maintenance Procedures" in the *INTERACTIVE UNIX Operating System Guide*.

5.5.1 Using `sysadm` to Configure X11 Drivers

Use the `sysadm` command to list, add, or delete X display server configuration information.

5.5.1.1 Adding X11 Drivers to the Configuration Using `sysadm`.

1. On the system console, use the `sysadm` command or log in as `sysadm` and access the Main menu. Your screen will look similar to this:

```
                        SYSTEM ADMINISTRATION

   1 diskmgmt        disk management menu
   2 filemgmt        file management menu
   3 machinemgmt     machine management menu
   4 packagemgmt     package management menu
   5 softwaremgmt    software management menu
   6 syssetup        system setup menu
   7 ttymgmt         tty management menu
   8 usermgmt        user management menu

 Enter a number, a name, the initial part of a name, or
 ? or <number>? for HELP,  q to QUIT:
```

2. Select option 4 to access the Package Management menu. Your screen will look similar to this:

```
                        PACKAGE MANAGEMENT

  1 lpmgmt          add line printer
  2 tcpipmgmt       extended networking utilities menu
  3 xmgmt           X Window System utilities management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 3 to access the X Window System Management menu. Your screen will look similar to this:

```
                    X WINDOW SYSTEM MANAGEMENT

  1 addxuser        enable users to use the X Window System
  2 configmgmt      manage display configuration entries (list, add, delete)
  3 delxuser        disable users from using the X Window System
  4 hostmgmt        manage /etc/X?.hosts entries (list, add, delete)
  5 servermgmt      manage X11 Servers (list, add, delete)
  6 xdrivermgmt     manage X11 kernel drivers (list, add, remove)

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ` to GO BACK,  q to QUIT:

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the default.
```

4. Select option 6 to manage X11 kernel drivers. Your screen will look similar to this:

```
This procedure is used to list, add, and delete entries in the
kernel configuration.

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the
default.

Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

5. Type 2 to add an X11 kernel driver. Your screen will look similar to this:

```
 1        Cornerstone Memory Mapping Driver

kernel driver would you like to add?
```

Note: If you are performing a new installation (*not* overwriting the previous X11 release) and you are running a type of mouse other than serial, you will need to install the appropriate mouse driver from the Additional Drivers subset. For information on the Additional Drivers subset, refer to the "INTERACTIVE UNIX Operating System Maintenance Procedures" in the *INTERACTIVE UNIX Operating System Guide*.

6. Type the number of the kernel driver you would like to add. Type 1 to add the `Cornerstone Memory Mapping Driver`. Your screen will look similar to this:

```
Checking for interrupt conflicts
Checking for DMA channel conflicts
Checking for shared memory address conflicts

The kernel must be rebuilt in order for the drivers that
have been configured to take effect.  Would you like to build
a kernel at this time?  [y, n]
```

7. Type y to build a kernel at this time. Type n if you plan to add additional kernel drivers or remove kernel drivers from the configuration. If you type y, your screen will look similar to this:

```
The following X device drivers are configured in the kernel:
        Cornerstone Memory Mapping Driver
```

8. The `Cornerstone` driver has been added to the kernel configuration.

5.5.1.2 Removing X11 Drivers From the Configuration Using `sysadm`.

1. Use the `sysadm` command or log in as `sysadm` and access the Main menu. Your screen will look similar to this:

```
                        SYSTEM ADMINISTRATION

   1 diskmgmt        disk management menu
   2 filemgmt        file management menu
   3 machinemgmt     machine management menu
   4 packagemgmt     package management menu
   5 softwaremgmt    software management menu
   6 syssetup        system setup menu
   7 ttymgmt         tty management menu
   8 usermgmt        user management menu

 Enter a number, a name, the initial part of a name, or
 ? or <number>? for HELP,  q to QUIT:
```

2. Select option 4 to access the Package Management menu. Your screen will look similar to this:

```
                        PACKAGE MANAGEMENT

  1 lpmgmt          add line printer
  2 tcpipmgmt       extended networking utilities menu
  3 xmgmt           X Window System utilities management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 3 to access the X Window System Management menu. Your screen will look similar to this:

```
                    X WINDOW SYSTEM MANAGEMENT

  1 addxuser        enable users to use the X Window System
  2 configmgmt      manage display configuration entries (list, add, delete)
  3 delxuser        disable users from using the X Window System
  4 hostmgmt        manage /etc/X?.hosts entries (list, add, delete)
  5 servermgmt      manage X11 Servers (list, add, delete)
  6 xdrivermgmt     manage X11 kernel drivers (list, add, remove)

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the default.
```

4. Select option 6 to manage X11 kernel drivers. Your screen will look similar to this:

```
This procedure is used to list, add, and delete entries in the
kernel configuration.

Type 'q' at any time to quit the current operation.
If a '?' appears as a choice, type '?' for help.

If a default appears in the question, type <ENTER> for the
default.

Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

5. Type 3 to remove an X11 kernel driver. Your screen will look similar to this:

```
  1       Cornerstone Memory Mapping Driver

kernel driver would you like to remove?
```

6. Type the number of the kernel driver you would like to remove from the configuration. For example, to remove the `Cornerstone Memory Mapping Driver`, you would type 1. Your screen will look similar to this:

```
The kernel must be rebuilt in order for the drivers that
have been configured to take effect.  Would you like to build
a kernel at this time?  [y, n]
```

7. Type `y` to build a kernel at this time. Type `n` if you plan to add additional kernel drivers or remove kernel drivers from the configuration. If you type `y`, your screen will look similar to this:

```
The following X device drivers have been removed from kernel
        Cornerstone Memory Mapping Driver

Enter the operation you want to perform:

        1 list
        2 add
        3 delete

[default: q]:
```

8. The `Cornerstone Memory Mapping Driver` has been removed from the kernel configuration.

6. RUNNING INTERACTIVE X11

6.1 Running the Server

Before you can run X11, you will need to set up your environment for using X. Refer to section 5.1 of this document on how to run the per-user X configuration procedures.

After you have set up your environment for using X, you can run the server by typing `xinit`. For more information on running servers, refer to *xinit*(1) and *xdm*(1).

6.2 Using the Development System

The `examples` directory contains miscellaneous files, such as `Xdefaults`, and source files, like `maze.c`.

If you have the INTERACTIVE Software Development System and the INTERACTIVE X11 Development System installed, you can compile the `maze.c` example client by typing:

```
cd /tmp
cp /usr/lib/X11/examples/maze.c .
cc -o maze maze.c -lX11 -lnsl_s
```

☛ Note that you must link with `libnsl_s.a` (`-lnsl_s`) when building X client programs. This library should be appended after all the X libraries. Refer to the "INTERACTIVE TCP/IP Programmer's Supplement" in the *INTERACTIVE X11 Development System Guide* or the *INTERACTIVE TCP/IP Guide*.

7. REMOVING X11 SYSTEM PACKAGES

This section describes how to remove an X11 system package. For additional information, refer to the "INTERACTIVE UNIX Operating System Maintenance Procedures" in the *INTERACTIVE UNIX Operating System Guide*.

To remove an X11 system package from the INTERACTIVE UNIX System:

1. Use the `sysadm` command or log in as `sysadm` and access the Main menu. Your screen will look similar to this:

```
                        SYSTEM ADMINISTRATION

   1 diskmgmt        disk management menu
   2 filemgmt        file management menu
   3 machinemgmt     machine management menu
   4 packagemgmt     package management menu
   5 softwaremgmt    software management menu
   6 syssetup        system setup menu
   7 ttymgmt         tty management menu
   8 usermgmt        user management menu

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  q to QUIT:
```

2. Select option 5 to access the Software Management menu. Your screen will look similar to this:

```
                        SOFTWARE MANAGEMENT

   1 installpkg      install new software package onto built-in disk
   2 listpkg         list packages already installed
   3 removepkg       remove previously installed package from built-in disk
   4 runpkg          run software package without installing it

Enter a number, a name, the initial part of a name, or
? or <number>? for HELP,  ^ to GO BACK,  q to QUIT:
```

3. Select option 3, `removepkg`. Your screen will look similar to this:

```
 1  File Management-Version 2.2
 2  TCP Ethernet Support-Version 1.2
 3  Kernel Configuration-Version 2.2
 4  Software Development System-Version 2.2
 5  STREAMS Facilities-Version 2.2
 6  INTERACTIVE X11: Contributed Software - Version 2.0
 7  INTERACTIVE X11: Development System - Version 2.0
 8  INTERACTIVE X11: Runtime System - System and Servers -
    Version 2.0
 9  INTERACTIVE X11: Server Kit - Release 1.3
10  INTERACTIVE X11: Runtime System - Clients - Version 2.0

Select a number (1 - 10) from this list to remove or q to quit:
```

4. Type the number corresponding to the system package you want to remove. The display below shows an example using option 8, `Runtime System - System and Servers`. You may replace this with the option number that corresponds to the system package you want to remove. Your screen will look similar to this:

```
Confirm

Insert the removable medium for the INTERACTIVE X11:
Runtime System - System and Servers - Version 2.0
you wish to remove.

Strike ENTER when ready
or ESC to stop.
```

5. Press ENTER to remove this system package, or press ESC to quit. If you press ENTER, your screen will look similar to this:

```
Remove the INTERACTIVE X11: Runtime System - System and
Servers package? (y):
```

6. The system asks you to confirm that you want to remove this package. If you type `y`, a list of the files being removed will be displayed. After the package has been removed, your screen will look similar to this:

```
The INTERACTIVE X11: Runtime System - System and Servers
has been removed.

Confirm

Do you want to remove another package?

Strike ENTER when ready
or ESC to stop.
```

7. The above message indicates that the X11 package has been removed successfully. Press [ENTER] to remove another package, or press [ESC] to stop. If you press [ESC], your screen will look similar to this:

```
Press the ENTER key to see the softwaremgmt menu [?, ^, q]:
```

8. Press [ENTER] to return to the Software Management menu, or type `q` to exit `sysadm`.

8. KERNEL CONSIDERATIONS

8.1 Adding Drivers to Your INTERACTIVE UNIX System Kernel System File

You can add device drivers to your INTERACTIVE UNIX System from the Additional Drivers subset. For information on the Additional Drivers subset, refer to the "INTERACTIVE UNIX Operating System Maintenance Procedures" in the *INTERACTIVE UNIX Operating System Guide*.

8.2 Adding Kernel Resource Settings to Your INTERACTIVE UNIX System Kernel Systems File

To modify resource settings on your INTERACTIVE UNIX System, refer to the "INTERACTIVE UNIX Operating System Maintenance Procedures" in the *INTERACTIVE UNIX Operating System Guide*.

Your kernel system file has the following kernel resource settings (if the values were less prior to this installation, they were increased; otherwise they remained the same):

```
SHLBMAX=8
NOFILES=64
MAXUP=60
NSTREAM=128
NQUEUE=512
NBLK4096=4
NBLK2048=32
NBLK1024=32
NBLK512=32
NBLK256=64
NBLK128=128
NBLK64=256
NBLK16=256
NBLK4=128
```

8.3 MSC Technologies Bus Mouse

The MSC Technologies, Inc. (formerly Mouse Systems Corporation) Bus Mouse card can usually be set to the COM1 or COM2 interrupt lines. If you have two COM ports on the base system, you must disable one of them to use this mouse. Refer to the hardware manual for your system.

8.4 Troubleshooting Potential Kernel Resource Problems

When you execute a client that used to work and it does not start up, X probably caused the kernel to run out of a resource. If this happens, you must increase the kernel resource whose limit X has reached. To determine which kernel resource should be increased, this checklist is suggested:

- Check the STREAMS resource usage:
  - If you have INTERACTIVE TCP/IP installed, you can use the command:

    ```
    netstat -m
    ```

  - Otherwise, use the `strstat` command of the *crash* utility (see *crash*(1M) in the *INTERACTIVE UNIX System User's/System Administrator's Reference Manual*). The `strstat` command gives information on STREAMS resource usage.

  You should increase the STREAMS resource that X is failing on.

- Check the number of pseudo terminals. The number of pseudo terminals that are configured when X is installed is 16. You must increase the number of pseudo terminals if you reach this limit. Refer to *pty*(7) for additional information.
- The ratio of STREAMS pipes to number of clients is 2-to-1. You must have two STREAMS pipes available for every X client you are running, that is, setting NUMSP to 50 will allow you to run 25 clients.
- Check to make sure that you have not encountered limits with NOFILES (number of files per process), NPROC (total number of processes), MAXUP (number of processes per user), or NUMSP (number of streams pipes).

The default settings when X is installed should be adequate for most users.

INDEX

Introduction

X11 manual entries that are referenced in the text or SEE ALSO sections but not included in this manual can be found in the O'Reilly & Associates *X Window System User's Guide, Motif Edition*.

The following entries included in the O'Reilly & Associates *X Window System User's Guide* are not applicable to INTERACTIVE X11:

xcol(1)
xmh(1)

INTERACTIVE X11
Reference Manual

CONTENTS

NAME

Xcpqag – COMPAQ AG1024 Smart Board Interface server for X11

SYNOPSIS

Xcpqag [option] ...

DESCRIPTION

Xcpqag is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System, Version 2.2 or later, using the COMPAQ Advanced Graphics 1024 board display adapter. The *Xcpqag* server is able to provide an effective display area that is larger than the one provided by the standard *Xcpqag* display adapter. See "panning" below for more details.

If you want to change the I/O address of the COMPAQ AG1024 board, then you must reconfigure the *cpqag* kernel driver. To do this, see *cpqag*(7). This should only be necessary if you want to have the I/O base address be at 0280h instead of the default 0290h.

Xcpqag operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

VT flipping is supported on this board. (For additional information on virtual terminals and VT flipping, refer to the section "USING VIRTUAL TERMINALS" in the "INTERACTIVE UNIX Operating System Primer" in the *INTERACTIVE UNIX System Guide for New Users*.) Note that only one *Xcpqag* server can be run at a time.

OPTIONS

–config *configuration file*
Specifies an alternate configuration file to be used instead of **/usr/lib/X11/Xconfig**.

Other options are described under *Xserver*(1).

CONFIGURATION

The *Info* field in the configuration file (see *Xconfig*(5)) is used to specify options to *Xcpqag*. This field should be a quoted string consisting of 0 or more keyword parameters, for example, **"config_mode=*0* pan=*y*"**.

The configuration mode specifies the board configuration (resolution, pixel size, monitor, etc.) that will be used. The following table describes the requirements for each of the monitor types:

Monitors Supported			
Monitor Type	*Horizontal Frequency*	*Vertical Frequency*	*Horinzontal/Vertical Sync Polarity*
High	54.0 kHz	66.0 Hz	Positive
Alt	48.0 kHz	60.0 Hz	Negative
Low	31.5 kHz	60.0 Hz	Negative

The monitor to be used with the COMPAQ AG1024 may limit the available configuration modes. The presence of the Optional Expansion Memory on the COMPAQ AG1024 card allows for additional screen area to pan. If the Optional Expansion Memory is not present,

then the entire 1024x768 resolution with 256 colors in modes 3 and 5 cannot be displayed due to insufficient memory. In this case, 1024x512 resolution will be displayed.

The following table describes the configuration modes available:

Configuration Modes Supported									
						Without Memory Expansion		*With Memory Expansion*	
Configuration Mode	*Monitor*	*Screen Width*	*Screen Height*	*Screen Depth*	*Colors*	*Panning Width*	*Panning Height*	*Panning Width*	*Panning Height*
0	Low	640	480	4	16	1024	1024	2048	1024
1	Low	640	480	8	256	1024	512	1024	1024
2	High	1024	768	4	16	1024	1024	2048	1024
3	High	1024	768*	8	256	1024	512	1024	1024
4	Alt	1024	768	4	16	1024	1024	2048	1024
5	Alt	1024	768*	8	256	1024	512	1024	1024

* The height will be 512 if the Memory Expansion is not present.

Xcpqag(1)

The *Xcpqag* server supports panning of the visible screen area across a larger effective screen. Panning the visible screen means using the screen as a window onto a larger display area. When the mouse travels to an edge of the screen, the entire screen will appear to slide over the full display area. This allows users without "extended" display adapters to have a larger work area than would otherwise be available. Panning can be enabled by adding the keyword parameter **pan=y** or disabled by adding the keyword parameter **pan=n** into the *Info* field of the configuration file.

The display width and height can be made smaller by specifying the keyword parameters **display_width=***nnn* and **display_height=***nnn*. This may be useful in limiting the panned area so that the screen does not become too large to be useful when using a window manager such as Motif, which utilizes pop-up boxes. It is not possible to increase either the width or the height of the display.

The screen width and height can be specified with the keywords **screen_width=***nnn* and **screen_height=***nnn*. This specifies the dimensions of the screen in inches. This value must be an integer.

For example, the following configuration file entry fully specifies display 0 to be a standard *Xcpqag* console display 640 pixels wide by 480 pixels high with 16 colors, with panning enabled, running on a screen 12 inches wide and 9 inches high:

```
display   CPQAG   "config_mode=0 pan=y screen_width=12 screen_height=9" 0  /dev/console
keyboard  AT      101                                                  0  /dev/console:/dev/vt%02d
mouse     LOGI-S  "1200 3"                                             0  /dev/tty00
```

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5), cpqag(7).

"USING VIRTUAL TERMINALS" in the "INTERACTIVE UNIX Operating System Primer" in the *INTERACTIVE UNIX System Guide for New Users*.

NAME

Xcvc – Cornerstone Smart Board Interface server for X11

SYNOPSIS

Xcvc [option] ...

DESCRIPTION

Xcvc is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System Version 2.2 or later, using any display adapter based on the Cornerstone Technology CVC 1 or CVC 2 display controllers.

Xcvc operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

OPTIONS

Options are described under *Xserver*(1).

CONFIGURATION

Xcvc requires the *Type* field in the configuration file (see *Xconfig*(5)) to be CVC.

The *Info* field must contain at least the screen dimensions. It can also contain the number of colors (gray levels), the screen resolution, and the starting physical address for video memory, although *Xcvc* has reasonable defaults for each of these. Entries in the *Info* field are blank-space separated and can appear in any order. The entire field must be enclosed in double quotes unless there is only one entry in the field. Possible entries are as follows:

Screen size in inches in the format *xinches***x***yinches*.

Number of colors (gray levels). The default is **2** (monochrome), which is all the current version of the server supports.

Video memory physical location in the format *loc***m** where *loc* is an even number between 2 and 14. This is the starting address in physical memory to which the video memory's 2-MB area is mapped. The default is **12m**, meaning a start address of 12 megabytes. This should not be changed unless the location interferes with other hardware in the machine. Check carefully before using an address of **14m** because some machines may have other uses for the extreme high end of the 16 MB memory space. (The CVC boards decode an entire 2 MB of address space.)

Screen resolution in the format *xpixels***x***ypixels*. This should only be specified for new boards. Normally, the server determines the resolution from the identification returned by the board itself; an explicit resolution value is required only for a board type unknown to the server. If the board type *is* known to the server and the resolution is specified in the *Info* field, the specified resolution must match what the server expects; no override is possible.

Xcvc supports a variety of display systems based on the Cornerstone video controller hardware. Supported display systems include at least the following:

Cornerstone DualPage, SinglePage XL
ADI AD-1700, AD-1901
Compugraphic Dawn WS

CPT Full-Page WS
DesignView 19/E
Hyundai HGC-1280
Panasonic M19/15 AT
Princeton IMAGER
Samsung OnePage
SYSDYNE! PC 19PW
Tatung GC-1580
Taxan Crystal View

For Version 2.0.1 or later, *Xcvc* supports the Cornerstone Video Controller PC202A-19 using a DualPage D20A/0191 monitor.

The following configuration file entry fully specifies display 0 to be a standard Cornerstone DualPage display running on a screen 14.5 inches wide and 11 inches high:

```
display     CVC      "14.5x11"   0.0   /dev/console
keyboard    AT       101         0.0   /dev/console:/dev/vt%02d
mouse       LOGI-S   "1200 3"    0.0   /dev/tty00
```

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5).

NAME

Xdcaga – Desktop Computing AGA 1024 Smart Board Interface server for X11

SYNOPSIS

Xdcaga [option] ...

DESCRIPTION

Xdcaga is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System, Version 2.2 or later, using the Desktop Computing Advanced Graphics Accelerator AGA 1024 display adapter. The *Xdcaga* server is able to provide an effective display area that is larger than the one provided by the standard *Xdcaga* display adapter. See "panning" below for more details.

Xdcaga operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

VT flipping is not supported on this board.

OPTIONS

–config *configuration file*
Specifies an alternate configuration file to be used instead of **/usr/lib/X11/Xconfig**.

Other options are described under *Xserver*(1).

CONFIGURATION

The *Info* field in the configuration file (see *Xconfig*(5)) is used to specify options to *Xdcaga*. This field should be a quoted string consisting of 0 or more keyword parameters, for example, **"config_mode=*0* pan=*y*"**.

The configuration mode specifies the board configuration (resolution, pixel size, monitor, etc.) that will be used. The following table describes the requirements for each of the monitor types:

Monitors Supported		
Monitor Type	*Horizontal Frequency*	*Vertical Frequency*
Low	31.5 kHz	60.0 Hz
Medium	37.0 kHz	60.0 Hz
High Interlaced	35.5 kHz	43.5 Hz
High	48.0 kHz	60.0 Hz

The following table describes the configuration modes available:

Configuration Modes Supported							
Configuration Mode	*Monitor*	*Screen Width*	*Screen Height*	*Screen Depth*	*Colors*	*Panning Width*	*Panning Height*
0	Low	640	480	8	256	1024	1024
1	High	1024	768	8	256	1024	1024
2	Alternate High	1024	768	8	256	1024	1024
3	High Interlaced	1024	768	8	256	1024	1024
4	Meduim	800	600	8	256	1024	1024

The *Xdcaga* server supports panning of the visible screen area across a larger effective screen. Panning the visible screen means using the screen as a window onto a larger display area. When the mouse travels to an edge of the screen, the entire screen will appear to slide over the full display area. This allows users without "extended" display adapters to have a larger work area than would otherwise be available. Panning can be enabled by adding the keyword parameter **pan=y** or disabled by adding the keyword parameter **pan=n** into the *Info* field of the configuration file.

The display width and height can be made smaller by specifying the keyword parameters **display_width=***nnn* and **display_height=***nnn*. This may be useful in limiting the panned area so that the screen does not become too large to be useful when using a window manager such as Motif, which utilizes pop-up boxes. It is not possible to increase either the width or the height of the display.

The screen width and height can be specified with the keywords **screen_width=***nnn* and **screen_height=***nnn*. This specifies the dimensions of the screen in inches. This value must be an integer.

For example, the following configuration file entry fully specifies display 0 to be a standard *Xdcaga* console display 640 pixels wide by 480 pixels high with 16 colors, with panning enabled, running on a screen 12 inches wide and 9 inches high:

```
display   DCAGA   "config_mode=0 pan=y screen_width=12 screen_height=9"   0   /dev/console
keyboard  AT      101                                                     0   /dev/console:/dev/vt%02d
mouse     LOGI-S  "1200 3"                                                0   /dev/tty00
```

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5), dcaga(7).

NAME

Xgp – 8514/A Graphics Processor server for X11

SYNOPSIS

Xgp [option] ...

DESCRIPTION

Xgp is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System Version 2.2 or later, using an 8514/A (or compatible) display adapter. *Xgp* operates on a 386- or 486-based ISA, EISA, or Micro Channel computer. It will normally be started by *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

Xgp supports a *PseudoColor* visual type when used with a color monitor and *GrayScale* visual type when used with a monochrome monitor.

OPTIONS

Options are described in *Xserver*(1).

CONFIGURATION

Xgp requires the *Type* field in the configuration file (*Xconfig*(5)) to be 8514.

The *Info* field is used to specify the board, refresh rate (in Hz), resolution, number of colors to be used, and the width and height of the attached monitor. The format of this field is:

"BOARD REFRESH XPIXELSxYPIXELS COLORS WIDTHxHEIGHT"

Standard 8514/A compatible display adapters will support XPIXELSxYPIXELS values of either 1024x768 at 43 Hz or 640x480 at 60 Hz. Some extended 8514/A compatible displays will support an additional resolution of 1280x1024 and refresh rates of 60 or 70 Hz. See the table below for details. The WIDTHxHEIGHT argument is specified in whole inches. The *colors* argument may be either 16 or 256.

In addition to the supported boards listed in the following table, the *Xvga* server also supports VESA SVPMI description files. These files are found in the directory **/usr/lib/X11/vesa/vga**. To see a list of these boards, type:

sysadm configmgmt

When you are asked to select a VGA board manufacturer, select:

SVPMI description file

See the *INTERACTIVE X11 Installation Instructions and Maintenance Procedures* for more information about running *sysadm configmgmt*.

The following table shows the boards and resolutions supported. All modes are non-interlaced unless otherwise noted:

8514 Boards Supported					
			Resolution		
Name	*Board*	*Refresh Rate*	*640x480*	*1024x768*	*1280x1024*
Adex					
8514/AT	ADEX	60, 70	*	*	
8514/MC	ADEX	60, 70	*	*	
1280/AT	ADEX	60, 70	*	*	(2)
IBM					
8514	IBM	43	*	(1)	
Matrox					
MWIN-1280	MATROX	60, 70	*	*	(2)
Western Digital					
8514	IBM	60, 70	*	*	

Legend:

1. Interlaced display.
2. 60 Hz only.

For example, the following configuration file entry fully specifies display 0 to be a standard 8514/A display 1024 pixels wide by 768 pixels high with 256 colors on an IBM 8514 monitor:

```
display     8514    "IBM 43 1024x768 256 12x9"    0.0    /dev/console
keyboard    AT      101                           0.0    /dev/console:/dev/vt0%d
mouse       LOGI-S  "1200 3"                      0.0    /dev/tty00
```

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5).

NAME

Xhrc – Hercules Monochrome Smart Board Interface server for X11

SYNOPSIS

Xhrc [option] ...

DESCRIPTION

Xhrc is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System Version 2.2 or later, using a Hercules monochrome display adapter. *Xhrc* operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

OPTIONS

Options are described under *Xserver*(1).

CONFIGURATION

Xhrc requires the *Type* field in the configuration file (*Xconfig*(5)) to be HRC.

The *Info* field is used to specify the resolution and number of colors to be used, and the width and height of of the attached monitor. The format of this field is "XPIXELSxYPIXELS *colors* WIDTHxHEIGHT The WIDTHxHEIGHT argument is specified in whole inches. The *colors* argument is 2.

For example, the following configuration file entry fully specifies display 0 to be a standard HRC monochrome display 720 pixels wide by 348 pixels high on an Amdek 310a monitor:

```
display     HRC     "720x348 2 9x7" 0.0     /dev/console
mouse       LOGI-S  "1200 3"               0.0
keyboard    AT      "101"                   0.0
```

Video Network Adapter (VNA) workstations are specified in the configuration file by indicating the VNA device name in place of **/dev/console**, i.e., **/dev/vna00** for the first VNA display. This example defines VNA unit 2 as display 2 with a LOGITECH serial mouse attached to the serial port:

```
display     HRC     "720x348 2 9x7" 2.0     /dev/vna20
mouse       LOGI-S  "1200 3"               2.0
keyboard    AT      "84"                    2.0
```

Following this example, create a variable DISPLAY in your environment, equal to *unix:2*, where 2 is the VNA unit number.

FILES

/usr/lib/X11/Xconfig – server device configuration file

SEE ALSO

X(1), Xserver(1), xinit(1), Xconfig(5).

NAME

Xigsp – IMAgraph TI1210 Smart Board Interface server for X11

SYNOPSIS

Xigsp [option] ...

DESCRIPTION

Xigsp is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System, Version 2.2 or later, using the IMAgraph TI1210 display adapter. The *Xigsp* server is able to provide an effective display area that is larger than the one provided by the standard *Xigsp* display adapter. See "panning" below for more details.

If you want to change the address jumper settings of the IMAgraph TI1210 board, then you must reconfigure the *igsp* kernel driver. To do this, see *igsp*(7).

Xigsp operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

VT flipping is not supported on this board.

OPTIONS

–config *configuration file*
Specifies an alternate configuration file to be used instead of **/usr/lib/X11/Xconfig**.

Other options are described under *Xserver*(1).

CONFIGURATION

The *Info* field in the configuration file (see *Xconfig*(5)) is used to specify options to *Xigsp*. This field should be a quoted string consisting of 0 or more keyword parameters, for example, **"config_mode=*0* pan=*y*"**.

The configuration mode specifies the board configuration (resolution, pixel size, etc.) that will be used. For the IMAgraph TI1210 board, only configuration mode 0 is used. The configuration is determined from the board.

The *Xigsp* server supports panning of the visible screen area across a larger effective screen. Panning the visible screen means using the screen as a window onto a larger display area. When the mouse travels to an edge of the screen, the entire screen will appear to slide over the full display area. This allows users without "extended" display adapters to have a larger work area than would otherwise be available. Panning can be enabled by adding the keyword parameter **pan=y** or disabled by adding the keyword parameter **pan=n** into the *Info* field of the configuration file.

The display width and height can be made smaller by specifying the keyword parameters **display_width=***nnn* and **display_height=***nnn*. This may be useful in limiting the panned area so that the screen does not become too large to be useful when using a window manager such as Motif, which utilizes pop-up boxes. It is not possible to increase either the width or the height of the display.

The screen width and height can be specified with the keywords **screen_width=***nnn* and **screen_height=***nnn*. This specifies the dimensions of the screen in inches. This value must be an integer.

For example, the following configuration file entry fully specifies display 0 to be a standard *Xigsp* console display, with panning enabled, running on a screen 12 inches wide and 9 inches high:

```
display     IGSP      "config_mode=0 pan=y screen_width=12 screen_height=9" 0  /dev/console
keyboard    AT        101                                                   0  /dev/console:/dev/vt%02d
mouse       LOGI-S    "1200 3"                                              0  /dev/tty00
```

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5), igsp(7).

NAME

Xlvp – LaserView PLUS Smart Board Interface server for X11

SYNOPSIS

Xlvp [option] ...

DESCRIPTION

Xlvp is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System Version 2.2 or later, using a Sigma Designs LaserView PLUS display adapter. *Xlvp* operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

OPTIONS

Options are described under *Xserver*(1).

CONFIGURATION

Xlvp requires the *Type* field in the configuration file (*Xconfig*(5)) to be LVP.

The *Info* field is used to specify the resolution and number of colors to be used and the width and height of the attached monitor. The format of this field is "XPIXELSxYPIXELS colors WIDTHxHEIGHT." The WIDTHxHEIGHT argument is specified in whole inches. The *colors* argument is **2**.

For example, the following configuration file entry fully specifies display 0 to be a standard LVP monochrome display 1664 pixels wide by 1200 pixels high on a Sigma Designs 19-inch monitor:

```
display     LVP       "1664x1200 2 14x10"   0.0   /dev/console
keyboard    AT        101                   0.0   /dev/console:/dev/vt0%d
mouse       LOGI-S    "1200 3"              0.0   /dev/tty00
```

Sigma Designs also offers a 15-inch monitor. The "display" line would be:

```
display     LVP       "1664x1200 2 11x8"    0.0   /dev/console
```

FILES

/usr/lib/X11/Xconfig – server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5).

NAME

Xmsfdp – MegaScan FDP-6120 Smart Board Interface server for X11

SYNOPSIS

Xmsfdp [option] ...

DESCRIPTION

Xmsfdp is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System, Version 2.2 or later, using the MegaScan FDP-6120 display adapter. This server is only supported in dual-headed mode. The *Xmsfdp* server is able to provide an effective display area that is larger than the one provided by the standard *Xmsfdp* display adapter. See "panning" below for more details.

If you want to change the input/output address or the window address of the MegaScan FDP-6120 board, then you must reconfigure the *msfdp* kernel driver. To do this, see *msfdp*(7).

Xmsfdp operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

VT flipping is not supported on this board.

OPTIONS

–config *configuration file*
Specifies an alternate configuration file to be used instead of **/usr/lib/X11/Xconfig**.

Other options are described under *Xserver*(1).

CONFIGURATION

The *Info* field in the configuration file (see *Xconfig*(5)) is used to specify options to *Xmsfdp*. This field should be a quoted string consisting of 0 or more keyword parameters, for example, **"config_mode=*0* pan=*y*"**.

The configuration mode specifies the board configuration (resolution, pixel size, etc.) that will be used. For the MegaScan FDP-6120 board, only configuration mode 0 is used.

The following table describes the configuration mode available:

Configuration Mode Supported						
Configuration Mode	*Screen Width*	*Screen Height*	*Screen Depth*	*Colors*	*Panning Width*	*Panning Height*
0	2560	1928	1	2	4096	1952

The *Xmsfdp* server supports panning of the visible screen area across a larger effective screen. Panning the visible screen means using the screen as a window onto a larger display area. When the mouse travels to an edge of the screen, the entire screen will appear to slide over the full display area. This allows users without "extended" display adapters to have a larger work area than would otherwise be available. Panning can be enabled by adding the keyword parameter **pan=y** or disabled by adding the keyword parameter **pan=n** into the *Info* field of the configuration file.

The display width and height can be made smaller by specifying the keyword parameters **display_width=*nnn*** and **display_height=*nnn*.** This

may be useful in limiting the panned area so that the screen does not become too large to be useful when using a window manager such as Motif, which utilizes pop-up boxes. It is not possible to increase either the width or the height of the display.

The screen width and height can be specified with the keywords **screen_width=***nnn* and **screen_height=***nnn*. This specifies the dimensions of the screen in inches. This value must be an integer.

For example, the following configuration file entry fully specifies display 0 to be a standard *Xmsfdp* console display 2560 pixels wide by 1928 pixels high with 2 colors, with panning enabled, running on a screen 12 inches wide and 9 inches high:

```
display   MSFDP   "config_mode=0 pan=y screen_width=12 screen_height=9"   0   /dev/console
keyboard  AT      101                                                    0   /dev/console:/dev/vt%02d
mouse     LOGI-S  "1200 3"                                               0   /dev/tty00
```

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5), msfdp(7).

NAME

Xnnp – Number Nine PEPPER PRO1280 Smart Board Interface server for X11

SYNOPSIS

Xnnp [option] ...

DESCRIPTION

Xnnp is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System, Version 2.2 or later, using the Number Nine PEPPER PRO1280 display adapter. The *Xnnp* server is able to provide an effective display area that is larger than the one provided by the standard *Xnnp* display adapter. See "panning" below for more details.

Xnnp operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

VT flipping is not supported on this board.

OPTIONS

–config *configuration file*
Specifies an alternate configuration file to be used instead of **/usr/lib/X11/Xconfig**.

Other options are described under *Xserver*(1).

CONFIGURATION

The *Info* field in the configuration file (see *Xconfig*(5)) is used to specify options to *Xnnp*. This field should be a quoted string consisting of 0 or more keyword parameters, for example, **"config_mode=*0* pan=*y*"**.

The configuration mode specifies the board configuration (resolution, pixel size, etc.) that will be used.

The following table describes the configuration modes available:

Configuration Modes Supported						
Configuration Mode	*Screen Width*	*Screen Height*	*Screen Depth*	*Colors*	*Panning Width*	*Panning Height*
0	1280	1024	4	16	2048	1024
1	1280	1024	8	256	1280	1024

The *Xnnp* server in configuration mode 0 supports panning of the visible screen area across a larger effective screen. Panning the visible screen means using the screen as a window onto a larger display area. When the mouse travels to an edge of the screen, the entire screen will appear to slide over the full display area. This allows users without "extended" display adapters to have a larger work area than would otherwise be available. Panning can be enabled by adding the keyword parameter **pan=y** or disabled by adding the keyword parameter **pan=n** into the *Info* field of the configuration file.

The display width and height can be made smaller by specifying the keyword parameters **display_width=***nnn* and **display_height=***nnn*. This may be useful in limiting the panned area so that the screen does not become too large to be useful when using a window manager such as

Motif, which utilizes pop-up boxes. It is not possible to increase either the width or the height of the display.

The screen width and height can be specified with the keywords **screen_width=***nnn* and **screen_height=***nnn*. This specifies the dimensions of the screen in inches. This value must be an integer.

For example, the following configuration file entry fully specifies display 0 to be a standard *Xnnp* console display 1280 pixels wide by 1024 pixels high with 16 colors, with panning enabled, running on a screen 12 inches wide and 9 inches high:

```
display   NNP      "config_mode=0 pan=y screen_width=12 screen_height=9"   0   /dev/console
keyboard  AT       101                                                    0   /dev/console:/dev/vt%02d
mouse     LOGI-S   "1200 3"                                               0   /dev/tty00
```

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5), nnp(7).

NAME

Xnnpi – Number Nine PEPPER PRO1024ISA Smart Board Interface server for X11

SYNOPSIS

Xnnpi [option] ...

DESCRIPTION

Xnnpi is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System, Version 2.2 or later, using the Number Nine PEPPER PRO1024ISA display adapter. The *Xnnpi* server is able to provide an effective display area that is larger than the one provided by the standard *Xnnpi* display adapter. See "panning" below for more details.

If you want to change the Host Interface address of the Number Nine PEPPER PRO1024ISA board, then you must reconfigure the *nnpi* kernel driver. To do this, see *nnpi*(7).

Xnnpi operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

VT flipping is supported on this board. (For additional information on virtual terminals and VT flipping, refer to the section "USING VIRTUAL TERMINALS" in the "INTERACTIVE UNIX Operating System Primer" in the *INTERACTIVE UNIX System Guide for New Users*.) Note that only one *Xnnpi* server can be run at a time.

OPTIONS

–config *configuration file*
Specifies an alternate configuration file to be used instead of **/usr/lib/X11/Xconfig**.

Other options are described under *Xserver*(1).

CONFIGURATION

The *Info* field in the configuration file (see *Xconfig*(5)) is used to specify options to *Xnnpi*. This field should be a quoted string consisting of 0 or more keyword parameters, for example, **"config_mode=*0* pan=*y*"**.

The configuration mode specifies the board configuration (resolution, pixel size, monitor, etc.) that will be used. The following table describes the requirements for each of the monitor types:

Monitors Supported		
Monitor Type	*Horizontal Frequency*	*Vertical Frequency*
High	48.7 kHz	60.0 Hz
High Interlaced	35.5 kHz	43.5 Hz
Low	31.1 kHz	59.0 Hz

The following table describes the configuration modes available:

Configuration Modes Supported							
Configuration Mode	*Monitor*	*Screen Width*	*Screen Height*	*Screen Depth*	*Colors*	*Panning Width*	*Panning Height*
0	High	1024	768	8	256	1024	1024
1	High Interlaced	1024	768	8	256	1024	1024
2	High	1024	768	4	16	2048	1024
3	High Interlaced	1024	768	4	16	2048	1024
4	High	1024	768	2	4	2048	2048
5	High Interlaced	1024	768	2	4	2048	2048
6	High	1024	768	1	2	4096	2048
7	High Interlaced	1024	768	1	2	4096	2048
8	Low	640	480	8	256	1024	1024
9	Low	640	480	4	16	2048	1024
10	Low	640	480	2	4	2048	2048
11	Low	640	480	1	2	4096	2048

The *Xnnpi* server supports panning of the visible screen area across a larger effective screen. Panning the visible screen means using the screen as a window onto a larger display area. When the mouse travels to an edge of the screen, the entire screen will appear to slide over the full display area. This allows users without "extended" display adapters to have a larger work area than would otherwise be available. Panning can be enabled by adding the keyword parameter **pan=y** or disabled by adding the keyword parameter **pan=n** into the *Info* field of the configuration file.

The display width and height can be made smaller by specifying the keyword parameters **display_width=***nnn* and **display_height=***nnn*. This may be useful in limiting the panned area so that the screen does not become too large to be useful when using a window manager such as Motif, which utilizes pop-up boxes. It is not possible to increase either the width or the height of the display.

The screen width and height can be specified with the keywords **screen_width=***nnn* and **screen_height=***nnn*. This specifies the dimensions of the screen in inches. This value must be an integer.

For example, the following configuration file entry fully specifies display 0 to be a standard *Xnnpi* console display 1024 pixels wide by 768 pixels high with 256 colors, with panning enabled, running on a screen 12 inches wide and 9 inches high:

```
display     NNPI     "config_mode=0 pan=y screen_width=12 screen_height=9"   0   /dev/console
keyboard    AT       101                                                     0   /dev/console:/dev/vt%02d
mouse       LOGI-S   "1200 3"                                                0   /dev/tty00
```

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5), nnpi(7).

"USING VIRTUAL TERMINALS" in the "INTERACTIVE UNIX Operating System Primer" in the *INTERACTIVE UNIX System Guide for New Users*.

NAME

Xpw – Pixelworks Smart Board Interface server for X11

SYNOPSIS

Xpw [option] ...

DESCRIPTION

Xpw is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System Version 2.2 or later, using the Pixelworks Clipper Graphics Series controller. *Xpw* operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

OPTIONS

–config *configuration file*

Specifies an alternate configuration file to be used, instead of **/usr/lib/X11/Xconfig**.

Other options are described under *Xserver*(1).

CONFIGURATION

Xpw requires the *Type* field in the configuration file (*Xconfig*(5)) to be the string PW-CG. The *Info* field is presently used to specify the width and height of the attached monitor. The format of this field is "WIDTHxHEIGHT" in whole inches. The *Info* field may be left as an empty string. In this case the width is assumed to be 12 inches and the height is assumed to be 9 inches. This means that the *Info* field for a 12-inch x 9-inch screen would be **12x9**. The *Devname* field in the configuration file must presently be set to **/dev/pw0**.

```
display    PW-CG    "12x9"    0.0    /dev/pw0
```

or

```
display    PW-CG    ""        0.0    /dev/pw0
```

An example of a complete configuration entry for the console using a Logitech serial mouse on **/dev/tty00** would be:

```
display     PW-CG     12x9       0.0    /dev/pw0
keyboard    AT        101        0.0    /dev/console:/dev/vt%02d
mouse       LOGI-S    "1200 3"   0.0    /dev/tty00
```

FILES

/usr/lib/X11/Xconfig	server device configuration file
/dev/pw0	pw device entry

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5).

NAME

Xrren2 – Renaissance Rendition II Smart Board Interface server for X11

SYNOPSIS

Xrren2 [option] ...

DESCRIPTION

Xrren2 is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System, Version 2.2 or later, using the Renaissance Rendition II display adapter. The *Xrren2* server is able to provide an effective display area that is larger than the one provided by the standard *Xrren2* display adapter. See "panning" below for more details.

Xrren2 operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

VT flipping is supported on this board. (For additional information on virtual terminals and VT flipping, refer to the section "USING VIRTUAL TERMINALS" in the "INTERACTIVE UNIX Operating System Primer" in the *INTERACTIVE UNIX System Guide for New Users*.) Note that only one *Xrren2* server can be run at a time.

OPTIONS

–config *configuration file*
Specifies an alternate configuration file to be used instead of **/usr/lib/X11/Xconfig**.

Other options are described under *Xserver*(1).

CONFIGURATION

The *Info* field in the configuration file (see *Xconfig*(5)) is used to specify options to *Xrren2*. This field should be a quoted string consisting of 0 or more keyword parameters, for example, **"config_mode=*0* pan=*y*"**.

The configuration mode specifies the board configuration (resolution, pixel size, monitor, etc.) that will be used. The following table describes the requirements for each of the monitor types:

Monitors Supported		
Monitor Type	*Horizontal Frequency*	*Vertical Frequency*
High	54.0 kHz	66.0 Hz
High Interlaced	35.5 kHz	43.5 Hz
Medium	37.0 kHz	60.0 Hz
Alt	48.0 kHz	60.0 Hz
Low	31.5 kHz	60.0 Hz

The monitor to be used with the Renaissance Rendition II may limit the available configuration modes. The presence of the Additional VRAM Memory on the Rendition II board allows for additional screen area to pan. If the Additional VRAM Memory is not present, then the entire 1024x768 resolution with 256 colors in modes 3, 5, 7, and 9 cannot be displayed due to insufficient memory. In this case, the height will be 512 rows.

The following table describes the configuration modes available:

Configuration Modes Supported											
						Without Additional VRAM Memory 512Kb VRAM		*With 256Kb Additional VRAM Memory 768Kb VRAM*		*With 512Kb Additional VRAM Memory 1024Kb VRAM*	
Configuration Mode	*Monitor*	*Screen Width*	*Screen Height*	*Screen Depth*	*Colors*	*Panning Width*	*Panning Height*	*Panning Width*	*Panning Height*	*Panning Width*	*Panning Height*
0	Low	640	480	4	16	1024	1024	1024	1536	2048	1024
1	Low	640	480	8	256	1024	512	1024	768	1024	1024
2	Medium	800	600	4	16	1024	1024	1024	1536	2048	1024
3	Medium	800	600*	8	256	1024	512	1024	768	1024	1024
4	High	1024	768	4	16	1024	1024	1024	1536	2048	1024
5	High	1024	768*	8	256	1024	512	1024	768	1024	1024
6	Alt	1024	768	4	16	1024	1024	1024	1536	2048	1024
7	Alt	1024	768*	8	256	1024	512	1024	768	1024	1024
8	High Interlaced	1024	768	4	16	1024	1024	1024	1536	2048	1024
9	High Interlaced	1024	768*	8	256	1024	512	1024	768	1024	1024

* The height will be 512 if additional VRAM memory is not present.

The *Xrren2* server supports panning of the visible screen area across a larger effective screen. Panning the visible screen means using the screen as a window onto a larger display area. When the mouse travels to an edge of the screen, the entire screen will appear to slide over the full display area. This allows users without "extended" display adapters to have a larger work area than would otherwise be available. Panning can be enabled by adding the keyword parameter **pan=y** or disabled by adding the keyword parameter **pan=n** into the *Info* field of the configuration file.

The display width and height can be made smaller by specifying the keyword parameters **display_width=***nnn* and **display_height=***nnn*. This may be useful in limiting the panned area so that the screen does not become too large to be useful when using a window manager such as Motif, which utilizes pop-up boxes. It is not possible to increase either the width or the height of the display.

The screen width and height can be specified with the keywords **screen_width=***nnn* and **screen_height=***nnn*. This specifies the dimensions of the screen in inches. This value must be an integer.

For example, the following configuration file entry fully specifies display 0 to be a standard *Xrren2* console display 640 pixels wide by 480 pixels high with 16 colors, with panning enabled, running on a screen 12 inches wide and 9 inches high:

```
display     RREN2    "config_mode=0 pan=y screen_width=12 screen_height=9" 0  /dev/console
keyboard    AT       101                                                   0  /dev/console:/dev/vt%02d
mouse       LOGI-S   "1200 3"                                              0  /dev/tty00
```

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5), rren2(7).

"USING VIRTUAL TERMINALS" in the "INTERACTIVE UNIX Operating System Primer" in the *INTERACTIVE UNIX System Guide for New Users*.

NAME

X – X Window System server

SYNOPSIS

X [:displaynumber] [–option ...] [ttyname]

DESCRIPTION

X is the generic name for the X Window System server. It is frequently a link or a copy of the appropriate server binary for driving the most frequently used server on a given machine. In INTERACTIVE X11 Releases 1.1 and later, *X* is a link to the program *xfront*(1), which automatically executes the appropriate server. The sample server from MIT supports the following platforms:

Xqvss	Digital monochrome vaxstationII or II
Xqdss	Digital color vaxstationII or II
Xsun	Sun monochrome or color Sun 2, 3, or 4
Xhp	HP Topcat 9000s300
Xapollo	Apollo monochrome (Domain/IX 9.6)
Xibm	IBM APA and megapel PC/RT
XmacII	Apple monochrome Macintosh II
Xplx	Parallax color and video graphics controller

The INTERACTIVE X11 release supports:

Xcpqag	COMPAQ AG1024 smart board interface server
Xcvc	Cornerstone smart board interface server
Xdcaga	Desktop Computing AGA1024 smart board interface server
Xgp	8514/A graphics processor smart board interface server
Xhrc	Hercules monochrome smart board interface server
Xigsp	IMAgraph TI1210 smart board interface server
Xlvp	LaserView PLUS smart board interface server
Xmsfdp	MegaScan FDP-6120 smart board interface server
Xnnp	Number Nine PEPPER PRO1280 smart board interface server
Xnnpi	Number Nine PEPPER PRO1024ISA smart board interface server
Xpw	Pixelworks smart board interface server
Xrren2	Renaissance Rendition II smart board interface server
Xsp200	Spectre SP200 smart board interface server
Xtisdb	Texas Instruments TMS34010 smart board interface server
Xv256	256 color VGA smart board interface server
Xvga	EGA/VGA color smart board interface server
Xviking	Moniterm 21/91 Viking smart board interface server
Xwge	Bell Technologies Workstation Graphics Engine (Blit) server

STARTING THE SERVER

The server is usually started from the X Display Manager program *xdm*. This utility is run from the system boot files and takes care of keeping the server running, prompting for user names and passwords, and starting up the user sessions. It is easily configured for sites that wish to provide nice, consistent interfaces for novice users (loading convenient sets of resources, starting up a window manager, clock, and nice selection of terminal emulator windows).

Since *xdm* now handles automatic starting of the server in a portable way, the −L option to *xterm* is now considered obsolete. Support for starting a login window from 4.3BSD-derived **/etc/ttys** files may not be included in future releases.

Installations that run more than one window system still need to use the *xinit* utility. However, *xinit* is to be considered a tool for building startup scripts and is not intended for use by end users. Site adminstrators are *strongly* urged to build nicer interfaces for novice users.

When the sample server starts up, it takes over the display. If you are running on a workstation whose console is the display, you cannot log in to the console while the server is running.

NETWORK CONNECTIONS

The sample server supports connections made using the following reliable byte-streams:

TCP/IP
: The server listens on port htons(6000+*n*), where *n* is the display number.

UNIX System Domain
: The sample server uses **/tmp/.X11-unix/Xn** as the file name for a STREAMS pipe node, where *n* is the display number.

DECnet
: The server responds to connections to object *X$Xn*, where *n* is the display number.

OPTIONS

All of the sample servers accept the following command line options:

−a *number*
: Sets pointer acceleration (i.e., the ratio of how much is reported to how much the user actually moved the pointer).

−bs
: Disables backing store support on all screens.

−c
: Turns off key-click.

−config *filename*
: Specifies configuration file to use instead of **/usr/lib/X11/Xconfig**.

c *volume*
: Sets key-click volume (allowable range: 0-8).

−f *volume*
: Sets beep (bell) volume (allowable range: 0-7).

−logo
: Turns on the X Window System logo display in the screen-saver. There is currently no way to change this from a client.

nologo
: Turns off the X Window System logo display in the screen-saver. There is currently no way to change this from a client.

−p *minutes*
: Sets screen-saver pattern cycle time in minutes.

−r
: Turns off auto-repeat.

r
: Turns on auto-repeat.

-s *minutes*
Sets screen-saver timeout time in minutes.

-su Disables save under support on all screens.

-t *numbers*
Sets pointer acceleration threshold in pixels (i.e., after how many pixels pointer acceleration should take effect).

-to *seconds*
Sets default screensaver timeout in seconds.

v Sets video-on screen-saver preference.

-v Sets video-off screen-saver preference.

-co *filename*
Sets name of *rgb* color database.

-help Prints a usage message.

-fp *fontPath*
Sets the search path for fonts.

-fc *cursorFont*
Sets the default cursor font.

-fn *font* Sets the default font.

-wm Forces the default backing-store of all windows to be When-Mapped; an easy way of getting backing-store to apply to all windows.

Many servers also have device-specific command line options. See the manual pages for the individual servers for more details.

SECURITY

The sample server uses an access control list for deciding whether or not to accept connections from clients on a particular machine. This list initially consists of the host on which the server is running as well as any machines listed in the file **/etc/Xn.*hosts***, where **n** is the display number of the server. Each line of the file should contain either an Internet hostname (e.g., expo.lcs.mit.edu) or a DECnet hostname in double colon format (e.g., hydra::). There should be no leading or trailing spaces on any lines. For example:

```
joesworkstation
corporate.company.com
star::
bigcpu::
```

Users can add or remove hosts from this list and enable or disable access control using the *xhost* command from the same machine as the server. For example:

```
% xhost +janesworkstation
janesworkstation being added to access control list
% xhost -star::
public:: being removed from access control list
% xhost +
all hosts being allowed (access control disabled)
% xhost -
all hosts being restricted (access control enabled)
% xhost
access control enabled (only the following hosts are allowed)
joesworkstation
janesworkstation
corporate.company.com
bigcpu::
```

Unlike some window systems, X does not have any notion of window operation permissions or place any restrictions on what a client can do; if a program can connect to a display, it has full run of the screen. The core protocol does have support for providing authentication information when establishing connections, but is not used in the sample implementation. Sites that have authentication and authorization systems (such as Kerberos) might wish to make use of the hooks in the libraries and the server to provide additional security.

SIGNALS

The sample server attaches special meaning to the following signals:

SIGHUP

This signal causes the server to close all existing connections, free all resources, and restore all defaults. It is sent by the display manager whenever the principal user's main application (usually an *xterm* or window manager) exits to force the server to clean up and prepare for the next user.

SIGTERM

This signal causes the server to exit cleanly.

FONTS

Fonts are usually stored as individual files in directories. The list of directories in which the server looks when trying to open a font is controlled by the *font path*. Although most sites will choose to have the server start up with the appropriate font path (using the **-fp** option mentioned above), it can be overridden using the *xset* program.

The default font path for the sample server contains three directories:

/usr/lib/X11/fonts/misc

This directory contains several miscellaneous fonts that are useful on all systems. It contains a very small family of fixed-width fonts (**6x10**, **6x12**, **6x13**, **8x13**, **8x13bold**, and **9x15**) and the cursor font. It also has font name aliases for the commonly used fonts **fixed** and **variable**.

/usr/lib/X11/fonts/75dpi

This directory contains fonts contributed by Adobe Systems, Inc. and Digital Equipment Corporation and by Bitstream, Inc. for 75 dots per inch displays. An integrated selection of sizes, styles, and weights are provided for each family.

/usr/lib/X11/fonts/100dpi
This directory contains versions of some of the fonts in the *75dpi* directory for 100 dot-per-inch displays.

Font databases are created by running the *mkfontdir* program in the directory containing the compiled versions of the fonts (the **.snf** files). Whenever fonts are added to a directory, *mkfontdir* should be rerun so that the server can find the new fonts. *If mkfontdir is not run, the server will not be able to find any fonts in the directory.*

DIAGNOSTICS

Too numerous to list them all. If run from *init*(1M), errors are logged in the file **/usr/adm/X*msgs**.

FILES

/etc/X*.hosts	initial access control list
/usr/lib/X11/fonts/misc /usr/lib/X11/fonts/75dpi /usr/lib/X11/fonts/100dpi	font directories
/usr/lib/X11/rgb.txt	color database
/tmp/.X11-unix/X*	UNIX System domain socket
/usr/adm/X*msgs	error log file

SEE ALSO

X(1), Xcpqag(1), Xcvc(1), Xdcaga(1), Xgp(1), Xhrc(1), Xigsp(1), Xlvp(1), Xmsfdp(1), Xnnp(1), Xnnpi(1), Xpw(1), Xrren2(1) Xsp200(1), Xtisdb(1), Xv256(1), Xvga(1), Xviking(1), Xwge(1), mkfontdir(1), uwm(1), xdm(1), xfront(1), xhost(1), xinit(1), xset(1), xsetroot(1), xterm(1).
init(1M) in the *INTERACTIVE UNIX System User's/System Administrator's Reference Manual.*
inittab(4) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual.*
X Window System Protocol.
Definition of the Porting Layer for the X v11 Sample Server.
Strategies for Porting the X v11 Sample Server.
Godzilla's Guide to Porting the X V11 Sample Server.

BUGS

The option syntax is inconsistent with itself and *xset*(1).

The acceleration option should take a numerator and a denominator like the protocol.

If *X* dies before its clients, new clients won't be able to connect until all existing connections have their TCP TIME_WAIT timers expire.

The color database is missing a large number of colors. However, there doesn't seem to be a better one available that can generate RGB values tailorable to particular displays.

The *xterm* **−L** method for starting an initial window from **/etc/ttys** is completely inadequate and should be removed. Users should use *xdm* instead.

COPYRIGHT

AUTHORS

The sample server was originally written by Susan Angebranndt, Raymond Drewry, Philip Karlton, and Todd Newman, with support from a cast of thousands. Also see the file **/doc/contributors** in the sample distribution for a more complete list.

NAME

Xsp200 – Spectre SP200 Smart Board Interface server for X11

SYNOPSIS

Xsp200 [option] ...

DESCRIPTION

Xsp200 is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System, Version 2.2 or later, using the Spectre SP200 display adapter.

If you want to change the I/O base address of the Spectre SP200 board, then you must reconfigure the *sp200* kernel driver. To do this, see *sp200*(7).

Xsp200 operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

VT flipping is supported on this board. (For additional information on virtual terminals and VT flipping, refer to the section "USING VIRTUAL TERMINALS" in the "INTERACTIVE UNIX Operating System Primer" in the *INTERACTIVE UNIX System Guide for New Users*.) Note that only one *Xsp200* server can be run at a time.

OPTIONS

–config *configuration file*
Specifies an alternate configuration file to be used instead of **/usr/lib/X11/Xconfig**.

Other options are described under *Xserver*(1).

CONFIGURATION

The *Info* field in the configuration file (see *Xconfig*(5)) is used to specify options to *Xsp200*. This field should be a quoted string consisting of 0 or more keyword parameters, for example, "**config_mode=***0* ".

The configuration mode specifies the board configuration (resolution, pixel size, etc.) that will be used. For the Spectre SP200 board, only configuration mode 0 is used.

The following table describes the configuration mode available:

Configuration Mode Supported						
Configuration Mode	*Screen Width*	*Screen Height*	*Screen Depth*	*Colors*	*Panning Width*	*Panning Height*
0	1280	1024	8	256	1280	1024

The screen width and height can be specified with the keywords **screen_width=***nnn* and **screen_height=***nnn*. This specifies the dimensions of the screen in inches. This value must be an integer.

For example, the following configuration file entry fully specifies display 0 to be a standard *Xsp200* console display 1280 pixels wide by 1024 pixels high with 256 colors, running on a screen 12 inches wide and 9 inches high:

display	SP200	"config_mode=0 screen_width=12 screen_height=9"	0	/dev/console
keyboard	AT	101	0	/dev/console:/dev/vt%02d
mouse	LOGI-S	"1200 3"	0	/dev/tty00

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5), sp200(7).

"USING VIRTUAL TERMINALS" in the "INTERACTIVE UNIX Operating System Primer" in the *INTERACTIVE UNIX System Guide for New Users*.

NAME

Xtisdb – Texas Instruments TMS34010 Software Development Board Smart Board Interface server for X11

SYNOPSIS

Xtisdb [option] ...

DESCRIPTION

Xtisdb is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System, Version 2.2 or later, using the Texas Instruments TMS34010 Software Development Board display adapter. The *Xtisdb* server is able to provide an effective display area that is larger than the one provided by the standard *Xtisdb* display adapter. See "panning" below for more details.

If you want to change the memory map address of the Texas Instruments TMS34010 Software Development Board, then you must reconfigure the *tisdb* kernel driver. To do this, see *tisdb*(7). This should only be necessary if you want to install a different address PAL.

Xtisdb operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

VT flipping is supported on this board. (For additional information on virtual terminals and VT flipping, refer to the section "USING VIRTUAL TERMINALS" in the "INTERACTIVE UNIX Operating System Primer" in the *INTERACTIVE UNIX System Guide for New Users*.) Note that only one *Xtisdb* server can be run at a time.

OPTIONS

–config *configuration file*
Specifies an alternate configuration file to be used instead of **/usr/lib/X11/Xconfig**.

Other options are described under *Xserver*(1).

CONFIGURATION

The *Info* field in the configuration file (see *Xconfig*(5)) is used to specify options to *Xtisdb*. This field should be a quoted string consisting of 0 or more keyword parameters, for example, "**config_mode=***1* **pan=***y*".

The configuration mode specifies the board configuration (resolution, pixel size, etc.) that will be used.

The following table describes the configuration modes available:

Configuration Modes Supported						
Configuration Mode	*Screen Width*	*Screen Height*	*Screen Depth*	*Colors*	*Panning Width*	*Panning Height*
0	640	480	4	16	640	480
1	448	480	4	16	448	960

The *Xtisdb* server in configuration mode 1 supports panning of the visible screen area across a larger effective screen. Panning the visible screen means using the screen as a window onto a larger display area. When the mouse travels to an edge of the screen, the entire screen will appear to slide over the full display area. This allows users without "extended" display adapters to have a larger work area than would

otherwise be available. Panning can be enabled by adding the keyword parameter **pan=y** or disabled by adding the keyword parameter **pan=n** into the *Info* field of the configuration file.

The display width and height can be made smaller by specifying the keyword parameters **display_width=***nnn* and **display_height=***nnn*. This may be useful in limiting the panned area so that the screen does not become too large to be useful when using a window manager such as Motif, which utilizes pop-up boxes. It is not possible to increase either the width or the height of the display.

The screen width and height can be specified with the keywords **screen_width=***nnn* and **screen_height=***nnn*. This specifies the dimensions of the screen in inches. This value must be an integer.

For example, the following configuration file entry fully specifies display 0 to be a standard *Xtisdb* console display 448 pixels wide by 480 pixels high with 16 colors, with panning enabled, running on a screen 12 inches wide and 9 inches high:

```
display   TISDB   "config_mode=1 pan=y screen_width=12 screen_height=9"   0   /dev/console
keyboard  AT      101                                                     0   /dev/console:/dev/vt%02d
mouse     LOGI-S  "1200 3"                                                0   /dev/tty00
```

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5), tisdb(7).

"USING VIRTUAL TERMINALS" in the "INTERACTIVE UNIX Operating System Primer" in the *INTERACTIVE UNIX System Guide for New Users*.

NAME

Xv256 – 256-color VGA server for X11

SYNOPSIS

Xv256 [option] ...

DESCRIPTION

Xv256 is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System Version 2.2 or later, using a VGA display adapter displaying 256 colors. Several "enhanced" versions of the VGA adapter standard are supported. The *Xv256* server also supports SunRiver Corporation workstations.

Xv256 operates on a 386- or 486-based ISA, EISA, or Micro Channel computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

Xv256 supports a *PseudoColor* visual type when used with a color monitor and a *GrayScale* visual type when used with a monochrome monitor.

OPTIONS

Options are described under *Xserver*(1).

CONFIGURATION

Xv256 requires the *Type* field in the configuration file (see *Xconfig*(5)) to be V256.

The *Info* field is used to specify the board, resolution and the width and height of the attached monitor. The format of this field is:

"BOARD XPIXELSxYPIXELS WIDTHxHEIGHT"

The WIDTHxHEIGHT argument describes the size of the display and is specified in whole inches.

In addition to the supported boards listed in the following table, the *Xvga* server also supports VESA SVPMI description files. These files are found in the directory **/usr/lib/X11/vesa/vga**. To see a list of these boards, type:

sysadm configmgmt

When you are asked to select a VGA board manufacturer, select:

SVPMI description file

See the *INTERACTIVE X11 Installation Instructions and Maintenance Procedures* for more information about running *sysadm configmgmt*.

The following table shows the VGA boards and resolutions supported. All modes are in non-interlaced mode unless otherwise noted:

VGA Boards Supported							
		Resolution					
Name	*Board*	*320x200*	*640x400*	*640x480*	*720x540*	*800x600*	*1024x768*
ATI							
VGA Wonder	ATI256			*		*	
VGA Wonder V4	ATI2564			*		*	
VGA Wonder V5	ATI2565			*		*	
VGA Wonder+	ATIPLUS256		*	*		*	
VGA Edge	EDGE256		*				
VGA Edge 16	EDGE256		*				
Compaq 486c Portable	LCD active matrix display			*			
CompuAdd							
Hi Rez VGA	ET4000256			*		*	(1)
Hi Rez VGA	ET4000256a					(3)	
Hi Rez VGA	ET4000256ni						*
Genoa							
Super VGA 5200	GENOA256			*		*	
Super VGA 5300	GENOA256			*		*	
Super VGA 5400	GENOA256			*		*	
IBM							
VGA	VGA256	*					
Microlabs							
VGA Solution	MICRO3256			*	*		
Ultimate VGA	MICRO4256			*	*	(1)	
Ultimate VGA	MICRO4256a					(3)	
Ultimate VGA	MICRO4256ni						*
Orchid							
Designer VGA	ORV256			*		*	
Designer 800 VGA	ORD800256			*		*	
ProDesigner VGA	ORV256			*		*	
Designer VGA	ORV256f					(2)	
ProDesigner VGA	ORV256f					(2)	
ProDesigner II	ORII256			*		*	(1)
ProDesigner II	ORII256a					(3)	
ProDesigner II	ORII256ni						*
Paradise							
VGA Plus	P256		*				
VGA Plus 16	P256		*				
VGA Professional	P256		*	*			
VGA 1024	P256_1024		*	*			
Sigma Designs							
VGA Legend	LEGEND256			*		*	(1)
VGA Legend	LEGEND256a					(3)	
VGA Legend	LEGEND256ni						*
STB							
VGA Extra/EM	STB256			*		*	
VGA Extra/EM 16	STB256			*		*	
VGA Extra/EM 16	STB256+			*		*	(1)
VGA Extra/EM 16 Plus	STB256+ni						*
SunRiver VGA+	P256		*				

VGA Boards Supported, Continued

Name	*Board*	*Resolution*					
		320x200	*640x400*	*640x480*	*720x540*	*800x600*	*1024x768*
Tecmar							
VGA AD	TECMAR256			*		*	
Trident							
VGA 8800	T88256		*	*			
VGA 8900	T89256			*		*	(1)
VGA 8900	T89256a					(3)	
VGA 8900	T89256ni						*
Tseng Labs							
VGA	ET4000256			*		*	(1)
VGA	ET4000256a					(3)	
VGA	ET4000256ni						*
Video 7 (Headland)							
VRAM VGA	V7256		*	*	*	*	
VRAM VGA	V7256a				(3)		
1024i VGA	1024i256		*	*			
FastWrite VGA	FW256		*	*			

Legend:

1. Interlaced display.
2. For use on fixed frequency monitors such as IBM 8514 or Seiko 1430.
3. Alternate configuration providing a better display on some monitors.

For example, the following configuration file entry fully specifies display 0 to be a standard VGA console display 320 pixels wide by 200 pixels high running on a screen 12 inches wide and 9 inches high:

```
display     V256     "VGA256 320x200 12x9"     0.0     /dev/console
keyboard    AT       101                       0.0     /dev/console:/dev/vt%02d
mouse       LOGI-S   "1200 3"                  0.0     /dev/tty00
```

SunRiver workstations are specified in the configuration file by indicating the SunRiver device name in place of **/dev/console**, i.e., **/dev/st00** for the first SunRiver display. This example defines SunRiver VGA+ unit 0 as display 1 with a LOGITECH serial mouse on the DB-25 connector:

```
display     V256     "P256 640x400 12x9"       1.0     /dev/st00
keyboard    AT       101                       1.0     /dev/st00:/dev/st0%d
mouse       LOGI-S   "1200 3"                  1.0     /dev/ser00
```

SEE ALSO

X(1), Xserver(1), Xvga(1), xconfig(1), xinit(1), Xconfig(5).

NAME

Xvga – VGA (and EGA) server for X11

SYNOPSIS

Xvga [option] ...

DESCRIPTION

Xvga is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System Version 2.2 or later, using a VGA or EGA display adapter. Many "enhanced" versions of these adapter standards are supported. The *Xvga* server also supports SunRiver Corporation workstations. For users without an "enhanced" display adapter, the *Xvga* server is able to provide an effective display area larger than that provided by standard EGA or VGA display adapters. See "panning" below for more details.

Xvga operates on a 386- or 486-based ISA, EISA, or Micro Channel computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

Xvga supports a *PseudoColor* visual type when used with a color monitor and *GrayScale* visual type when used with a monochrome monitor.

OPTIONS

Options are described under *Xserver*(1).

CONFIGURATION

Xvga requires the *Type* field in the configuration file (see *Xconfig*(5)) to be one of the following types:

EGA For a 100 percent register compatible Enhanced Graphics Adapter (EGA) display (including "enhanced" EGA's).

VGA For a 100 percent register compatible Video Graphics Array (VGA) display (including "enhanced" VGA's).

The *Info* field is used to specify the board, resolution, number of colors to be used, and the width and height of the attached monitor. The format of this field is:

"BOARD XPIXELSxYPIXELS COLORS WIDTHxHEIGHT"

The WIDTHxHEIGHT argument describes the size of the display and is specified in whole inches. Users of one of the panning modes should specify the true width and height. The *Xvga* server will handle the conversions needed for clients internally.

The following table shows the VGA boards (and resolutions) supported. All modes are in non-interlaced mode unless otherwise noted:

VGA Boards Supported						
		Resolution				
Name	*Board*	*640x480*	*720x540*	*800x560*	*800x600*	*1024x768*
ATI						
VGA Wonder	ATIVGA				*	
VGA Wonder V4	ATIVGA4				*	
VGA Wonder V5	ATIVGA5				*	
VGA Wonder+	ATIVGA+				*	
VGA Edge	ATIVGA4				*	
VGA Edge16	ATIVGA5				*	
CompuAdd						
Hi Rez VGA	ET4000				*	
Hi Rez VGA	ET4000a				(4)	
Dell VGA	DELL		*		*	
Genoa						
Super VGA 5200	GVGA				*	(1)
Super VGA 5300	GVGA				*	(1)
Super VGA 5400	GVGA				*	(1)
Super VGA 5400	GVGAni					*
IBM						
VGA	VGA	*				
Microlabs						
VGA Solution	MICRO3			*	*	(1)
Ultimate VGA	MICRO4				*	
Ultimate VGA	MICRO4a				(4)	
Orchid						
Designer VGA	ORVGA				*	(1)
Designer 800 VGA	ORVGA800				*	
ProDesigner VGA	ORVGA				*	(1)
ProDesigner VGA	ORVGAni					*
Designer VGA	ORVGAf				(3)	(3)
ProDesigner VGA	ORVGAf				(3)	(3)
ProDesigner II	ORVGAII				*	
ProDesigner II	ORVGAIIa				(4)	
Paradise						
VGA Plus	PVGA1A				*	
VGA Plus 16	PVGA1A				*	
VGA Professional	PVGA1A				*	
VGA 1024	PVGA1024				*	(1)
Quadram						
VGA Spectra	QVGA			*	*	(1)
Sigma						
VGA/H	SIGMA/H			*	*	
VGA/HP16	HP16				*	
VGA Legend	LEGEND				*	
VGA Legend	LEGENDa				(4)	
STB						
VGA Extra/EM	STBVGA			*	*	(1)
VGA Extra/EM 16	STBVGA			*	*	(1)
VGA Extra/EM 16	STBVGAni					*
VGA Extra/EM 16 Plus	STBVGA+				*	
SunRiver VGA+	PVGA1A				*	

VGA Boards Supported, Continued						
		Resolution				
Name	*Board*	*640x480*	*720x540*	*800x560*	*800x600*	*1024x768*
Tatung TVGA-8	CIRRUS		*		*	
Tecmar						
VGA	TVGA			*	*	
VGA AD	TVGA			*	*	(1)
VGA AD	TVGAni					*
Trident						
VGA 8800	T88VGA				*	(1)
VGA 8900	T89VGA				*	(1)
VGA 8900	T89VGAa				(4)	
VGA 8900	T89VGAni					*
Tseng Labs						
VGA	ET4000				*	
VGA	ET4000a				(4)	
Video 7 (Headland)						
VRAM VGA	VRAM		*		*	(2)
VRAM VGA	VRAMa				(4)	
FastWrite VGA	FASTWRITE		*		*	
FastWrite VGA	FASTWRITEa				(4)	
Vega VGA	VEGA		*		*	
1024i VGA	1024		*		*	(1)
1024i VGA	1024a					(4)

Legend:

1. Interlaced display.
2. Support limited to 2 or 4 colors.
3. For use on fixed frequency monitors such as IBM 8514 or Seiko 1430.
4. Alternate configuration providing a better display on some monitors.

The following table shows the EGA boards (and resolutions) supported:

EGA Boards Supported				
		Resolution		
Name	*Board*	*640x350*	*640x480*	*800x600*
IBM				
EGA	EGA	*		
Paradise				
AutoSwitch EGA 480	PEGA		*	
SunRiver EGA+	PEGA		*	
Genoa				
Super EGA	GEGA		*	
Super EGA Hi Res	GEGA		*	*

In addition to the above extended modes, the *Xvga* server supports panning of the visible screen area across a larger effective screen. Panning the visible screen means using the screen as a window onto a larger display area. When the mouse travels to an edge of the screen, the entire screen will appear to slide over the full display area. This allows users without "extended" display adapters to have a larger work area than would otherwise be available.

The following table shows the panning modes supported:

Panning Modes Supported				
		Effective Resolution		
Name	*Board*	*1024x480*	*800x600*	*640x800*
EGA	EGAPAN	*	*	*
VGA	VGAPAN	*	*	*
Paradise AutoSwitch EGA 480	PEGAPAN	*	*	*
SunRiver EGA+	PEGAPAN	*	*	*
Genoa				
Super EGA	GEGAPAN	*	*	*
Super EGA Hi Res	GEGAPAN	*	*	*

For the EGAPAN type, the visible resolution is 640x350. The other types have a visible resolution of 640x480.

All displays work using 2, 4, and 16 colors unless otherwise noted. When displaying a resolution of 1024x768, the *Xvga* server uses the entire area reserved for video memory, which requires that there be no other video adapters which use video memory, such as Monochrome (MDA) or Color (CGA) adapters, in the system.

For example, the following configuration file entry fully specifies display 0 to be a standard VGA console display 640 pixels wide by 480 pixels high with 16 colors running on a screen 12 inches wide and 9 inches high:

```
display     VGA       "VGA 640x480 16 12x9"   0.0   /dev/console
keyboard    AT        101                     0.0   /dev/console:/dev/vt%02d
mouse       LOGI-S    "1200 3"                0.0   /dev/tty00
```

This configuration file entry specifies display 0 to be an EGA console display 640 pixels wide by 350 pixels high capable of panning to a resolution of 800 pixels wide by 600 pixels high:

```
display    EGA    "EGAPAN 800x600 16 12x9"   0.0   /dev/console
```

SunRiver workstations are specified in the configuration file by indicating the SunRiver device name in place of **/dev/console**, i.e., **/dev/st00** for the first SunRiver display. This example defines SunRiver EGA+ unit 0 as display 1 with a LOGITECH serial mouse on the DB-25 connector:

```
display     EGA       "PEGA 640x480 16 12x9"   1.0   /dev/st00
keyboard    AT        101                      1.0   /dev/st00:/dev/st0%d
mouse       LOGI-S    "1200 3"                 1.0   /dev/ser00
```

A non-EGA+ SunRiver would substitute the following for the "display" line:

```
display    EGA    "EGA 640x350 16 12x9"   1.0   /dev/st00
```

SEE ALSO

X(1), Xserver(1), Xv256(1), xconfig(1), xinit(1), Xconfig(5).

NAME

Xviking – Moniterm 21/91 Viking Smart Board Interface server for X11

SYNOPSIS

Xviking [option] ...

DESCRIPTION

Xviking is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX System, Version 2.2 or later, using a Moniterm 21/91 Viking display adapter. *Xviking* operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment. The display adapter board must be installed in a 16-bit slot because the *Xviking* server supports only this configuration.

OPTIONS

Options are described under *Xserver*(1).

CONFIGURATION

Xviking requires the *Type* field in the configuration file (see *Xconfig*(5)) to be VIKING.

The *Info* field is used to specify the resolution and the number of colors to be used and the width and height of the attached monitor. The format of this field is "XPIXELSxYPIXELS colors WIDTHxHEIGHT." The WIDTHxHEIGHT argument is specified in whole inches. The *colors* argument is 256.

For example, the following configuration file entry fully specifies display 0 to be a standard VIKING monochrome display 1280 pixels wide by 960 pixels high on a Sigma Designs 21-inch monitor:

```
display     VIKING    "1280x960 256 15x11"   0.0   /dev/console
keyboard    AT        101                    0.0   /dev/console:/dev/vt0%d
mouse       LOGI-S    "1200 3"               0.0   /dev/tty00
```

Currently, this is the only configuration supported.

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5).

NAME

Xwge – Bell Technologies Workstation Graphics Engine (Blit Express) server for X11

SYNOPSIS

Xwge [option] ...

DESCRIPTION

Xwge is the server for INTERACTIVE X11 running on an INTERACTIVE UNIX Operating System Version 2.2 or later, using the Bell Technologies Workstation Graphics Engine video processor, more commonly referred to as the Blit. This release of the server supports the Blit in monochrome mode only, with a resolution of 1664 x 1200 pixels. *Xwge* operates on a 386- or 486-based ISA or EISA computer. It is normally invoked by a program such as *xinit*(1) or by the INTERACTIVE Easy Windows Environment.

OPTIONS

–config *configuration file*

Specifies an alternate configuration file to be used, instead of **/usr/lib/X11/Xconfig**.

Other options are described under *Xserver*(1).

CONFIGURATION

Xwge requires the *Type* field in the configuration file (see *Xconfig*(5)) to be WGE.

The *Info* field is used to specify the memory and I/O addresses used to access the board, as well as the width and height of the attached monitor. This field may be freely formatted, and it is composed of strings of the following forms:

mem=*value*

This specifies the physical memory address where the Blit's *Graphics Memory* is located. This value defaults to 0xD80000, which is the board's address as shipped. The value may be specified in hexadecimal, octal, or decimal, using the standard C programming language conventions.

reg=*value*

This specifies the address of the Blit's *Graphics I/O* area. The default value is 0xE80000.

io=*value*

This specifies the I/O port used to access the Blit's **CONFIG** register. This defaults to 0x304, which is recommended to avoid conflicts with third-party tape drives.

WIDTHxHEIGHT

This specifies the monitor's width and height, which should be specified in whole inches. The default is 14x11, which is the size of the Moniterm display shipped with the Blit.

The following configuration file entry fully specifies display 0 to be the Blit configured with the factory defaults:

```
display     WGE     "mem=0xD80000 reg=0xE80000 io=0x304 14x11"   0.0   /dev/blit
keyboard    AT      101                                          0.0   /dev/console:/dev/vt%02d
mouse       LOGI-S  "1200 3"                                     0.0   /dev/tty00
```

Notes for Users of Cached Systems

The Blit's onboard memory is mapped into the 80386's memory space in the high end of the standard 16-megabyte address space. Systems which implement memory caching may have problems due to the cache controller intercepting reads and writes destined for the Blit, causing the Blit to behave slowly or not at all. The usual symptom of this is sluggish performance and the server occasionally printing the message:

"Warning: board still BUSY. Hardware problem?"

If this happens, caching needs to be disabled for the memory area used by the Blit. On many systems using the Intel 82385 cache controller, such as COMPAQ, Dell, and CSS, the solution is to add 0x80000000 to the address, causing the cache controller to not cache that particular access. In this case, the **mem=** and **reg=** values in the *Xconfig* file should be **mem=***0x80D80000* and **reg=***0x80E80000*.

The actual line in the configuration file for a cached system would be:

```
display    WGE    "mem=0x80D80000 reg=0x80E80000 io=0x304 14x11"   0.0   /dev/blit
```

Note that this change can also be made using the **sysadm** menu **configmgmt** option. Other systems, such as older Mylex motherboards, may require moving a jumper or possibly replacing a PAL or PROM to disable caching for the Blit.

FILES

/usr/lib/X11/Xconfig server device configuration file

SEE ALSO

X(1), Xserver(1), xconfig(1), xinit(1), Xconfig(5), blit(7).

NAME

gv.sh, getvesa – sysadm program to get VESA information

SYNOPSIS

gv.sh [**–v**] [**–f**] [**–d**]
getvesa type [files ...]

DESCRIPTION

gv.sh is run automatically from the sysadm menu **Software/x/configmgmt**. *gv.sh* calls *getvesa* for each of the three types: VGA, V256, and GP. *gv.sh* then builds the appropriate Xconfig files in the respective directories under **/usr/lib/X11/Xservers**. If *getvesa* exits successfully, *gv.sh* concatenates the output of *getvesa* with **Xconfig.base** and puts the result in **Xconfig**. Otherwise, it runs the *sed* script **Xconfig.sed** on the **Xconfig.base** file to remove the VESA (Video Electronics Standards Association) entry in the manufacturer's menu. It then puts the result in **Xconfig**.

The command line options for *gv.sh* are:

–f Forces the Xconfig files to be updated, even if they are not out of date.

–v Prints a message when the Xconfig files are out of date. Normally, *gv.sh* works silently.

–d Prints out information useful for debugging.

getvesa builds the appropriate menus for use by the **Xconfig** program, using information obtained from the SVPMI files. *getvesa* is run automatically by *gv.sh*.

FILES

Xconfig.base
Xconfig.sed

WARNINGS

These programs contain information that is dependent on the contents and location of the **Xconfig.base** files and the location of the SVPMI files.

The **Xconfig.base** and **Xconfig.sed** files must be edited with care. These files contain embedded tabs, which some editors do not preserve.

NAME

rgb – rebuild the *rgb* database from ASCII input

SYNOPSIS

rgb [output_file]

DESCRIPTION

The *rgb* utility reads, from standard input, a list of lines in the form:

red-value green-value blue-value colorname

and generates a color database for use with an X11 server. The red, green, and blue values have a range from 0 to 255.

An optional file name without suffixes may be specified. By default, the files **/usr/lib/X11/rgb.dir** and **/usr/lib/X11/rgb.pag** will be created. These are the files used by the INTERACTIVE X11 servers as a list of color database names.

The file **/usr/lib/X11/rgb.txt** is normally used as input to create the standard X11 set of named colors.

SEE ALSO

X(1), Xserver(1).

COPYRIGHT

NAME

xconfig – manipulate the X11 configuration file

SYNOPSIS

xconfig [–help] [–edit] [–list] [–delete] [–config *configuration_file*] [–dir *server_directory*] [–display *display_number*]

SUMMARY

The *xconfig* utility allows the system administrator to interactively create, edit, list, and delete X11 display server configuration information stored in the system configuration file.

OPTIONS

–help Print out a summary of command line options

–edit Allow interactive editing (or creating) of a display.

–list List the information about a display.

–delete Delete displays from the configuration file.

–config *config_file*
Specify an alternate configuration file in place of **/usr/lib/X11/Xconfig**.

–dir *server_directory*
Specify an alternate server directory in place of **/usr/lib/X11/Xservers**. This directory contains subdirectories, each of which contains a prototype *Xconfig* file.

DESCRIPTION

The *xconfig* utility reads the standard system-wide display configuration file (**/usr/lib/X11/Xconfig**, unless overidden with the **–config** option) and allows this information to be viewed, edited, or deleted by the user.

The *xconfig* utility understands an X11 "display" to be a collection of "resources" referred to by display name, where a "resource" is typically a single input or device, such as a mouse or video display. The *xconfig* utility presents the collection of resources to the user as a single group, and allows each resource to be modified interactively via menus and questions.

If invoked with the **–display** *name* option, *xconfig* will use the named display, then quit; otherwise, the top level will prompt for the display to use, perform the operation, and prompt for the next display. At the display prompt, the user can view a list of all currently-configured displays using the **l** response.

If the **–list** option was specified, *xconfig* will show the information for all resources specified for this display. It will not allow any modifications to be made.

If the **–delete** option was specified, *xconfig* will display the information for the display, then prompt the user for confirmation before actually deleting the display information. If the user confirms, all references to the named display will be removed from the configuration file.

Otherwise, *xconfig* will enter interactive editing mode. It first presents a menu of resources to use, such as *keyboard*, *mouse*, and *display*. Once the user selects a resource, *xconfig* will present another menu

listing all possible choices for this type of resource. For example, assuming the user chose *display*, he would be presented a menu that includes **EGA** and **VGA** displays, among others. Next, *xconfig* will provide device-specific menus and questions based on the particular selection. The actual data describing each resource type and device-specific data for each device is contained in the *Xconfig* files in each subdirectory of the **/usr/lib/X11/Xservers** directory. Once the user has made all the changes, *xconfig* will print the new resource settings and ask for confirmation before saving the changes. If the user confirms, all old information for the named display will be updated with the changed data, and any new data will be appended to the end of the *xconfig* file.

Any time that *xconfig* presents a menu or asks a question, if a default answer exists, it will be shown inside brackets, and simply entering a carriage return will select the default. If no default exists, you must select from the menu choices or enter an acceptable answer to a question. Most menus and questions also allow using **q** to quit that part of *xconfig* without affecting the current display. At all times, all valid menu choices will be shown to the user.

FILES

/usr/lib/X11/Xconfig	standard system configuration file
/usr/lib/X11/Xservers	server directory
/usr/lib/X11/Xservers/*/Xconfig	prototype *Xconfig* files for each server

SEE ALSO

Xconfig(5).

NAME

xfront – front-end to X(1) server programs

SYNOPSIS

xfront [Xserver options **] [**: display number **]**

DESCRIPTION

For the display specified, *xfront* reads the **/usr/lib/X11/Xconfig** file (see *Xconfig*(5)), scans the directories in **/usr/lib/X11/Xservers/*** for matching *Xconfig* "display" entries, and invokes the corresponding server. If no display is given, *xfront* uses the default server *:0*.

For example, to invoke *X :0* (where **/usr/bin/X11/X** is linked to **/usr/bin/X11/xfront** as distributed), *xfront* will scan the **/usr/lib/X11/Xconfig** file for the line specifying the "display" resource for display **0**. It will then search each subdirectory of the **Xservers** directory for an *Xconfig* file with a matching "display" type resource. The actual name of the server to invoke is **X<***dirname***>**, which *xfront* will execute.

FILES

/usr/lib/X11/Xconfig
/usr/lib/X11/Xservers/*/Xconfig

SEE ALSO

X(1), Xserver(1), xdm(1), xinit(1), Xconfig(5).

NAME

xpcterm – PC-compatible terminal emulator for X

SYNOPSIS

xpcterm [–*toolkitoption* ...] [–option ...]

DESCRIPTION

The *xpcterm* program is a terminal emulator for INTERACTIVE X11. It provides ANSI 3.64-compatible terminal emulation (also known under the INTERACTIVE UNIX Operating System as AT386) for programs that cannot use the window system directly. If the underlying operating system supports terminal resizing capabilities (for example, the SIGWINCH signal in systems derived from 4.3 BSD), *xpcterm* will use the facilities to notify programs running in the window whenever it is resized.

The *xpcterm* emulator is based on the *xterm* terminal emulator, but has been stripped of features that are less useful to users of PC-compatible terminals, such as Tektronix 4014 emulation. In addition, *xpcterm* can produce PC scan codes rather than ASCII characters, which allows for a higher degree of PC emulation in some applications.

OPTIONS

The *xpcterm* terminal emulator accepts all of the standard X Toolkit command line options as well as the additional options listed below (if the option begins with a "+" instead of a "–," the option is restored to its default value):

–help This causes *xpcterm* to print out a verbose message describing its options.

–ah This option indicates that *xpcterm* should always highlight the text cursor and borders. By default, *xpcterm* will display a hollow text cursor whenever the focus is lost or the pointer leaves the window.

+ah This option indicates that *xpcterm* should do text cursor highlighting.

–b *number*
This option specifies the size of the inner border (the distance between the outer edge of the characters and the window border) in pixels. The default is "2."

–cc *characterclassrange*:*value*[,...]
This sets classes indicated by the given ranges for use in selecting by words. See the section specifying character classes.

–cr *color*
This option specifies the color to use for the text cursor. The default is to use the same foreground color that is used for text.

–cu This option indicates that *xpcterm* should work around a bug in the *curses*(3x) cursor motion package that causes the *more*(1) program to display lines that are exactly the width of the window and that are followed by a line beginning with a tab to be displayed incorrectly (the leading tabs are not displayed).

+cu This option indicates that *xpcterm* should not work around the *curses*(3x) bug mentioned above in option **−cu**.

−e *program* [*arguments ...*]
This option specifies that the program (and its command line arguments) are to be run in the *xpcterm* window. It also sets the window title and icon name to be the basename of the program being executed if neither **−T** nor **−n** are given on the command line. *This must be the last option on the command line.*

−fb *font* This option specifies a font to be used when displaying bold text. This font must be the same height and width as the normal font. If only one of the normal or bold fonts is specified, it will be used as the normal font and the bold font will be produced by overstriking this font. By default, there is no bold font and *xpcterm* will overstrike the normal font.

−j This option indicates that *xpcterm* should do jump scrolling. Normally, text is scrolled one line at a time; the **−j** option allows *xpcterm* to move multiple lines at a time so that it does not fall as far behind. Its use is strongly recommended since it makes *xpcterm* much faster when scanning through large amounts of text.

+j This option indicates that *xpcterm* should not do jump scrolling.

−l This option indicates that *xpcterm* should send all terminal output to a log file as well as to the screen. This option can be enabled or disabled using the "*xpcterm* X11" menu.

+l This option indicates that *xpcterm* should not do logging.

−lf *filename*
This option specifies the name of the file to which the output log described above is written. If *file* begins with a pipe symbol (|), the rest of the string is assumed to be a command to be used as the endpoint of a pipe. The default *filename* is "**XPctermLog.***XXXXX*" (where *XXXXX* is the process ID of *xpcterm*), and it is created in the directory from which *xpcterm* was started (or in the user's home directory, in the case of a login window).

−ls This option indicates that the shell that is started in the *xpcterm* window be a login shell (i.e., the first character of **argv[0]** will be a dash, indicating to the shell that it should read the user's **.login** or **.profile**).

+ls This option indicates that the shell that is started should not be a login shell (i.e., it will be a normal "subshell").

−mb This option indicates that *xpcterm* should ring a margin bell when the user types near the right end of a line. This option can be turned on and off from the "Modes" menu.

+mb This option indicates that the margin bell should not be rung.

−ms *color*
This option specifies the color to be used for the pointer cursor. The default is to use the foreground color.

−nb *number*
This option specifies the number of characters from the right end of a line at which the margin bell, if enabled, will ring. The default is 10.

−noisc This option indicates that *xpcterm* will not assume that it is connected to an INTERACTIVE server when emitting PC scan codes. However, it will assume a U.S. keyboard layout when determining which scancode to emit.

−rw This option indicates that reverse-wraparound should be allowed. This allows the cursor to back up from the leftmost column of one line to the rightmost column of the previous line. This is very useful for editing long shell command lines and is encouraged. This option can be turned on and off from the "Modes" menu.

+rw This option indicates that reverse-wraparound should not be allowed.

−s This option indicates that *xpcterm* may scroll asynchronously, meaning that the screen does not have to be kept completely up to date while scrolling. This allows *xpcterm* to run faster when network latencies are very high and is typically useful when running across a very large internet or many gateways.

+s This option indicates that *xpcterm* should scroll synchronously.

−sb This option indicates that some number of lines that are scrolled off the top of the window should be saved and that a scrollbar should be displayed so that those lines can be viewed. This option may be turned on and off from the "Modes" menu.

+sb This option indicates that a scrollbar should not be displayed.

-sc This option indicates that *xpcterm* should emit PC scan codes instead of ASCII characters in response to keyboard events. This allows certain programs (such as INTERACTIVE Systems Corporation's VP/ix Environment) to perform much better PC emulation. This feature can also be enabled or disabled from the "Modes" menu.

+sc This option indicates that *xpcterm* should emit ASCII characters rather than PC scan codes.

−si This option indicates that output to a window should not automatically reposition the screen to the bottom of the scrolling region. This option can be turned on and off from the "Modes" menu.

+si This option indicates that output to a window should cause it to scroll to the bottom.

−sk This option indicates that pressing a key while using the scrollbar to review previous lines of text should cause the

window to be repositioned automatically in the normal position at the bottom of the scroll region.

+sk This option indicates that pressing a key while using the scrollbar should not cause the window to be repositioned.

−sl *number*
This option specifies the number of lines to save that have been scrolled off the top of the screen. The default is 64.

−sw This option indicates that *xpcterm* should never send a SIGWINCH signal on window resizes, even on systems that support the SIGWINCH signal. This is useful if sending a SIGWINCH signal to the controlling process would cause some undesirable behavior. This feature can also be enabled or disabled from the "Modes" menu.

+sw This option indicates that a change of window size should cause *xpcterm* to send a SIGWINCH signal to the controlling process. This option will have no effect on systems that do not support the SIGWINCH signal. This feature can also be enabled or disabled from the "Modes" menu.

−tm *string*
This option specifies a series of terminal setting keywords followed by the characters that should be bound to those functions, similar to the *stty* program. This is ignored when **−L** is given since *getty* resets the terminal. Allowable keywords include: **intr, quit, erase, kill, eof, eol, swtch, start, stop, brk, susp, dsusp, rprnt, flush, weras,** and **lnext**. Control characters can be specified as **^***char* (e.g. **^c** or **^u**), and **^?** may be used to indicate delete.

−tn *name*
This option specifies the name of the terminal type to be set in the TERM environment variable. This terminal type must exist in the *termcap(5)* or *terminfo(5)* database and should have *li#* and *co#* entries.

−ut This option indicates that *xpcterm* should not write a record into the system log file **/etc/utmp**.

+ut This option indicates that *xpcterm* should write a record into the system log file **/etc/utmp**.

−vb This option indicates that a visual bell is preferred over an audible one. Instead of ringing the terminal bell whenever a CTRL-G (**^g**) is received, the window will be flashed.

+vb This option indicates that a visual bell should not be used.

−C This option indicates that this window should receive console output. This option is not supported on all systems.

−S*ccn* This option specifies the last two letters of the name of a pseudoterminal to use in slave mode. This allows *xpcterm* to be used as an input and output channel for an existing program and is sometimes used in specialized applications.

The following command line arguments are provided for compatibility with older versions of *xterm*, from which *xpcterm* was derived. They may not be supported in the next release because the X Toolkit provides standard options that accomplish the same task.

#geom This option specifies the preferred position of the icon window. It is shorthand for specifying the **iconGeometry* resource.

−T *string*
This option specifies the title for *xpcterm*'s windows. It is equivalent to **−title**.

−n *string*
This option specifies the icon name for *xpcterm*'s windows. It is shorthand for specifying the **iconName* resource. Note that this is not the same as the toolkit option **−name** (see below). The default icon name is the application name.

−r This option indicates that reverse video should be simulated by swapping the foreground and background colors. It is equivalent to **−reversevideo** or **−rv**.

−w *number*
This option specifies the width in pixels of the border surrounding the window. It is equivalent to **−borderwidth** or **−bw**.

−L This option indicates that *xpcterm* was started by *init*. In this mode, *xpcterm* does not try to allocate a new pseudoterminal as *init* has already done so. In addition, the system program *getty* is run instead of the user's shell. *This option has been superseded by the new xdm program; furthermore, this option should never be used by users when starting terminal windows.*

The following standard X Toolkit command line arguments are commonly used with *xpcterm*:

−bg *color*
This option specifies the color to use for the background of the window. The default is "white."

−bd *color*
This option specifies the color to use for the border of the window. The default is "black."

−bw *number*
This option specifies the width in pixels of the border surrounding the window.

−fg *color*
This option specifies the color to use for displaying text. The default is "black".

−fn *font* This option specifies the font to be used for displaying normal text. The default is "vtsingle."

−name *name*
This option specifies the application name under which resources are to be obtained, rather than the default executable file name. *Name* should not contain "." or "*" characters.

−title *string*
This option specifies the window title string, which may be displayed by window managers if the user so chooses. The default title is the command line specified after the **−e** option, if any, otherwise the application name.

−rv This option indicates that reverse video should be simulated by swapping the foreground and background colors.

−geometry *geometry*
This option specifies the preferred size and position of the window; see *X*(1).

−display *display*
This option specifies the X server to contact; see *X*(1).

−xrm *resourcestring*
This option specifies a resource string to be used. This is especially useful for setting resources that do not have separate command line options.

−iconic This option indicates that *xpcterm* should ask the window manager to start it as an icon rather than as the normal window.

X DEFAULTS

The program understands all of the core X Toolkit resource names and classes as well as:

iconGeometry (class **IconGeometry**)
Specifies the preferred size and position of the application when iconified. It is not necessarily obeyed by all window managers.

termName (class **TermName**)
Specifies the terminal type name to be set in the TERM environment variable.

title (class **Title**)
Specifies a string that may be used by the window manager when displaying this application.

ttyModes (class **TtyModes**)
Specifies a string containing terminal setting keywords and the characters to which they may be bound. This option is ignored when **−L** is given since *getty* resets the terminal. Allowable keywords include: **intr**, **quit**, **erase**, **kill**, **eof**, **eol**, **swtch**, **start**, **stop**, **brk**, **susp**, **dsusp**, **rprnt**, **flush**, **weras**, and **lnext**. Control characters may be specified as ^*char* (e.g., **^c** or **^u**) and **^?** may be used to indicate delete. This is very useful for overriding the default terminal settings without having to do an *stty* every time an *xpcterm* is started.

utmpInhibit (class **UtmpInhibit**)
Specifies whether or not *xpcterm* should try to record the user's terminal in **/etc/utmp**.

The following resources are specified as part of the "at386" widget (class "AT386"):

allowSendEvents (class **AllowSendEvents**)
Specifies whether synthetic key and button events (generated using the X protocol SendEvent request) should be interpreted or discarded. The default is "false," meaning they are discarded. *Note that allowing such events creates a very large security hole.*

alwaysHighlight (class **AlwaysHighlight**)
Specifies whether *xpcterm* should always display a highlighted text cursor. By default, a hollow text cursor is displayed whenever the pointer moves out of the window or the window loses the input focus.

font (class **Font**)
Specifies the name of the normal font. The default is "ega."

boldFont (class **Font**)
Specifies the name of the bold font. By default, there is no bold font and bold characters are produced by overstriking the normal font.

charClass (class **CharClass**)
Specifies comma-separated lists of character class bindings of the form *[low-]high:value*. These are used in determining which sets of characters should be treated the same when doing cut and paste. See the section on specifying character classes.

curses (class **Curses**)
Specifies whether the last column bug in *curses*(3x) should be worked around. The default is "false."

background (class **Background**)
Specifies the color to use for the background of the window. The default is "black."

foreground (class **Foreground**)
Specifies the color to use for displaying text in the window. Setting the class name instead of the instance name is an easy way to have everything that would normally appear in the "text" color change color. The default is "white."

cursorColor (class **Foreground**)
Specifies the color to use for the text cursor. The default is "white."

geometry (class **Geometry**)
Specifies the preferred size and position of the window.

internalBorder (class **BorderWidth**)
Specifies the number of pixels between the characters and the window border. The default is "2."

jumpScroll (class **JumpScroll**)
Specifies whether or not jump scroll should be used. The default is "false".

logFile (class **Logfile**)
Specifies the name of the file to which a terminal session is logged. The default is "**XPctermLog.***XXXXX*" (where *XXXXX* is the process ID of *xpcterm*).

logging (class **Logging**)
Specifies whether a terminal session should be logged. The default is "false."

logInhibit (class **LogInhibit**)
Specifies whether terminal session logging should be inhibited. The default is "false."

loginShell (class **LoginShell**)
Specifies whether the shell to be run in the window should be started as a login shell. The default is "false."

marginBell (class **MarginBell**)
Specifies whether the bell should be run when the user types near the right margin. The default is "false."

multiScroll (class **MultiScroll**)
Specifies whether asynchronous scrolling is allowed. The default is "false."

nMarginBell (class **Column**)
Specifies the number of characters from the right margin at which the margin bell should be rung, when enabled.

noiscserver (class **NOiscserver**)
Indicates that *xpcterm* should not assume that it is connected to an INTERACTIVE server when emitting scan codes. However, it will assume a U.S. keyboard layout when determining which scan code to emit.

pointerColor (class **Foreground**)
Specifies the color of the pointer. The default is "black."

pointerShape (class **Cursor**)
Specifies the name of the shape of the pointer. The default is "xterm."

reverseVideo (class **ReverseVideo**)
Specifies whether reverse video should be simulated. The default is "false."

reverseWrap (class **ReverseWrap**)
Specifies whether reverse-wraparound should be enabled. The default is "false."

saveLines (class **SaveLines**)
Specifies the number of lines to save beyond the top of the screen when a scrollbar is turned on. The default is "64."

scanCodes (class **ScanCodes**)
Indicates that *xpcterm* should emit PC scan codes instead of ASCII characters in response to keyboard events. The default is "false."

scrollBar (class **ScrollBar**)
Specifies whether the scrollbar should be displayed. The default is "false."

scrollInput (class **ScrollCond**)
Specifies whether output to the terminal should automatically cause the scrollbar to go to the bottom of the scrolling region. The default is "true."

scrollKey (class **ScrollCond**)
Specifies whether pressing a key should automatically cause the scrollbar to go to the bottom of the scrolling region. The default is "false."

signalInhibit (class **SignalInhibit**)
Specifies whether the entries in the "xpcterm X11" menu for sending signals to *xpcterm* should be disallowed. The default is "false."

titeInhibit (class **TiteInhibit**)
Specifies whether or not *xpcterm* should remove *ti* or *te* termcap entries (used to switch between alternate screens on startup of many screen-oriented programs) from the TERMCAP string.

translations (class **Translations**)
Specifies the key and button bindings for menus, selections, "programmed strings," etc. See **KEY/BUTTON BINDINGS** below.

visualBell (class **VisualBell**)
Specifies whether a visible bell (i.e., flashing window) should be used instead of an audible bell when CTRL **g** is received. The default is "false."

sendSigWinch (class **SendSigWinch**)
Specifies whether a SIGWINCH signal is sent to the controlling process when the *xpcterm* window is resized. This resource is only meaningful on systems that provide the SIGWINCH signal. Otherwise, no signal is sent, regardless of the value of this resource. The default is "true."

The following resources are specified as part of the "menu" widget:

menuBorder (class **MenuBorder**)
Specifies the size in pixels of the border surrounding menus. The default is "2."

menuFont (class **Font**)
Specifies the name of the font to use for displaying menu items.

menuPad (class **MenuPad**)
Specifies the number of pixels between menu items and the menu border. The default is "3."

The following resources are useful when specified for the Athena Scrollbar widget:

thickness (class **Thickness**)
Specifies the width in pixels of the scrollbar.

background (class **Background**)
Specifies the color to use for the background of the scrollbar.

foreground (class **Foreground**)
Specifies the color to use for the foreground of the scrollbar. The "thumb" of the scrollbar is a simple checkerboard pattern that alternates pixels for foreground and background color.

EMULATIONS

The ANSI X3.64 emulation is fairly complete, but the blinking text attribute is not truly supported. Instead, blinking characters are shown as bold characters or painted in an alternate color. The *termcap*(5) entries that work with *xpcterm* include "xpcterm," "AT386," "AT386-M," "ansi3.64," and "ansi". The *xpcterm* program automatically searches the *termcap* file or *terminfo* database in this order for these entries and then sets the TERM and the TERMCAP environment variables.

POINTER USAGE

Once the emulator window is created, *xpcterm* allows you to select text and copy it within the same or other windows.

The selection functions are invoked when the pointer buttons are used with no modifiers and when they are used with the "shift" key. The assignment of the functions described below to keys and buttons may be changed through the resource database; see **KEY/BUTTON BINDINGS** below.

Pointer button one (usually on the left) is used to save text into the cut buffer. Move the cursor to the beginning of the text, and then hold the button down while moving the cursor to the end of the region and releasing the button. The selected text is highlighted, saved in the global cut buffer, and made the PRIMARY selection when the button is released. Double-clicking selects by words. Triple-clicking selects by lines. Quadruple-clicking goes back to characters, etc. Since multiple-click is determined by the time from button up to button down, you can change the selection unit in the middle of a selection. If the key/button bindings specify that an X selection is to be made, *xpcterm* will leave the selected text highlighted for as long as it is the selection owner.

Pointer button two (usually in the middle) types (pastes) the text from the PRIMARY selection, if any (and otherwise from the cut buffer), inserting it as keyboard input.

Pointer button three (usually on the right) extends the current selection (without loss of generality; that is you can swap "right" and "left" everywhere in the rest of this paragraph ...). If pressed while closer to the right edge of the selection than the left, it extends/contracts the right edge of the selection. If you contract the selection past the left edge of the selection, *xpcterm* assumes you really meant the left edge,

restores the original selection, then extends/contracts the left edge of the selection. Extension starts in the selection unit mode that the last selection or extension was performed in; you can multiple-click to cycle through them.

By cutting and pasting pieces of text without trailing new-line characters, you can take text from several places in different windows and form a command to the shell, for example, or take output from a program and insert it into your favorite editor. Because the cut buffer is globally shared among different applications, you should regard it as a "file" whose contents you know. The terminal emulator and other text programs should be treating it as if it were a text file, i.e., the text is delimited by new-line characters.

The scroll region displays the position and amount of text currently showing in the window (highlighted) relative to the amount of text actually saved. As more text is saved (up to the maximum), the size of the highlighted area decreases.

Clicking button one with the pointer in the scroll region moves the adjacent line to the top of the display window.

Clicking button three moves the top line of the display window down to the pointer position.

Clicking button two moves the display to a position in the saved text that corresponds to the pointer's position in the scrollbar.

MENUS

The *xpcterm* program has two different menus, named **xpcterm** and **Modes.** The **Modes** menu will pop up when the user presses the CTRL key and the middle mouse button simultaneously while the pointer is anywhere within the emulator window. The **xpcterm** menu can be accessed by pressing the CTRL key and the left mouse button while the pointer is anywhere within the window.

Menus are divided into two sections, separated by a horizontal line. The top portion contains various modes that can be altered. A check mark appears next to a mode that is currently active. Selecting one of these modes toggles its state. The bottom portion of the menu consists of command entries; selecting one of these performs the indicated function.

The **xpcterm** menu contains items that apply to the AT386 windows. The **Secure Keyboard** mode should be used when typing in passwords or other sensitive data; see SECURITY below. Notable entries in the command section of the menu are **Continue**, **Suspend**, **Interrupt**, **Hangup**, **Terminate**, and **Kill** which send the SIGCONT, SIGTSTP, SIGINT, SIGHUP, SIGTERM, and SIGKILL signals, respectively, to the process group of the process running under *xpcterm* (usually the shell). The **Continue** function is especially useful if the user has accidentally typed **^z**, suspending the process.

The **Modes** menu sets various modes in the AT386 emulation. The full reset entry will clear the screen, reset tabs to every eight columns, and reset the terminal modes (such as wrap and smooth scroll) to their initial states just after *xpcterm* has finished processing the command line options.

SECURITY

X environments differ in their security consciousness. The servers provided by MIT use a host-based mechanism to control access to the server (see *xhost*(1)). If you enable access for a host and other users are also permitted to run clients on that host, there is the possibility that someone will run an application that will attempt to use the basic services of the X protocol to snoop on your activities, and potentially capture a transcript of everything you type at the keyboard. This is of particular concern when you want to type in a password or other sensitive data. The best solution to this problem is for the industry to choose a standard authorization mechanism with the necessary operating system support and to incorporate this into the X protocol (which is already designed to handle such a mechanism). In the meantime, since passwords are most commonly typed to something running in an *xpcterm* window, a simple mechanism exists for protecting keyboard input in *xpcterm.*

The **xpcterm** menu (see **MENUS** above) contains a **Secure Keyboard** entry which, when enabled, ensures that all keyboard input is directed *only* to *xpcterm* (using the GrabKeyboard protocol request). When an application prompts you for a password (or other sensitive data), you can enable **Secure Keyboard** using the menu, type in the data, and then disable **Secure Keyboard** using the menu again. Only one X client at a time can secure the keyboard, so when you attempt to enable **Secure Keyboard** it may fail. In this case, the bell will sound. If the **Secure Keyboard** succeeds, the foreground and background colors will be exchanged (as if you selected the **Reverse Video** entry in the **Modes** menu); they will be exchanged again when you exit secure mode. If the colors do *not* switch, then you should be *very* suspicious that you are being spoofed. If the application you are running displays a prompt before asking for the password, it is safest to enter secure mode *before* the prompt gets displayed and to make sure that the prompt gets displayed correctly (in the new colors) to minimize the probability of spoofing. You can also bring up the menu again and make sure that a check mark appears next to the entry.

Secure Keyboard mode will be disabled automatically if your xpcterm window becomes iconified (or otherwise unmapped) or if you start up a reparenting window manager (that places a title bar or other decoration around the window) while in **Secure Keyboard** mode. (This is a feature of the X protocol not easily overcome.) When this happens, the foreground and background colors will be switched back and the warning bell will sound.

CHARACTER CLASSES

Clicking the middle mouse button twice in rapid succession will cause all characters of the same class (e.g., letters, white space, punctuation) to be selected. Because different people have different preferences for what should be selected (for example, should file names be selected as a whole or only the separate subnames), the default mapping can be overridden through the use of the *charClass* (class *CharClass*) resource.

This resource is simply a list of *range:value* pairs where the range is either a single number or *low-high* in the range of 0 to 127,

corresponding to the ASCII code for the character or characters to be set. The *value* is arbitrary, although the default table uses the character number of the first character occurring in the set.

The default table is:

```
static int charClass[128] = {
/* NUL SOH STX ETX EOT ENQ ACK BEL */
   32,  1,  1,  1,  1,  1,  1,  1,
/*  BS  HT  NL  VT  NP  CR  SO  SI */
    1, 32,  1,  1,  1,  1,  1,  1,
/* DLE DC1 DC2 DC3 DC4 NAK SYN ETB */
    1,  1,  1,  1,  1,  1,  1,  1,
/* CAN  EM SUB ESC  FS  GS  RS  US */
    1,  1,  1,  1,  1,  1,  1,  1,
/*  SP   !   "   #   $   %   &   ' */
   32, 33, 34, 35, 36, 37, 38, 39,
/*   (   )   *   +   ,   -   .   / */
   40, 41, 42, 43, 44, 45, 46, 47,
/*   0   1   2   3   4   5   6   7 */
   48, 48, 48, 48, 48, 48, 48, 48,
/*   8   9   :   ;   <   =   >   ? */
   48, 48, 58, 59, 60, 61, 62, 63,
/*   @   A   B   C   D   E   F   G */
   64, 48, 48, 48, 48, 48, 48, 48,
/*   H   I   J   K   L   M   N   O */
   48, 48, 48, 48, 48, 48, 48, 48,
/*   P   Q   R   S   T   U   V   W */
   48, 48, 48, 48, 48, 48, 48, 48,
/*   X   Y   Z   [   \   ]   ^   _ */
   48, 48, 48, 91, 92, 93, 94, 48,
/*   `   a   b   c   d   e   f   g */
   96, 48, 48, 48, 48, 48, 48, 48,
/*   h   i   j   k   l   m   n   o */
   48, 48, 48, 48, 48, 48, 48, 48,
/*   p   q   r   s   t   u   v   w */
   48, 48, 48, 48, 48, 48, 48, 48,
/*   x   y   z   {   |   }   ~ DEL */
   48, 48, 48,123,124,125,126,  1};
```

For example, the string "33:48,37:48,45-47:48,64:48" indicates that the exclamation mark, percent sign, dash, period, slash, and ampersand characters should be treated the same way as characters and numbers. This is very useful for cutting and pasting electronic mailing addresses and UNIX System file names.

KEY TRANSLATIONS

It is possible to rebind keys (or sequences of keys) to arbitrary strings for input by changing the translations for the AT386 widget. Changing the translations for events other than key and button events is not expected and will cause unpredictable behavior.

The actions available for key translations are:

secure() Toggles the **Secure Keyboard** mode; see **SECURITY**.

insert() Processes the key in the normal way, i.e., inserts the ASCII character code corresponding to the *keysym* found in the keyboard mapping table into the input stream. If the key event is an up transition and scan code mode is not set, the event is silently ignored.

string(*string*) Rebinds the key or key sequence to the string value, that is, inserts the string argument into the input stream. Quotation is necessary if the string contains white space or non-alphanumeric characters. If the string argument begins with the characters **0x**, it is interpreted as a hex character constant, and the corresponding character is sent in the normal way.

keymap(*name*) Takes a single string argument naming a resource to be used to dynamically define a new translation table; the name of the resource is obtained by appending the string *Keymap* to *name*. The keymap name **None** restores the original translation table (the very first one; a stack is not maintained). Uppercase/lowercase is significant.

insert-selection(*name*[,*name*]...)
Retrieves the value of the first (left-most) named selection that exists or cut buffer that is non-empty and inserts the value into the input stream. *Name* can be the name of any selection, for example, **PRIMARY** or **SECONDARY**, or it can be the name of a cut buffer: **CUT_BUFFER0**, **CUT_BUFFER7**. Uppercase/lowercase is significant.

For example, a debugging session might benefit from the following bindings:

```
*AT386.Translations: #override <Key>F13: keymap(dbx)
*AT386.dbxKeymap.translations: \
    <Key>F14: keymap(None) \n\
    <Key>F17: string("next") string(0x0d) \n\
    <Key>F18: string("step") string(0x0d) \n\
    <Key>F19: string("continue") string(0x0d) \n\
    <Key>F20: string("print ") insert-selection(PRIMARY, CUT_BUFFER0)
```

KEY/BUTTON BINDINGS

Within the AT386 widget the key and button bindings for selecting text, pasting text, and activating the menus are controlled by the translation bindings. In addition to the actions listed above under **KEY TRANSLATIONS**, the following actions are available:

mode-menu() Posts one of the two mode menus, depending on which button is pressed.

select-start() Unselects any previously selected text and begins selecting new text.

select-extend() Continues selecting text from the previous starting position.

start-extend() Begins extending the selection from the farthest (left or right) edge.

select-end(*name***[,***name***]...)**
Ends the text selection. The *name* is the name of a selection or the name of a cut buffer into which the text is to be copied. The *xpcterm* program will assert ownership of all the selections named and will copy the text into each of the cut buffers. Uppercase/lowercase is significant.

ignore() Quietly discards the key or button event.

bell([*volume***])** Rings the bell at the specified volume increment above/below the base volume.

The default bindings are:

```
                <KeyPress>:          insert() \n\
                <KeyRelease>:        insert() \n\
Ctrl ~Meta      <Btn1Down>:          mode-menu() \n\
     ~Meta      <Btn1Down>:          select-start() \n\
     ~Meta      <Btn1Motion>:        select-extend() \n\
Ctrl ~Meta      <Btn2Down>:          mode-menu() \n\
~Ctrl~Meta      <Btn2Down>:          ignore() \n\
     ~Meta      <Btn2Up>:            insert-selection(PRIMARY, CUT_BUFFER0) \n\
~Ctrl~Meta      <Btn3Down>:          start-extend() \n\
     ~Meta      <Btn3Motion>:        select-extend() \n\
     ~Meta      <BtnUp>:             select-end(PRIMARY, CUT_BUFFER0) \n\
                <BtnDown>:           bell(0)
```

STARTING XPCTERM FROM INIT

Warning: This feature is now obsolete and may not be supported in future releases. Sites using this method should switch to xdm instead.

On operating systems such as 4.3 BSD and ULTRIX, the server and initial login window are normally started automatically by *init*(8).

By convention, the pseudoterminal with the highest minor device number (e.g., **/dev/ttyqf** and **/dev/ptyqf**) is renamed for the lowest display number (e.g., **/dev/ttyv0** and **/dev/ptyv0**). Machines that have more than one display can repeat this process using *ttyqe* for **ttyv1**, and so on.

Once the pseudoterminals are in place, a line similar to the following may be added to **/etc/ttys** (replacing *Xqvss* with the appropriate server and putting it all on one line):

```
ttyv0 "/usr/bin/X11/xpcterm -L -geom 80x24+1+1 -display :0"
    xpcterm on secure window="/usr/bin/X11/Xqvss :0"
```

Sites that used to run X10 should note that the colon in the server display number is required.

Although the release will install both the X server and *xpcterm* in **/usr/bin/X11** by default, many sites choose to make a copy of both of these programs on the **root** partition (usually in **/etc**) so that they may still be used even if the partition containing **/usr/bin/X11** is not mounted.

Some versions of *init* have relatively small program name buffer sizes and treat all sharp signs as comment delimiters. Sites that wish to list large numbers of options on the xpcterm line will need to write a small shell script to exec the long *xpcterm* line. The best solution, of course, is to use *xdm*.

OTHER FEATURES

The *xpcterm* emulator automatically highlights the window border and text cursor when the pointer enters the window (selected) and unhighlights them when the pointer leaves the window (unselected). If the window is the focus window, then the window is highlighted no matter where the pointer is.

There are escape sequences to activate and deactivate an alternate screen buffer, which is the same size as the display area of the window. When activated, the current screen is saved and replaced with the alternate screen. Saving of lines scrolled off the top of the window is disabled until the normal screen is restored. The *termcap*(5) entry for *xpcterm* allows the visual editor *vi*(1) to switch to the alternate screen for editing and restore the screen on exit.

There are escape sequences to change the name of the windows and to specify a new log file name.

USING VP/ix

If you have VP/ix installed on your system, you may use *xpcterm* as a PC-compatible terminal window. To do this, set the environment variable TERM to *xpcterm*.

ENVIRONMENT

The *xpcterm* program sets the environment variables TERM and TERMCAP properly for the size window you have created. It also uses and sets the environment variable DISPLAY to specify which bitmap display terminal to use. The environment variable WINDOWID is set to the X window ID number of the *xpcterm* window.

SEE ALSO

resize(1), X(1), pty(4), tty(4), keyboard(7), display(7).

BUGS

The **−L** option is no longer needed as the new *xdm* display manager system handles logging in in a much cleaner way. It is no longer necessary to try to match colors in **/etc/ttys** or worry about an unwanted login window. *This option may be removed in future releases.*

The xpcterm program will hang forever if you try to paste too much text at one time. It is both producer and consumer for the *pty* and can deadlock.

Variable-width fonts are not handled reasonably.

This program still needs to be rewritten. It should be split into very modular sections, with the various emulators being completely separate widgets that do not know about each other. Ideally, you would like to be able to pick and choose emulator widgets and stick them into a single control widget.

The focus is considered lost if some other client (e.g., the window manager) grabs the pointer; it is difficult to do better without an addition to the protocol.

There needs to be a dialog box to allow entry of a log file name and the COPY file name.

Many of the options are not resettable after *xpcterm* starts.

This manual entry is too long. There should be a separate user's manual defining all of the non-standard escape sequences.

All programs should be written to use X directly; then this program could be eliminated.

Currently, only the "vga" and "ega" fonts that are shipped with INTERACTIVE X11 provide the correct glyphs for characters whose encoding values are greater than 127 or less than 32.

COPYRIGHTS

NAME

xttymap – configure the X11 server keyboard mapping

SYNOPSIS

xttymap [**–display** *dpy*] [**–help**] [**–noisc**] *mapfile*

DESCRIPTION

xttymap allows users of non-U.S. keyboards to remap keys on the keyboard in a method compatible with the *ttymap* utility.

The command *xttymap mapfile* will read the contents of the file *mapfile* and set the key mapping as specified in the scancode section of the input file. The format of the mapfile is described below.

Unless invoked with the **–noisc** option, *xttymap* will assume that an INTERACTIVE X11 display server is being used, and will use a built-in table to determine how to map X11 KeyCodes to keyboard scancodes. The **–noisc** option will use a server-independent method of determining which scancode corresponds to which X11 KeySym (as opposed to KeyCode) and then query the server for the correct KeyCode before changing the mapping. This option should be used with non-INTERACTIVE display servers, e.g., on X terminals, and will only work correctly when the mapping currently in effect is the one for a U.S. keyboard layout.

Mapfiles

xttymap uses the same input file format as *ttymap*, allowing users to use the same input file with both *ttymap* and *xttymap*. (A detailed description can be found in *ttymap*(1).)

New acceptable values for columns 2, 3, 4, or 5 in the input file are the English descriptions of X11 KeySyms, preceded with **XK_** e.g. **XK_***eacute*. A full list of possible KeySyms can be found in **/usr/include/X11/keysymdef.h**. *xttymap* supports the LATIN1-4, GREEK, and CYRILLIC sets of KeySyms.

Functionality

xttymap will read the input file until it reaches the scancode section. The scancode section will be examined, and the server-wide mapping will be changed according to the following rules:

Each time an entry is found with a **|N**, **|O**, or **|[** flag, the entry is ignored, along with entries containing only a dash. No action is taken for function key entries. With *ttymap*, these flags are intended to generate escape sequences or give string values to function keys. Within the X11 environment, this should be done on a per-client basis. Most X clients, such as *xterm* and *xpcterm*, provide a method to allow user-specifiable bindings for function keys.

Values found in columns 2 and 3 are used to change the keyboard mapping used when these keys are pressed alone or in conjunction with the **SHIFT** key. This is identical to what *ttymap* does. When the **CAPS** mnemonic is found in column 6, the KeySyms generated by the unshifted and shifted keys are swapped when **CAPS LOCK** is on. The **CTRL** and **NUM** mnemonics are silently ignored.

Values found in columns 4 and 5 are used to change the keyboard mapping used when these keys are pressed or shifted while the **Mod3** modifier key is depressed. This is typically used to generate keys

corresponding to a secondary character set, such as u-umlaut or c-cedilla, by pressing the key while holding the **Mod3** key. This is similar to *ttymap*'s behavior with the **ALT** keys. By default, the X server treats both the right and left **ALT** keys as **Mod1** modifier keys. Some clients exhibit special behavior when the **Mod1** key is used, so *xttymap* uses **Mod3** as the equivalent for **ALT**. (By default, the **NUM LOCK** key is bound to **Mod2**.)

To rebind the right **ALT** key to generate **Mod3**, execute the following command before using *xttymap*:

```
xmodmap -e "remove mod1 = Alt_R" -e "add mod3 = Alt_R"
```

FILES

/usr/lib/keyboard/*.map sample keyboard map files

SEE ALSO

xmodmap(1).
ttymap(1) in the *INTERACTIVE UNIX System User's/System Administrator's Reference Manual*.

NAME

Xcolors – X Window System color specification file

DESCRIPTION

The file **/usr/lib/X11/Xcolors** is used at runtime by the *X*(1) server when a static colormap is being simulated on a server that supports dynamic colormaps. It allows specification of the colors that should be supplied in the static colormap. For example, if the default visual type in the file **/usr/lib/X11/Xconfig** were specified to be *StaticColor,* for a VGA display, the *Xcolors* file could be used to specify the 16 colors that are to be provided by the colormap.

This is useful mainly on displays supporting small colormaps because when a static colormap is used, the X server will always return the closest match when a color is allocated. (In contrast, with a dynamic colormap, if the color requested cannot be allocated, the server sends the application an error. Many applications are not prepared to deal with such errors because they erroneously assume that a "color" display will have at least 256 colors available.)

The *Xcolors* file is made up of sections, each describing a colormap. There may be multiple colormaps defined in the file; however for any combination of a display, visual type, and colormap size, there may be only one colormap defined.

The first line of a colormap definition should consist of four fields and be of the form:

colormap Type Display Size

Subsequent lines should be in the form of either a single field with the name of a color, or three fields specifying the red, green, and blue components of a color. These fields are separated by commas. The number of colors specified should match the size specified in the first line of the section.

A '#' character begins a comment; characters to the end of the line will be ignored. Unused fields must be accounted for; null (dummy) fields such as "" or " can be used. An entire section will be ignored if the first line is commented out.

The following example defines a *StaticColor* colormap with 16 colors:

```
#colormap      Type         Display   Size
colormap       StaticColor  0.0       16
    black,                            # black
    white,                            # white
    0x2000,    0x2000,      0x8C00,   # navy blue
    0x0000,    0x7C00,      0xFC00,   # "true blue"
    0xA000,    0xE000,      0xFC00,   # lightish blue
    0x0000,    0xFC00,      0xFC00,   # cyan
    0x2000,    0x8C00,      0x2000,   # forest green
    0x3800,    0xDC00,      0x3800,   # lime green
    0x2000,    0xFC00,      0x8C00,   # pale green
    0xAAAA,    0x5555,      0x0000,   # brown
    0xFC00,    0x7C00,      0x0000,   # coral
    0xF800,    0xF800,      0x0000,   # yellow
    0x9800,    0x3000,      0xC000,   # purplish
```

```
0xFC00,     0x0000,     0xFC00,   # magenta
grey,                             # grey
red,                              # red
```

FILES

/usr/lib/X11/Xcolors

SEE ALSO

X(1), Xcpqag(1), Xcvc(1), Xdcaga(1), Xgp(1), Xhrc(1), Xigsp(1), Xlvp(1), Xmsfdp(1), Xnnp(1), Xnnpi(1), Xpw(1), Xrren2(1) Xserver(1), Xsp200(1), Xtisdb(1), Xv256(1), Xvga(1), Xviking(1), Xwge(1), Xconfig(5).

NAME

Xconfig – X Window System device description file

DESCRIPTION

The file **/usr/lib/X11/Xconfig** is used at runtime by the *X*(1) server to determine which input and output devices to use and how each should be configured. Each line describes exactly one device "resource."

Each non-comment line in *Xconfig* should consist of five fields and be of the form:

Resource Type Info (Default) Display Devname

Each field can be separated by any number of blanks or tabs, blank lines are permitted, and long lines may be continued by appending a '\' character to the end of the line. A '#' begins a comment; characters to the end of line will be ignored. Unused fields must be accounted for; dummy (null) fields such as "" or '' can be used. White space can be embedded within a field by quoting it within a pair of single or double quotes (' or "). White space can be escaped with a backslash (\).

Resource indicates the general class of input or output device described by this line. Currently recognized resource types are **keyboard**, **display**, **mouse**, and **tablet**.

Type is used to indicate the type of device, e.g., "VGA" or "WGE." This field is used to determine which driver is needed for the device in question. For information on supported keyboards, mice, and tablets, see the sections "Keyboards," "Mice," and "Tablets" below and consult the "INTERACTIVE X11 Release Notes" for a complete list of supported display types.

Default is an optional field specifying a default for this resource. For **display** resources, this is used to specify a default visual type. It may be set to one of the following:

StaticGray
: A colormap where the entries are pre-defined gray values, typically in a linear increasing ramp.

GrayScale
: A colormap where the entries are limited to shades of gray, but may be changed dynamically.

StaticColor
: A colormap where the entries are pre-defined colors.

PseudoColor
: A colormap where the entries may be dynamically set to any color.

TrueColor
: A colormap where the entries are pre-defined colors, and each pixel value is made up of distinct subfields for Red, Green, and Blue.

DirectColor
: A colormap where the entries may be dynamically set, and each pixel value is made up of distinct subfields for Red, Green, and Blue.

Selecting one of these types may override the visual normally used as the default by the server. If the server is capable of supporting a *PseudoColor* visual, a default of *PseudoColor, StaticColor, GrayScale,* or *StaticGray* may be chosen. If the server is capable of supporting a *GrayScale* visual, a default of *GrayScale* or *StaticGray* may be chosen.

In some cases, it is beneficial to choose a *StaticColor* or *StaticGray* visual to prevent running out of colors in a small colormap. When using a static visual, the closest match to the requested color is used, rather than generating an error as happens when using a dynamic visual.

If a server supporting a dynamic visual type is overridden with a static colormap default, a colormap will be generated automatically. It is also possible to specify which colors should be used in the colormap. See *Xcolors*(5) for details.

Info is a device-specific string used by the device-dependent portion of the X server to distinguish between subtypes of similar devices and to properly initialize the device. For further information on the exact meaning of this field, read the descriptions below for keyboards and mice, and consult your server manual for each display.

Display is used to match the description lines to the X server, using the standard display notation described in *X*(1).

Devname is the full path name to the file in **/dev/** that corresponds to the physical hardware. For the "keyboard" resource, this field contains a string that indicates the "basic" keyboard device, usually followed by a colon and a *printf*-style template used to construct the name of a virtual terminal device (see *vt*(4)) using *sprintf*(3s) and information obtained from the VT_OPENQRY *ioctl* call. Thus, for the standard console device, this field should be **/dev/console:/dev/vt%02d**.

This configuration will cause the X server for display 0 to use the standard console display as a VGA capable of 640 by 480 pixel resolution in 16 colors, the standard 101-key keyboard, and a 3-button LOGITECH bus mouse:

resource*	type	info	display	devname
display	VGA	"VGA 640x480 16 10x7"	0	/dev/console
keyboard	AT	101	0	/dev/console:/dev/vt%02d
mouse	LOGI-B	3	0	/dev/logi

*Server 0: the console VGA, the console keyboard, and a LOGITECH bus mouse, all referred to as: **:0**.

A second example is for a SunRiver EGA+ workstation capable of 640x480 16-color resolution, using unit 0 as X display number 1. The default colormap will be *StaticColor*. The mouse is a LOGITECH serial, attached to the DB-9 serial connector on the back of the SunRiver system unit:

display	EGA	StaticColor	"PEGA 640x480 16 10x7"	0	/dev/st00
keyboard	AT		101	0	/dev/st00:/dev/st0%d
mouse	LOGI-S		"1200 3"	0	/dev/ser01

Keyboards

The X Window System supports the standard AT 84- and 101-key keyboards only, and the Wyse 60 serial terminal in scancode mode. For the standard system keyboard, the *Type* field should be **AT**, and the third field (*Info*) should indicate the number of keys; i.e., **84** or **101**, although it is not currently checked. For the Wyse 60, the *Type* field should be **WY60**, and the third field (*Info*) should indicate the terminal's baud rate (one of 300, 1200, 2400, 4800, 9600, or 19200 baud) or 0 to use the current baud rate if the terminal is normally used for logins. Note that the Wyse must be in its native "WY 60" emulation mode. (See the terminal's user manual for details.) The last field (*Devname*) should contain the full path name of the serial port the terminal is connected to, and does *not* allow the **:/dev/vt%02d** notation. Because virtual terminals are currently tied into the system keyboard (and SunRiver workstations), the WY 60 keyboard type cannot be used with displays requiring VTs, but can only be used with a secondary display, such as the Bell Technologies Blit (*Xwge*) server.

Mice

X currently supports eight types of mice:

LOGITECH Serial
LOGITECH Bus
Microsoft Serial
Microsoft Bus
MSC (Mouse Systems) Serial
MSC Bus
MSC OMNIMOUSE
PS/2 and compatible keyboard mice.

The general form of the mouse *Info* field is "*speed numbuttons*"; both values are optional, and there are reasonable defaults for each type of mouse. Specific examples follow:

The LOGITECH serial mouse is supported at speeds of 1200, 2400, 4800, and 9600 baud, as specified in the configuration entry. The default speed is 1200 baud, with three buttons. The default is illustrated in this example using display 0, with the mouse attached to **/dev/tty01**:

```
mouse LOGI-S "1200 3" 0 /dev/tty01
```

The LOGITECH bus mouse requires the installation of the **logi** kernel device driver, and also defaults to three buttons. Again, for display 0, device **/dev/logi** (note that baud rate does not matter):

```
mouse LOGI-B 3 0 /dev/logi
```

The Microsoft serial mouse supports two buttons at most (the default), and a speed of 1200 baud. This example assumes the mouse is attached to a SunRiver device, **/dev/ser01**, for display 1:

```
mouse MS-S "1200 2" 1 /dev/ser01
```

The Microsoft bus mouse adapter supports up to three buttons, but the standard mouse is only equipped with two buttons, which is the default. As with the LOGITECH bus mouse, baud rate does not matter and need not be specified. The example is for display 0, using the standard mouse device **/dev/mouse**:

```
mouse MS-B 2 0 /dev/mouse
```

The MSC Technologies PC Mouse only supports three buttons and 1200 baud, so these are the defaults. This is a serial mouse connected to **/dev/tty01**:

```
mouse MSC-S "1200 3" 0 /dev/tty01
```

This is a bus mouse on **/dev/tty02**:

```
mouse MSC-B "1200 3" 0 /dev/tty02
```

The OMNIMOUSE from MSC Technologies only comes in a serial version, running at a fixed speed of 1200 baud, with two buttons:

```
mouse OMNI-S "1200 2" 0 /dev/tty01
```

The PS/2 or keyboard mouse is supported on PS/2 and compatible Micro Channel systems, as well as COMPAQ and similar AT-bus systems equipped with a mouse port integrated into the keyboard controller. The default configuration specifies two buttons, but several versions of the mouse are available and supported with three buttons. The default is illustrated in this example using display 0, with the mouse using two buttons:

```
mouse PS/2 "2" 0 /dev/kdmouse
```

In certain cases, the configuration information you choose for a mouse might not be related to the actual brand of mouse you are using. Use the following table to select the correct configuration for your mouse:

Vendor	*Type*	*Xconfig*	*Name*
LOGITECH			
	C7 – 3-button serial	LOGI-S	/dev/tty0?
	P7 – 3-button bus	LOGI-B	/dev/logi
	PC-93-9MD – Series/9 3-button bus	LOGI-B	/dev/logi
	CA-93-6MD – Series/9 3-button [plugged into serial port]	LOGI-S	/dev/tty0?
	CA-93-6MD – Series/9 3-button [plugged into on-board mouse port]	PS/2	/dev/kdmouse
	2-button serial	MS-S	/dev/tty0?
	"Series 2-7S" 2-button [plugged into on-board mouse port]	PS/2	/dev/kdmouse
Mouse Systems			
	3-button optical, serial	MSC-S	/dev/tty0?
	3-button optical, mouse [this is a serial card that requires configuring the kernel serial driver]	MSC-B	/dev/tty0?
	2-button optical, serial	MS-S	/dev/tty0?
	"OMNIMOUSE" 2-button serial	OMNI-S	/dev/tty0?
Microsoft			
	Mouse w/Bus Interface	MS-B	/dev/mouse
	Bus mouse [old style]	MS-B	/dev/mouse
	Mouse w/Serial-PS/2 Interface [plugged into serial port]	MS-S	/dev/tty0?
	Mouse w/Serial-PS/2 Interface [plugged into on-board mouse port]	PS/2	/dev/kdmouse

Tablets

X currently supports the following digitizing tablets (also referred to as *digitizing pads* or *digitizers*) and compatibles:

- Summagraphics SummaSketch Plus, models MM961 and MM1201
- Kurta IS/ONE tablets in MM961/1201 mode (switches C5 up, C6 down, C7 down, and C8 up)

The Summagraphics tablet allows the following tokens to appear in any order in the *Info* field:

Positioning (applicable to the MM961 model only):

Positioning	*Description*
horiz	Specifies that the tablet is in a horizontal orientation, with the longest axis running along the horizontal.
vertical	Specifies that the tablet has the longest axis on the vertical (default).

Type of pointer attached to the tablet:

Pointer	*Description*
stylus	Using the 3 button stylus (pen) (default)
cursor	Using the 4 button cursor (puck)

Tablet's pointing behavior:

Pointer	*Mode*
absolute	Absolute mode (like a conventional tablet), where the on-screen cursor goes exactly where the pointer is (the default).
relative	Relative mode (like a mouse); allows the user to lift the pointer up and put it back down somewhere else without changing the on-screen cursor position.

The following lines of resolution can be used:

1000 lines per inch (lpi)
500 lines per inch
400 lines per inch
200 lines per inch
100 lines per inch
40 lines per mm (1016 lpi)
20 lines per mm (508 lpi)
10 lines per mm (254 lpi)

Report rate divisor – throttles the tablet's data throughput:

1 maximum throughput (recommended for faster systems only)
2 1/2th the maximum throughput
8 1/8th the maximum throughput
32 1/32nd the maximum throughput

The following tablet baud rates can be used:

300
1200
2400
4800
9600 (The default: other settings require jumper settings inside tablet.)
19200

Notes

The higher the tablet resolution setting, the more sensitive the pointer will be to movement, but the drawback is that the tablet will generate more serial data, possibly causing the system to drop bytes on the serial port. Increasing the report rate by using a smaller divisor has a similar effect. The best compromise is to use a moderate resolution (200, 400, or 500 lines per inch) in combination with a divisor of 2, then adjust the values to suit the system speed and your taste.

When using a pointing device with the digitizing tablet, the pointer must be removed from the tablet work area when it is not in use. Because of the sensitivity of the tablet, X will detect motion if the pointer is left on the tablet, and the screen saver device will not activate.

FILES

/usr/lib/X11/Xconfig

SEE ALSO

X(1), Xcpqag(1), Xcvc(1), Xdcaga(1), Xgp(1), Xhrc(1), Xigsp(1), Xlvp(1), Xmsfdp(1), Xnnp(1), Xnnpi(1), Xpw(1), Xrren2(1) Xserver(1), Xsp200(1), Xtisdb(1), Xv256(1), Xvga(1), Xviking(1), Xwge(1), Xcolors(5).

NAME

blit – Bell Technologies Blit Express graphics card device driver

DESCRIPTION

The Blit device driver provides a minimal interface between user processes and the Blit graphics card. This driver supports only the *open*(2), *close*(2), and *ioctl*(2) system calls. The user process is expected to open the device and use the KDMAPDISP *ioctl* to map the board's memory into the user process's address space, whereupon the process can manipulate the board at will.

Supported Ioctl Commands

The following commands may be used with the *ioctl*(2) call:

KDMAPDISP

This call is used to map the on-board memory into the user's address space. This *ioctl* takes a pointer to a **struct kd_memloc**, as defined in the file **<sys/kd.h>**, indicating the memory range to map. Refer to *display*(7) for details of this structure. Note that the Blit memory is separated into two areas, referred to as the *graphics memory* and *graphics I/O* or *register memory* areas. These two areas must be mapped using two separate *ioctl* calls. Also, while the Blit's register memory area is only 256 bytes long, the values passed to KDMAPDISP must indicate a page-sized area (4096 bytes in the current implementation).

KDUNMAPDISP

This call is used to unmap the area mapped into the user's address space. If both areas have been mapped, both will be unmapped. This call should be passed an argument of 0.

FILES

/dev/blit

SEE ALSO

Xwge(1), display(7).

NAME

cpqag – COMPAQ AG1024 driver

DESCRIPTION

The COMPAQ AG1024 kernel driver facilitates accessing the COMPAQ AG1024 board. It is a regular character-based driver that manages one AG1024 board. It is used by programs that draw using the graphics modes of AG1024.

CONFIGURATION

The *sdevice*(4) file **/etc/conf/sdevice.d/cpqag** describes the I/O port addresses as well as the shared memory addresses. The seventh and eighth fields in this file describe the starting and ending I/O port addresses. The ninth and tenth fields in this file describe the starting and ending shared memory addresses. The shared memory fields are not used for the *Xcpqag* server for X11. All of these fields are given in hexadecimal. The default *sdevice* file entry with an I/O base address of 0290h is:

```
cpqag  Y    1    0    0    0    290   29f   0    0
```

If you want to use an I/O base address of 0280h instead, then the following *sdevice* entry should be used:

```
cpqag  Y    1    0    0    0    280   28f   0    0
```

If you change the default *sdevice* entry, you will have to build a new kernel and boot it for these changes to become effective.

PROGRAMMING INTERFACE

The COMPAQ AG1024 device is opened by a call to *open*. Currently, only minor device 0 is supported. If a non-zero minor device is specified, then an error will occur and *errno* will be set to ENOENT. A simple check is done to see if the AG1024 board is in the system. If it is not, an error will result and *errno* will be set to ENXIO. An I/O address clash may also cause this error. Only one user is allowed to open the AG1024 board at a time, unless the subsequent user is the superuser. If a subsequent open is attempted by another user who is not the superuser, an error will result and *errno* will be set to EPERM.

Read and write interfaces are provided to access the memory of the Texas Instruments 34010 processor in the AG1024. Seek addresses are specified in bytes. Because addresses on a 34010 processor are given in bits, it is necessary to divide the bit address by 8 to get a byte address. The state of the 34010 is saved before each read and write, and it is restored when the read or write is completed.

The following *ioctl* calls allow a user program to access the AG1024 board directly:

CPQAG_GET_INFO

This *ioctl* returns a structure containing the addresses of the board described in the section "Configuration" above. The structure returned is:

```
typedef struct  cpqag_info  {
  short   cpqag_port;       /* i/o port */
  short   cpqag_endport;    /* i/o port */
  char    *cpqag_sram;      /* physical address */
  char    *cpqag_endsram;/* physical address */
  }  cpqag_info_t;
```

This structure is filled in with information in the seventh through tenth fields of the *sdevice* file. It is stored at the location specified by the argument that is passed.

CPQAG_MAP_BOARD
: This *ioctl* allows the process to do in and out instructions at the I/O ports specified in the *sdevice* file. The argument passed is unused.

CPQAG_UNMAP_BOARD
: This *ioctl* prevents the process from doing in and out instructions at the I/O ports specified in the *sdevice* file. The argument passed is unused.

CPQAG_MAP_SRAM
: This *ioctl* allows the process to have direct access to the shared memory. The argument passed is the address of the memory to map to the AG1024 board. It must be page (4096 byte) aligned and span a length specified by the *sdevice* file (which is also a multiple of 4096 bytes). The memory that had existed at the argument passed will be irretrievably deleted, and the current contents of the shared memory will be placed at those locations. This assumes that the I/O ports describing the shared memory are set up properly. One way of allocating the virtual memory in the user's address space that will be mapped to the shared memory is to call *malloc* requesting (*length* + 4096) bytes. Then, using the address that *malloc* returns, round it up to the next page (4K) boundary and use the result as the argument to this *ioctl*.

CPQAG_UNMAP_SRAM
: This *ioctl* call unmaps the direct access to the shared memory. The map must have been set up with CPQAG_MAP_SRAM by the current process. The argument passed should be the same one given to CPQAG_MAP_SRAM. The memory where the shared memory had been becomes zeros.

FILES

/dev/cpqag

SEE ALSO

Xcpqag(1).
sdevice(4) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual*.

NAME

dcaga – Desktop Computing AGA 1024 driver

DESCRIPTION

The Desktop Computing AGA 1024 kernel driver facilitates accessing the Desktop Computing AGA 1024 board. It is a regular character-based driver that manages one AGA 1024 board. It is used by programs that draw using the graphics modes of AGA 1024.

CONFIGURATION

The *sdevice*(4) file **/etc/conf/sdevice.d/dcaga** describes the memory addresses of the AGA 1024 board. Since the memory address of the AGA 1024 is not configurable, it should not be necessary to change the default *sdevice* file. The ninth and tenth fields in this file describe the starting and ending memory addresses. These fields are given in hexadecimal. The default *sdevice* file entry is:

```
dcaga  Y   1   0   0   0   0   0   cf000  cffff
```

If you change the default *sdevice* entry, you will have to build a new kernel and boot it for these changes to become effective.

PROGRAMMING INTERFACE

The Desktop Computing AGA 1024 device is opened by a call to *open*. Currently, only minor device 0 is supported. If a non-zero minor device is specified, then an error will occur and *errno* will be set to ENOENT. A simple check is done to see if the AGA 1024 board is in the system. If it is not, an error will result and *errno* will be set to ENXIO. A memory address clash may also cause this error. Only one user is allowed to open the AGA 1024 board at a time, unless the subsequent user is the superuser. If a subsequent open is attempted by another user who is not the superuser, an error will result and *errno* will be set to EPERM.

Read and write interfaces are provided to access the memory of the Texas Instruments 34010 processor in the AGA 1024. Seek addresses are specified in bytes. Because addresses on a 34010 processor are given in bits, it is necessary to divide the bit address by 8 to get a byte address. The state of the 34010 is saved before each read and write, and restored when completed.

The *ioctl* calls allow a user program to access the AGA 1024 board directly.

DCAGA_GET_INFO

This *ioctl* returns a structure containing the addresses of the board described in the section "Configuration" above. The structure returned is:

```
typedef struct  dcaga_info  {
  char   *dcaga_addr;     /* physical address */
  char   *dcaga_endaddr; /* physical address */
  }  dcaga_info_t;
```

This structure is filled in with information in the ninth through tenth fields of the *sdevice* file. It is stored at the location specified by the argument that is passed.

DCAGA_MAP_BOARD
This *ioctl* allows the process to have direct access to the AGA 1024 memory. The argument passed is the address of the memory to map to the AGA 1024 board. It must be page (4096 byte) aligned, and it must span a length specified by the *sdevice* file (which is also a multiple of 4096 bytes). The memory that had existed at the argument passed will be irretrievably deleted, and the current contents of the AGA 1024 memory will be placed at those locations. One way of allocating the virtual memory in the user's address space that will be mapped to the AGA 1024 memory is to call *malloc* requesting (*length* + 4096) bytes. Then, using the address that *malloc* returns, round it up to the next page (4K) boundary and use the result as the argument to this *ioctl*.

DCAGA_UNMAP_BOARD
This *ioctl* call unmaps the direct access to the AGA 1024 memory. The map must have been set up with DCAGA_MAP_BOARD by the current process. The argument passed should be the same one given to DCAGA_MAP_BOARD. The memory where the AGA 1024 memory had been becomes zeros.

FILES

/dev/dcaga

SEE ALSO

Xdcaga(1).
sdevice(4) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual*.

NAME

display – system console display

DESCRIPTION

The system console (and user's terminal) is composed of two separate pieces: the keyboard (see *keyboard*(7)) and the display. Because of their complexity, and because there are three possible display interfaces (monochrome, color graphics, and enhanced graphics adapters), they are discussed in separate manual entries.

The display normally consists of 25 lines of 80 columns each; 40-column lines are also supported by the color/graphics adapter, and 43 lines of 80-columns each are supported by the enhanced graphics adapter. Writing characters to the console or one of its virtual screens (**/dev/console** or **/dev/vt***xx*) has an effect which depends on the characters. All characters written to **/dev/console** are first processed by the terminal interface (see *termio*(7)). For example, mapping new-line characters to carriage return plus new-line, and expanding tabs to spaces, will be done before the following processing:

x Where *x* is not one of the following, displays *x*.

BEL Generates a *bell* (audible tone, no modulation).

CR Places the cursor at column 1 of the current line.

LF, VT Places the cursor at the same column of the next line (scrolls if the current line is line 25).

FF Clears the screen and places the cursor at line 1, column 1.

BS Depends on the previous character: if an _ (underscore), see below; otherwise, if the cursor is not at column 1, it is moved to the left one position on the same line. If the cursor is at column 1, it is not moved.

_BS*x* Sets the *underscore* attribute for the character *x* to be displayed. The *underscore* attribute for the color/graphics adapter is a red background with a white foreground.

ESC*x* Where *x* is *any* of the 256 possible codes (except **c**, **[**, and **H**), displays that value uninterpreted. This is useful for utilizing the full set of graphics available on the display. Note again that the characters are processed through the terminal interface prior to this escape sequence. Therefore, to get some of the possible 256 characters, it is necessary that the character not be postprocessed. The easiest way to accomplish this is to turn off OPOST in the *c_oflag* field (see *termio*(7)); however, this may have other side effects.

The display can be controlled by means of ANSI X3.64 *escape sequences*, which are specific sequences of characters, preceded by the ASCII character ESC. The escape sequences, which work on either the monochrome, color graphics, or enhanced graphics adapter, are the following:

ESCc Clears the screen and places the cursor at line 1, column 1.

ESC H — Sets a tab stop at the current cursor position. This escape sequence will be ignored if a tab is already set.

ESC Q *n* '*string*' — Defines the function key *n* with *string*. The string delimiter ' may be any character not in *string*. Function keys are numbered 0 through 11 (F1 = 0, F2 = 1, etc.).

ESC[*n* @ — Insert character—inserts *n* blanks at the current cursor position.

ESC[*n* ` — Horizontal Position Absolute – moves active position to column given by *n*.

ESC[*n* A — Cursor up – moves the cursor up *n* lines (default: *n*=1).

ESC[*n* a — Horizontal Position Relative – moves active position *n* characters to the right (default: *n*=1).

ESC[*n* B — Cursor down – moves the cursor down *n* lines (default: *n*=1).

ESC[*n* C — Cursor right – moves the cursor right *n* columns (default: *n*=1).

ESC[*n* D — Cursor left – moves the cursor left *n* columns (default: *n*=1).

ESC[*n* d — Vertical Position Absolute – moves active position to line given by *n*.

ESC[*n* E — Cursor next line – moves the cursor to column 1 of the next line, then down *n*–1 lines (default: *n*=1).

ESC[*n* e — Vertical Position Relative – moves the active position down *n* lines (default: *n*=1).

ESC[*n* F — Cursor previous line – moves the cursor to column 1 of the current line, then up *n* lines (default: *n*=1).

ESC[*n* G — Cursor horizontal position – moves the cursor to column *n* of the current line (default: *n*=1).

ESC[*n* ; *m* H — Position cursor – moves the cursor to column *m* of line *n* (default: *n*=1, *m*=1).

ESC[*n* ; *m* f — Position cursor – moves the cursor to column *m* of line *n* (default: *n*=1, *m*=1).

ESC[*n* J — Erase window – erases from the current cursor position to the end of the window if *n*=0, from the beginning of the window to the current cursor position if *n*=1, and the entire window if *n*=2 (default: *n*=0).

ESC[*n* K — Erase line – erases from the current cursor position to the end of the line if *n*=0, from the beginning of the line to the current cursor position if *n*=1, and the entire line if *n*=2 (default: *n*=0).

ESC[*n* L — Insert line – inserts *n* lines at the current cursor position (default: *n*=1).

ESC[*n* M — Delete line – deletes *n* lines starting at the current cursor position (default: *n*=1).

ESC[*n* P — Delete character – deletes *n* characters from a line starting at the current cursor position (default: *n*=1).

ESC[*n* S — Scroll up – scrolls the characters in the current window up *n* lines. The bottom *n* lines are cleared to blanks (default: *n*=1).

ESC[*n* T — Scroll down – scrolls the characters in the current window down *n* lines. The top *n* lines are cleared to blanks (default: *n*=1).

ESC[*n* X — Erase character – erases *n* character positions starting at the current cursor position (default: *n*=1).

ESC[*n* Z — Cursor Backward Tabulation – moves active position back *n* tab stops.

ESC[2 h — Locks the keyboard and ignores keyboard input until unlocked. Characters are not saved.

ESC[2 i — Sends the screen to the host. The current screen display is sent to the application.

ESC[2 l — Unlocks the keyboard. Re-enables keyboard input.

ESC[? 7 h — Disables the no-wrap flag, causing characters typed after column 80 to continue on the next line.

ESC[? 7 l — Sets the no-wrap flag, causing characters typed after column 80 to remain at column 80.

ESC[*n* g — Delete tab stop – removes the tab stop at the current cursor position if *n*=0, removes all tab stops if *n*=3 (default: *n*=0).

ESC[*Ps* ; *Ps*; ... m

Character attributes – each *Ps* is one of the following characters; multiple characters are separated by semicolons. These parameters apply to successive characters being displayed, in an additive manner (e.g., both bold and underscoring can be selected). Only the parameters through 7 apply to the monochrome adapter; all parameters apply to the color/graphics adapter and the enhanced graphics adapter (default: *Ps*=0).

Ps	*Meaning*		
0	all attributes off (normal display) (white foreground with black background)		
1	bold intensity		
4	underscore on (white foreground with red background on color)		
5	blink on		
7	reverse video		
30	black	(gray)	foreground
31	red	(light red)	foreground
32	green	(light green)	foreground
33	brown	(yellow)	foreground
34	blue	(light blue)	foreground
35	magenta	(light magenta)	foreground
36	cyan	(light cyan)	foreground
37	white	(bright white)	foreground
40	black	(gray)	background
41	red	(light red)	background
42	green	(light green)	background
43	brown	(yellow)	background
44	blue	(light blue)	background
45	magenta	(light magenta)	background
46	cyan	(light cyan)	background
47	white	(bright white)	background
ESC[3 *Cm*	Set foreground color *C* where *C* represents the color defined in Table 1 below.		
ESC[4 *Cm*	Set backgound color *C* where *C* represents the color defined in Table 1 below.		
ESC[8 *m*	sets blank (non-display)		
ESC[10 *m*	selects the primary font		
ESC[11 m	selects the first alternate font; lets ASCII characters less than 32 be displayed as ROM characters		
ESC[12 *m*	selects a second alternate font; toggles high bit of extended ASCII code before displaying as ROM characters		
ESC[=*cF*	Set normal foreground color to *c* where *c* represents a color defined in Table 2 below.		
ESC[=*cG*	Set normal background color to *c* where *c* represents a color defined in Table 1 below.		
ESC[=*cH*	Set reverse foreground color to *c* where *c* represents a color defined in Table 2 below.		
ESC[=*cI*	Set reverse background color to *c* where *c* represents a color defined in Table 2 below.		
ESC[=*cJ*	Set graphic foreground color to *c* where *c* represents a color defined in Table 2 below.		
ESC[=*cK*	Set graphic background color to *c* where *c* represents a color defined in Table 2 below.		
ESC[=*cA*	Set the overscan (border) color to *c* where *c* represents a color defined in Table 2 below.		

TABLE 1	
C	*Color*
0	Black
1	Red
2	Green
3	Yellow
4	Blue
5	Magenta
6	Cyan
7	White

TABLE 2			
C	*Color*	*C*	*Color*
0	Black	8	Grey
1	Blue	9	Lt. Blue
2	Green	10	Lt. Green
3	Cyan	11	Lt. Cyan
4	Red	12	Lt. Red
5	Magenta	13	Lt. Magenta
6	Brown	14	Yellow
7	White	15	Lt. White

Note that for character attributes 30-37, the color selected for foreground will depend on whether the *bold intensity* attribute (1) is currently on. If not, the first color listed will result; otherwise the second color listed will result.

Similarly, for character attributes 40-47, the color selected for background will depend on whether the *blink* attribute (5) is currently on. If the *blink* attribute is not on, then the first color listed will result. If the *blink* attribute is on, then the second color listed will result.

Ioctl Calls

The following *ioctl* calls can be used to change the display used for the video monitor:

SWAPMONO
: This call selects the monochrome adapter as the output device for the system console.

SWAPCGA
: This call selects the color/graphics adapter as the output device for the system console.

SWAPEGA
: This call selects the enhanced graphics adapter as the output device for the system console.

SWAPVGA
: This call selects the video graphics array adapter as the output device for the system console.

The following *ioctl* call may be used to obtain more information about the display adapter currently attached to the video monitor:

CONS_CURRENT
: This call returns the display adapter type currently attached to the video monitor. The return value can be one of: MONO, CGA, EGA, or VGA.

The following *ioctl* calls may be used to switch display modes on the various video adapters:

SW_B40x25
This call selects 40x25 (40 columns by 25 rows) black and white text display mode. It is valid only for CGA, EGA, and VGA devices.

SW_C40x25
This call selects 40x25 (40 columns by 25 rows) color text display mode. It is valid only for CGA, EGA, and VGA devices.

SW_B80x25
This call selects 80x25 (80 columns by 25 rows) black and white text display mode. It is valid only for CGA, EGA, and VGA devices.

SW_C80x25
This call selects 80x25 (80 columns by 25 rows) color text display mode. It is valid only for CGA, EGA, and VGA devices.

SW_BG320
This call selects 320x200 black and white graphics display mode. It is valid only for CGA, EGA, and VGA devices.

SW_CG320
This call selects 320x200 color graphics display mode. It is valid only for CGA, EGA, and VGA devices.

SW_BG640
This call selects 640x200 black and white graphics display mode. It is valid only for CGA, EGA, and VGA devices.

SW_CG320_D
This call selects EGA support for 320x200 graphics display mode (EGA mode D). It is valid only for EGA and VGA devices.

SW_CG640_E
This call selects EGA support for 640x200 graphics display mode (EGA mode E). It is valid only for EGA and VGA devices.

SW_EGAMONOAPA
This call selects EGA support for 640x350 graphics display mode (EGA mode F). It is valid only for EGA and VGA devices.

SW_ENH_MONOAPA2
This call selects EGA support for 640x350 graphics display mode with extended memory (EGA mode F*). It is valid only for EGA and VGA devices.

SW_CG640x350
This call selects EGA support for 640x350 graphics display mode (EGA mode 10). It is valid only for EGA and VGA devices.

SW_ENH_CG640
: This call selects EGA support for 640x350 graphics display mode with extended memory (EGA mode 10*). It is valid only for EGA and VGA devices.

SW_EGAMONO80x25
: This call selects EGA monochrome text display mode (EGA mode 7), which emulates support provided by the monochrome adapter. It is valid only for EGA and VGA devices.

SW_ENHB40x25
: This call selects enhanced 40x25 black and white text display mode. It is valid only for EGA and VGA devices.

SW_ENHC40x25
: This call selects enhanced 40x25 color text display mode. It is valid only for EGA and VGA devices.

SW_ENHB80x25
: This call selects enhanced 80x25 black and white display mode. It is valid only for EGA and VGA devices.

SW_ENHC80x25
: This call selects enhanced 80x25 color text display mode. It is valid only for EGA and VGA devices.

SW_ENHB80x43
: This call selects enhanced 80x43 black and white text display mode. It is valid only for EGA and VGA devices.

SW_ENHC80x43
: This call selects enhanced 80x43 color text display mode. It is valid only for EGA and VGA devices.

SW_MCAMODE
: This call reinitializes the monochrome adapter. It is valid only for monochrome adapters.

SW_VGA_B132x25
: This call selects enhanced 132x25 black and white text display mode. It is valid only on Sigma VGA/H adapters with 132 column support.

SW_VGA_C132x25
: This call selects enhanced 132x25 color text display mode. It is valid only on Sigma VGA/H adapters with 132 column support.

SW_VGA_B132x43
: This call selects enhanced 132x43 black and white text display mode. It is valid only on Sigma VGA/H adapters with 132 column support.

SW_VGA_C132x43
: This call selects enhanced 132x43 color text display mode. It is valid only on Sigma VGA/H adapters with 132 column support.

SW_VGAMONOAPA
: This call selects VGA support for 640x350 monochrome graphics display support (VGA mode F+). It is valid only for VGA devices.

SW_VGA_CG640
: This call selects VGA support for 640x350 color graphics display support (VGA mode 10+). It is valid only for VGA devices.

SW_VGA_B40x25
: This call selects VGA support for VGA 40x25 (40 columns by 25 rows) black and white text display mode. It is valid only for VGA devices.

SW_VGA_C40x25
: This call selects VGA support for VGA 40x25 (40 columns by 25 rows) color text display mode. It is valid only for VGA devices.

SW_VGAB80x25
: This call selects VGA support for VGA 80x25 (80 columns by 25 rows) black and white text display mode. It is valid only for VGA devices.

SW_VGAC80x25
: This call selects VGA support for VGA 80x25 (80 columns by 25 rows) color text display mode. It is valid only for VGA devices.

SW_VGAMONO80x25
: This call selects VGA monochrome text display mode (VGA mode 7+). It is valid only for VGA devices.

SW_BG640x480
: This call selects VGA 640x480 black and white graphics display mode (VGA mode 11). It is valid only for VGA devices.

SW_CG640x480
: This call selects VGA 640x480 color graphics display mode (VGA mode 12). It is valid only for VGA devices.

SW_VGA_CG320
: This call selects VGA 320x200 color graphics display mode (VGA mode 13). It is valid only for VGA devices.

Switching to an invalid display mode for a display device will result in an error.

The following *ioctl*s may be used to obtain information about the current display modes:

CONS_GET
: This call returns the current display mode setting for whatever display adapter is being used. Possible return values include:

  DM_B40x25 (0), black and white 40 columns. CGA and EGA only.

DM_C40x25 (1), color 40 columns. CGA and EGA only.

DM_B80x25 (2), black and white 80 columns. CGA and EGA only.

DM_C80x25 (3), color 80 columns. CGA and EGA only.

DM_BG320 (4), black and white graphics 320x200. CGA and EGA only.

DM_CG320 (5), color graphics 320x200. CGA and EGA only.

DM_BG640 (6), black and white graphics 640x200 high-resolution. CGA and EGA only.

DM_EGAMONO80x25 (7), EGA-mono 80x25. EGA only.

DM_ENH_B80x43 (10), EGA enhanced black and white 80x43.

DM_ENH_C80x43 (11), EGA enhanced color 80x43.

DM_CG320_D (13), EGA mode D.

DM_CG640_E (14), EGA mode E.

DM_EGAMONOAPA (15), EGA mode F.

DM_CG640x350 (16), EGA mode 10.

DM_ENHMONOAPA2 (17), EGA mode F with extended memory.

DM_ENH_CG640 (18), EGA mode 10*.

DM_ENH_B40x25 (19), EGA enhanced black and white 40 columns.

DM_ENH_C40x25 (20), EGA enhanced color 40 columns.

DM_ENH_B80x25 (21), EGA enhanced black and white 80 columns.

DM_ENH_C80x25 (22), EGA enhanced color 80 columns.

DM_VGA_C40x25 (23), VGA color 40x25.

DM_VGA_C80x25 (24), VGA color 80x25.

DM_VGAMONO80x25 (25), VGA mode 7+.

DM_BG640x480 (26), VGA black and white graphics 640x480 (VGA mode 11).

DM_CG640x480 (27), VGA color graphics 640x480 (VGA mode 12).

DM_VGA_CG320 (28), VGA color graphics 320x200 (VGA mode 13).

DM_VGA_B40x25 (29), VGA black and white 40x25.

DM_VGA_B80x25 (30), VGA black and white 80x25.

DM_VGAMONOAPA (31), VGA mode F+.

DM_VGA_CG640 (32), VGA mode 10+.

DM_VGA_B132x25 (35), VGA enhanced black and white 132x25.

DM_VGA_C132x25 (36), VGA enhanced color 132x25.

DM_VGA_B132x43 (37), VGA enhanced black and white 132x43.

DM_VGA_C132x43 (38), VGA enhanced color 132x43.

M_MCA_MODE (0xff), monochrome adapter mode.

MCA_GET
This call returns the current display mode setting of the monochrome adapter. See CONS_GET for a list of return values. If the monochrome adapter is not installed, the call will fail and *erno* will be set to 22 (EINVAL).

CGA_GET
This call returns the current display mode setting of the color/graphics adapter. See CONS_GET for a list of return values. If the color graphics adapter is not installed, the call will fail and *erno* will be set to 22 (EINVAL).

EGA_GET
This call returns the current display mode setting of the enhanced graphics adapter. See CONS_GET for a list of return values. If the enhanced graphics adapter is not installed, the call will fail and *erno* will be set to 22 (EINVAL).

VGA_GET
This call returns the current display mode setting of the video graphics array adapter. See CONS_GET for a list of return values. If the video graphics array adapter is not installed, the call will fail and *erno* will be set to 22 (EINVAL).

The following *ioctl* calls may be used to map the video adapter's memory into the user's data space:

MAPCONS
This call maps the display memory of the adapter currently being used into the user's data space.

MAPMONO
This call maps the monochrome adapter's display memory into the user's data space.

MAPCGA
This call maps the color/graphics adapter's display memory into the user's data space.

MAPEGA
This call maps the enhanced graphics adapter's display memory into the user's data space.

MAPVGA
This call maps the video graphics array adapter's display memory into the user's data space.

You can use *ioctl* calls to input a byte from the graphics adapter port or to output a byte to the graphics adapter port. The argument to the *ioctl* uses the **port_io_arg** data structure:

```
struct port_io_arg {
        struct port_io_struc_args[4];
};
```

As shown in the previous example, the **port_io_arg** structure points to an array of four **port_io_struc** data structures. The **port_io_struc** has the following format:

```
struc port_io_struc {
        char dir;                /* direction flag (in vs. out) */
        unsigned short port;     /* port address */
        char data;               /* byte of data */
};
```

You can specify one, two, three, or four of the **port_io_struc** structures in the array for one *ioctl* call. The value of *dir* can be either IN_ON_PORT (to specify a byte being input from the graphics adapter port) or OUT_ON_PORT (to specify a byte being output to the graphics adapter port). *Port* is an integer specifying the port address of the desired graphics adapter port. *Data* is the byte of data being input or output as specified by the call. If you are not using any of the **port_io_struc** structures, load the *port* with 0, and leave the unused structures at the end of the array. Refer to your hardware manuals for port addresses and functions for the various adapters.

The following *ioctl* calls may be used to input or output bytes on the graphics adapter port:

MCAIO This call inputs or outputs a byte on the monochrome adapter port as specified.

CGAIO This call inputs or outputs a byte on the color/graphics adapter port as specified.

EGAIO This call inputs or outputs a byte on the enhanced graphics adapter port as specified.

VGAIO This call inputs or outputs a byte on the video graphics array adapter port as specified.

To input a byte on any of the graphics adapter ports, load *dir* with IN_ON_PORT and load *port* with the port address of the graphics adapter. The byte input from the graphics adapter port will be returned in *data.*

To output a byte, load *dir* with OUT_ON_PORT, load *port* with the port address of the graphics adapter, and load *data* with the byte you want to output to the graphics adapter port.

The following *ioctl*s can be used with either the monochrome, color graphics, or enhanced graphics adapters:

KDDISPTYPE
This call returns display information to the user. The argument expected is the buffer address of a structure of type **kd_disparam** into which display information is returned to the user. The **kd_disparam** structure is defined as follows:

```
struct kd_disparam {
   long type;                    /* display type */
   char *addr;                   /* display memory address */
   ushort ioaddr[MKDIOADDR];     /* valid I/O addresses */
}
```

Possible values for the type field include:

KD_MONO (0x01)	IBM monochrome display adapter
KD_HERCULES (0x02)	Hercules monochrome graphics adapter
KD_CGA (0x03)	IBM color graphics adapter
KD_EGA (0x04)	IBM enhanced graphics adapter
KD_VGA (0x05)	VGA adapter

The *addr* member indicates the physical address of the display. It will be one of:

MONO_BASE	0xb0000
COLOR_BASE	0xb8000
EGA_BASE	0xa0000

The *ioaddr* member contains the global keyboard/display port list that is used for direct *in*s and *out*s to the screen. When using a VGA display, the result of a KDISPTYPE *ioctl* is the same as with an EGA display.

KIOCSOUND

This call starts the sound generation. It turns on sound. The argument is the inverse frequency desired. A value of 0 turns off the sound.

KDGETLED

This call gets the keyboard LED status. The argument is a pointer to a character. The character will be filled with a boolean combination of the following values:

LED_SCR	0x01(flag bit for scroll lock)
LED_CAP	0x04(flag bit for caps lock)
LED_NUM	0x02(flag bit for num lock)

KDSETLED

This call sets the keyboard LED status. The argument is a character whose value is the boolean combination of the values listed under KDGETLED.

KDMKTONE

This call generates a fixed length tone. The argument is a 32-bit value, with the lower 16 bits set to the inverse frequency and the upper 16 bits set to the duration (in milliseconds).

KDGKBTYPE

This call gets keyboard type. The argument is a pointer to a character type. The character will be returned with one of the following values:

KB_84	0x01(84-key keyboard)
KB_101	0x02(101-key keyboard)
KB_OTHER	0x03

KDADDIO
: This call adds the I/O port address to the list of valid video adapter addresses. Argument is an unsigned short type which should contain a valid port address for the installed video adapter.

KDDELIO
: This call deletes the I/O port address from the list of valid video adapter addresses. Argument is an unsigned short type which should contain a valid port address for the installed video adapter.

KDENABIO
: This call enables *in*s and *out*s to video adapter ports. No argument.

KDDISABIO
: This call disables *in*s and *out*s to video adapter ports. No argument.

KDSBORDER
: This call sets the screen color border in EGA text mode. The argument is of type character. Each bit position corresponds to a color selection. From bit position 0 to bit position 6, the color selections are respectively blue, green, red, secondary blue, secondary green, and secondary red. Setting the bit position to a logic one will select the desired color or colors.

KDSCROLL
: This call is used to set the hardware scrolling feature to be on or off. Most CGA, EGA, and VGA displays provide hardware-assisted scrolling for better output performance. The default is hardware scrolling turned off. A nonzero argument enables scrolling; a zero argument disables it.

KDSETMODE
: This call sets the mode of the display driver to the integer argument given. The modes that may be specified are:

KD_TEXT0	0x00
KD_GRAPHICS	0x01
KD_TEXT1	0x02

KD_TEXT is a synonym for KD_TEXT0.

KD_TEXT0 indicates that all text on the display must be written with the *write* system call. The display will automatically be saved and restored on "hot key" screen switches. Upon leaving KD_TEXT0 mode, the display contents will be saved; upon returning to KD_TEXT0 mode, the display will be restored from the saved display. KD_TEXT1 works in the same manner as KD_TEXT0, except that the display is not restored when the user returns to KD_TEXT1 mode.

KD_GRAPHICS mode indicates that the user will have direct control of the display. It will be necessary to map in the display using the KDMAPDISP *ioctl* subsequent to setting KD_GRAPHICS mode to store characters directly onto the display. In this mode, all writes to the display using the *write* system call are ignored. In KD_GRAPHICS mode, the user is responsible for saving and restoring the display on "hot key" switches. This requires that the virtual terminal must be in process (VT_PROCESS) mode prior to setting KD_GRAPHICS mode.

In KD_GRAPHICS mode, there will not necessarily be graphics on the display. When setting KD_TEXT0 or KD_TEXT1 mode, the *ioctl* will fail if the display is mapped in [EIO]. When setting KD_GRAPHICS mode, the *ioctl* will fail if the virtual terminal is not in process (VT_PROCESS) mode [EACCES].

KDGETMODE

This call gets the current mode of the console. It returns an integer argument containing KD_TEXT, KD_TEXT1, or KD_GRAPHICS as defined in the KDSETMODE *ioctl* description.

KDMAPDISP

This call allows one to have direct access to the display and I/O ports. It is especially useful in providing a way to do non-portable but fast graphics on the display. The following structure, defined in **<sys/kd.h>**, is pointed to by the argument to the *ioctl*:

```
struct kd_memloc {
   char     *vaddr;     /* virtual address to map to*/
   char     *physaddr;  /* physical address to map from */
   long     length;     /* size in bytes to map */
   long     ioflg;      /* enable I/O addresses if non-zero */
};
```

The *vaddr* argument is the linear address in the process where the display buffer will appear. This address must be on a page (4K byte) boundary. The *physaddr* argument is the physical address of the screen. It must be between 0xA0000 and 0xC0000. It must also be on a page boundary. The *length* argument is the size of the display buffer that will be mapped in. It must be a multiple of 4K bytes. The *ioflg* argument tells whether (1) or not (0) to enable the global keyboard/display ports for direct access to the I/O ports similar to the KDENABIO and KDDISABIO *ioctl*.

The memory that had existed at address *vaddr* for *length* bytes will be irretrievably deleted, and the current contents of the display buffer will be placed at those locations. It is necessary for the virtual terminal to be in process (VT_PROCESS) mode and for the display device to be in KD_GRAPHICS mode. One way of allocating the virtual memory in the user's

address space that will be mapped to the screen is to call *malloc* requesting (*length* + 4096) bytes. Then using the address that *malloc* returns, round it up to the next page (4K byte) boundary and use the result as *vaddr*.

The *ioctl* will fail if the virtual terminal is not in process mode or if the display is not in KD_GRAPHICS mode [EACCES]. It will fail if any of the arguments are out of range or not properly aligned [EFAULT]. It will fail if the display is already mapped to [EIO].

KDUNMAPDISP

This call unmaps the direct access to the display and disables the direct usage of the I/O ports. The map must have been set up with KDMAPDISP by the current process. The memory where the display had been becomes zeroes. The *ioctl* will fail if the current process is not the one that did the mapping [EACCES].

VT_OPENQRY

This call finds an available virtual terminal. The argument is a pointer to a long. The long will be filled with the number of the first available "VT" that no other process has open, or −1 if none are available.

VT_GETMODE

This call determines what mode the active virtual terminal is currently in, either VT_AUTO or VT_PROCESS. The argument to the *ioctl* is the address of the following type of structure:

```
struct vt_mode {
  char    mode;     /* VT mode */
  char    waitv;    /* if set, hang on writes when not active */
  short   relsig;   /* signal to use for release request */
  short   acqsig;   /* signal to use for display acquired */
  short   frsig;    /* signal to use for forced release */
  }

#define VT_AUTO     0x00   /* automatic VT switching */
#define VT_PROCESS  0x01   /* process controls switching */
```

The **vt_mode** structure will be filled in with the current value for each field.

VT_SETMODE

Set the virtual terminal mode. The argument is a pointer to a **vt_mode** structure, as defined above.

VT_RELDISP

This call is used to tell the virtual terminal manager that the display has or has not been released by the process. A non-zero argument indicates that the display has been released; a zero argument indicates refusal to release the display.

VT_ACTIVATE
This call makes the virtual terminal number specified in the argument the active VT. The VT manager will cause a switch to occur in the same manner as if a hotkey sequence had been typed at the keyboard. If the specified VT is not open or does not exist, the call will fail and *errno* will be set to ENXIO.

KIOCINFO
This call tells the user what the device is.

GIO_SCRNMAP
This call gets the screen mapping table from the kernel.

GIO_ATTR
This call returns the current screen attribute. The bits are interpreted as follows:

Bit 0 determines underlining for black and white monitors (1=underlining on).

Bits 0-2, for color monitors only, select the foreground color. The following list indicates what colors are selected by the given value:

The value 0 selects black.
The value 1 selects red.
The value 2 selects green.
The value 4 selects blue.
The value 5 selects magenta.
The value 6 selects cyan.
The value 7 selects white.

Bit 3 is the intensity bit (1=blink on).

Bits 4-6, for color monitors only, select the background color. For a list of colors and their values, see the list under foreground colors.

Bit 7 is the blink bit (1=blink on).

GIO_COLOR
This call returns zero if the current display is a color display; otherwise, it returns a non-zero value.

PIO_SCRNMAP
This call puts the screen mapping table in the kernel.

The screen mapping table maps extended ASCII (8-bit) characters to ROM characters. It is an array [256] of char (typedef **scrnmap_t**) and is indexed by extended ASCII values. The value of the elements of the array are the ROM character to display.

For example, the following program will cause the ASCII character # to be displayed as an English pound sign:

```
#include <sys/types.h>
#include <sys/at_ansi.h>
#include <sys/kd.h>
main() {
scrnmap_t scrntab;
        /* get screen mapping table of standard output */
if (ioctl(0,GIO_SCRNMAP, scrntab)==-1)
        {
        perror("screenmap read");
        exit(-1);
        }
/* 156 is the ROM value of English pound sign and 30
   is the ASCII value of '#'.
*/
scrntab[30] = 156;
if (ioctl(0, PIO_SCRNMAP, scrntab) == -1)
        {
        perror("screenmap write");
        exit(-1);
        }
}
```

FILES

/dev/console
/dev/vt00-*n*
/usr/include/sys/kd.h

SEE ALSO

stty(1), console(7), keyboard(7), termio(7) in the *INTERACTIVE UNIX System User's/System Administrator's Reference Manual*.
ioctl(2) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual*.

WARNINGS

Although it is possible to write character sequences which set arbitrary bits on the screen in any of the three graphics modes, this mode of operation is not currently supported.

It is currently not possible to access the 6845 start address registers. Thus, it is impossible to determine the beginning of the color monitor's screen memory.

The alternate/background color bit (bit 4) of the color select register does not appear to affect background colors in alphanumeric modes.

The low-resolution graphics mode appears to be 80 across by 100 down.

NAME

dma – Generic DMA driver

SYNOPSIS

```
dmacheck(chan, paddr, count)
int chan;
paddr_t paddr;
int count;

dmareq(chan, paddr, count, rflag)
int chan;
paddr_t paddr;
int count;
char rflag;

dmadisable(chan)
int chan;
```

DESCRIPTION

Chan is the channel number that is to be used. *Paddr* is the **physical** address of the user buffer that is to be transferred or checked. *Count* is the number of bytes in the user buffer at the specified physical address that is to be transferred. *Rflag* is TRUE if the transfer is from the device to memory, and is FALSE if the transfer is from memory to the device.

The DMA driver is a UNIX System device driver that programs the Intel 8237A-5 chips based on a specified channel.

Dmacheck checks the alignment of the buffer and the count based on the channel number specified. This routine is usually called before the transfer is initiated to check the validity of the user buffer.

Dmareq initiates a DMA transfer. This routine is called to actually transfer to or from a device that supports DMA. This routine is usually called at interrupt time and hence the buffer has to be in memory, that is, it cannot cause a page fault. The interrupt routine of a driver usually breaks up a user buffer into smaller buffers aligned on page boundaries and calls this routine to transfer these smaller buffers. The buffer has to be checked for alignment using *dmacheck* before this routine is called.

Dmadisable disables the specified channel. This routine is called by the driver after all transfers of the user buffer have occurred.

FILES

None.

NAME

igsp – IMAgraph TI1210 driver

DESCRIPTION

The IMAgraph TI1210 kernel driver facilitates accessing the IMAgraph TI1210 board. It is a regular character-based driver that manages one TI1210 board. It is used by programs that draw using the graphics modes of TI1210.

CONFIGURATION

The *sdevice*(4) file **/etc/conf/sdevice.d/igsp** describes the board address, the I/O port addresses, and the shared video memory (VRAM). There are two entries in this file. The second field in both entries tells whether or not the *igsp* is configured. Both should be the same value, either **Y** or **N**. The ninth and tenth fields in the first entry in this file describe the starting and ending memory addresses. The seventh and eighth fields in the first entry describe the I/O port address including the bank select register. The ninth and tenth fields in the second entry in this file describe the starting and ending memory address of the shared VRAM addresses. The I/O ports and the VRAM fields are not used for the *Xigsp* server for X11. All of these fields are given in hexadecimal. The default *sdevice* file entries with memory addresses of C700 hex are:

```
igsp   Y  1  0  0  0  2b0  2bf  c7000  c7fff
igsp   Y  1  0  0  0  0    0    d0000  d3fff
```

If you want to use a memory address of B700 hex instead, then the following *sdevice* entries should be used:

```
igsp   Y  1  0  0  0  2b0  2bf  b7000  b7fff
igsp   Y  1  0  0  0  0    0    d0000  d3fff
```

If you want to use a memory address of D700 hex instead, then the following *sdevice* entries should be used:

```
igsp   Y  1  0  0  0  2b0  2bf  d7000  d7fff
igsp   Y  1  0  0  0  0    0    d0000  d3fff
```

If you change the default *sdevice* entry, you will have to build a new kernel and boot it for these changes to become effective.

Be sure to configure both sub-devices of the *igsp* driver when adding the *igsp* device.

PROGRAMMING INTERFACE

The IMAgraph TI1210 device is opened by a call to *open*. Currently, only minor device 0 is supported. If a non-zero minor device is specified, then an error will occur and *errno* will be set to ENOENT. A simple check is done to see if the TI1210 board is in the system. If it is not, an error will result and *errno* will be set to ENXIO. A memory address clash can also cause this error. Only one user is allowed to open the TI1210 board at a time, unless the subsequent user is the superuser. If a subsequent open is attempted by another user who is not the superuser, an error will result and *errno* will be set to EPERM.

Read and write interfaces are provided to access the memory of the Texas Instruments 34010 processor in the TI1210. Seek addresses are specified in bytes. Because addresses on a 34010 processor are given in

bits, it is necessary to divide the bit address by 8 to get a byte address. The state of the 34010 is saved before each read and write, and it is restored when they are completed.

The following *ioctl* calls allow a user program to access the TI1210 board directly:

IGSP_GET_INFO

This *ioctl* returns a structure containing the addresses of the board described in the section "Configuration" above. The structure returned is:

```
typedef struct  igsp_info  {
  char    *igsp_addr;      /* physical address */
  char    *igsp_endaddr;   /* physical address */
  short   igsp_port;       /* i/o port */
  short   igsp_endport;    /* i/o port */
  char    *igsp_vram;      /* physical address */
  char    *igsp_endvram;   /* physical address */
  }  igsp_info_t;
```

This structure is filled in with information in the seventh through tenth fields of the *sdevice* file. It is stored at the location specified by the argument that is passed.

IGSP_MAP_BOARD

This *ioctl* allows the process to have direct access to the TI1210 memory. The argument passed is the address of the memory to map to the TI1210 board. It must be page (4096 byte) aligned and span a length specified by the *sdevice* file (which is also a multiple of 4096 bytes). The memory that had existed at the argument passed will be irretrievably deleted, and the current contents of the TI1210 memory will be placed at those locations. One way of allocating the virtual memory in the user's address space that will be mapped to the TI1210 memory is to call *malloc* requesting (*length* + 4096) bytes. Then, using the address that *malloc* returns, round it up to the next page (4K) boundary and use the result as the argument to this *ioctl*.

IGSP_UNMAP_BOARD

This *ioctl* call unmaps the direct access to the TI1210 memory. The map must have been set up with IGSP_MAP_BOARD by the current process. The argument passed should be the same one given to IGSP_MAP_BOARD. The memory where the TI1210 memory had been becomes zeros.

IGSP_MAP_VRAM

This *ioctl* allows the process to do in and out instructions at the I/O ports specified in the *sdevice* file as well as mapping an area of memory for the shared VRAM. The argument passed is the address of memory to map to the VRAM. The VRAM is mapped in the same way as the board is, with the same restrictions.

IGSP_UNMAP_VRAM
This *ioctl* prevents the process from doing in and out instructions at the I/O ports specified in the *sdevice* file as well as unmapping the VRAM. The argument passed should be the same one given to IGSP_MAP_VRAM.

FILES

/dev/igsp

SEE ALSO

Xigsp(1).
sdevice(4) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual*.

NAME

kdmouse – built-in mouse device interface

DESCRIPTION

The *kdmouse* driver supports Micro Channel architecture mice and compatibles (such as the IBM PS/2 mouse) on machines with built-in mouse interfaces such as the COMPAQ 20e and the IBM PS/2 model 80. It allows applications to obtain information about the mouse's movements and the status of its buttons.

Programs are able to read directly from the device. The data returned corresponds to the byte sequences as defined in the *IBM PS/2 Technical Reference Manual*. Programs are not able to write to the *kdmouse* device.

Ioctl Call

MOUSEIOCREAD

This call returns status information about the mouse buttons and the mouse's relative movement since the previous MOUSEIOCREAD (or since mouse initialization, for the first MOUSEIOCREAD). The argument is the address of a *mouseinfo* structure; the following structure definition and **#defines** may be found in **<sys/mouse.h>**:

```
struct mouseinfo {
 unsigned char status;          /* see definitions below*/
 char xmotion, ymotion;         /* between -128 and 127*/
}
 #define BUT3STAT   1          /* button 3 status (1=down) */
 #define BUT2STAT   2          /* button 2 status (1=down) */
 #define BUT1STAT   4          /* button 1 status (1=down) */
 #define BUT3CHNG   8          /* button 3 changed? (1=yes) */
 #define BUT2CHNG   0x10       /* button 2 changed? (1=yes) */
 #define BUT1CHNG   0x20       /* button 1 changed? (1=yes) */
 #define MOVEMENT   0x40       /* mouse moved? (1=yes) */
```

FILES

/dev/kdmouse

NAME

logi – LOGITECH Bus Mouse device interface

DESCRIPTION

The *mouse* driver supports the LOGITECH Bus Mouse on the AT release. It allows applications to obtain information about the mouse's movements and the status of its buttons.

Ioctl Call

MOUSEIOCREAD

This call returns status information about the mouse buttons and the mouse's relative movement since the previous MOUSEIOCREAD (or since mouse initialization, for the first MOUSEIOCREAD). The argument is the address of a *mouseinfo* structure; the following structure definition and **#defines** may be found in **<sys/mouse.h>**:

```
struct mouseinfo {
 unsigned char status;          /* see definitions below*/
 char xmotion, ymotion;         /* between -128 and 127*/
}
 #define BUT3STAT   1          /* button 3 status (1=down) */
 #define BUT2STAT   2          /* button 2 status (1=down) */
 #define BUT1STAT   4          /* button 1 status (1=down) */
 #define BUT3CHNG   8          /* button 3 changed? (1=yes) */
 #define BUT2CHNG   0x10       /* button 2 changed? (1=yes) */
 #define BUT1CHNG   0x20       /* button 1 changed? (1=yes) */
 #define MOVEMENT   0x40       /* mouse moved? (1=yes) */
```

FILES

/dev/logi

NAME

mouse – mouse device interface

DESCRIPTION

The *mouse* driver supports the Microsoft Inport Bus Mouse on the AT release. It allows applications to obtain information about the mouse's movements and the status of its buttons.

Ioctl Call

MOUSEIOCREAD

This call returns status information about the mouse buttons and the mouse's relative movement since the previous MOUSEIOCREAD (or since mouse initialization, for the first MOUSEIOCREAD). The argument is the address of a *mouseinfo* structure; the following structure definition and **#defines** may be found in **<sys/mouse.h>**:

```
struct mouseinfo {
 unsigned char status;          /* see definitions below*/
 char xmotion, ymotion;         /* between -128 and 127*/
}
 #define BUT3STAT    1         /* button 3 status (1=down) */
 #define BUT2STAT    2         /* button 2 status (1=down) */
 #define BUT1STAT    4         /* button 1 status (1=down) */
 #define BUT3CHNG    8         /* button 3 changed? (1=yes) */
 #define BUT2CHNG    0x10      /* button 2 changed? (1=yes) */
 #define BUT1CHNG    0x20      /* button 1 changed? (1=yes) */
 #define MOVEMENT    0x40      /* mouse moved? (1=yes) */
```

FILES

/dev/mouse

NAME

msfdp – MegaScan FDP-6120 driver

DESCRIPTION

The MegaScan FDP-6120 kernel driver facilitates accessing the MegaScan FDP-6120 board. It is a regular character-based driver that manages one FDP-6120 board. It is used by programs that draw using the graphics modes of FDP-6120.

CONFIGURATION

The *sdevice*(4) file **/etc/conf/sdevice.d/msfdp** describes the I/O port addresses and the memory addresses of the FDP-6120 board. The seventh and eighth fields in this file describe the starting and ending input/output port addresses. The ninth and tenth fields in this file describe the starting and ending memory addresses. All of these fields are given in hexadecimal. The default *sdevice* file entry is:

```
msfdp Y 1 0 0 0 300 30f a0000 a7fff
```

As an example, if you want to change the default *sdevice* to use addresses of C00000- FFFFFF and I/O ports of 200-20f, then the following *sdevice* file is used:

```
msfdp Y 1 0 0 0 200 20f c00000 ffffff
```

If you change the default *sdevice* entry, you will have to build a new kernel and boot it for these changes to become effective.

PROGRAMMING INTERFACE

The MegaScan FDP-6120 device is opened by a call to *open*. Currently, only minor device 0 is supported. If a non-zero minor device is specified, then an error will occur and *errno* will be set to ENOENT. Only one user is allowed to open the FDP-6120 board at a time, unless the subsequent user is the superuser. If a subsequent open is attempted by another user who is not the superuser, an error will result and *errno* will be set to EPERM.

Read and write interfaces are provided to access the memory of the Texas Instruments 34020 processor in the FDP-6120. Seek addresses are specified in bytes. Because addresses on a 34020 processor are given in bits, it is necessary to divide the bit address by 8 to get a byte address. The state of the 34020 is saved before each read and write, and restored when completed.

The following *ioctl* calls allow a user program to access the FDP-6120 board directly:

MSFDP_GET_INFO

This *ioctl* returns a structure containing the addresses of the board described in the section "Configuration" above. The structure returned is:

```
typedef struct msfdp_info {
  char   *msfdp_addr;    /* physical address */
  char   *msfdp_endaddr; /* physical address */
  short  msfdp_csr;      /* i/o port */
  short  msfdp_endcsr;   /* i/o port */
  } msfdp_info_t;
```

This structure is filled in with information in the seventh through tenth fields of the *sdevice* file. It is stored at the location specified by the argument that is passed.

MSFDP_MAP_BOARD
: This *ioctl* allows the process to have direct access to the FDP-6120 memory as well as allowing the user process to do in and out instructions at the I/O ports specified in the *sdevice* file. The argument passed is the address of the memory to map to the FDP-6120 board. It must be page (4096 byte) aligned and span a length specified by the *sdevice* file (which is also a multiple of 4096 bytes). The memory that had existed at the argument passed will be irretrievably deleted, and the current contents of the FDP-6120 memory will be placed at those locations. One way of allocating the virtual memory in the user's address space that will be mapped to the FDP-6120 memory is to call *malloc* requesting (*length* + 4096) bytes. Then, using the address that *malloc* returns, round it up to the next page (4K) boundary and use the result as the argument to this *ioctl*.

MSFDP_UNMAP_BOARD
: This *ioctl* call unmaps the direct access to the FDP-6120 memory as well as preventing the process from doing in and out instructions at the I/O ports specified in the *sdevice* file. The map must have been set up with MSFDP_MAP_BOARD by the current process. The argument passed should be the same one given to MSFDP_MAP_BOARD. The memory where the FDP-6120 memory had been becomes zeros.

FILES

/dev/msfdp

SEE ALSO

Xmsfdp(1).
sdevice(4) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual*.

NAME

nnp – Number Nine PEPPER PRO1280 driver

DESCRIPTION

The Number Nine PEPPER PRO1280 kernel driver facilitates accessing the Number Nine PEPPER PRO1280 board. It is a regular character-based driver that manages one PRO1280 board. It is used by programs that draw using the graphics modes of the PRO1280.

CONFIGURATION

The *sdevice*(4) file **/etc/conf/sdevice.d/nnp** describes the memory addresses of the PRO1280 board. Since the memory address of the PRO1280 are not configurable, it should not be necessary to change the default *sdevice* file. The ninth and tenth fields in this file describe the starting and ending memory addresses. These fields are given in hexadecimal. The default *sdevice* file entry is:

```
nnp   Y   1   0   0   0   0   0   cf000   cffff
```

If you change the default *sdevice* entry, you will have to build a new kernel and boot it for these changes to become effective.

PROGRAMMING INTERFACE

The Number Nine PEPPER PRO1280 device is opened by a call to *open*. Currently, only minor device 0 is supported. If a non-zero minor device is specified, then an error will occur and *errno* will be set to ENOENT. A simple check is done to see if the PRO1280 board is in the system. If it is not, an error will result and *errno* will be set to ENXIO. A memory address clash can also cause this error. Only one user is allowed to open the PRO1280 board at a time, unless the subsequent user is the superuser. If a subsequent open is attempted by another user who is not the superuser, an error will result and *errno* will be set to EPERM.

Read and write interfaces are provided to access the memory of the Texas Instruments 34010 processor in the PRO1280. Seek addresses are specified in bytes. Because addresses on a 34010 processor are given in bits, it is necessary to divide the bit address by 8 to get a byte address. The state of the 34010 is saved before each read and write, and it is restored when they are completed.

The following *ioctl* calls allow a user program to access the PRO1280 board directly:

NNP_GET_INFO

This *ioctl* returns a structure containing the addresses of the board described in the section "Configuration" above. The structure returned is:

```
typedef struct  nnp_info  {
  char   *nnp_addr;      /* physical address */
  char   *nnp_endaddr;   /* physical address */
  }  nnp_info_t;
```

This structure is filled in with information in the ninth through tenth fields of the *sdevice* file. It is stored at the location specified by the argument that is passed.

NNP_MAP_BOARD

This *ioctl* allows the process to have direct access to the PRO1280 memory. The argument passed is the address of the memory to map to the PRO1280 board. It must be page (4096 byte) aligned and span a length specified by the *sdevice* file (which is also a multiple of 4096 bytes). The memory that had existed at the argument passed will be irretrieva deleted, and the current contents of the PRO1280 memory wii be placed at those locations. One way of allocating the virtual memory in the user's address space that will be mapped to the PRO1280 memory is to call *malloc* requesting (*length* + 4096) bytes. Then, using the address that *malloc* returns, round it up to the next page (4K) boundary and use the result as the argument to this *ioctl*.

NNP_UNMAP_BOARD

This *ioctl* call unmaps the direct access to the PRO1280 memory. The map must have been set up with NNP_MAP_BOARD by the current process. The argument passed should be the same one given to NNP_MAP_BOARD. The memory where the PRO1280 memory had been becomes zeros.

FILES

/dev/nnp

SEE ALSO

Xnnp(1).
sdevice(4) in the *INTERACTIVE SDS Guide and Programmer's Rej ence Manual*.

NAME

nnp – Number Nine PEPPER PRO1024ISA driver

DESCRIPTION

The Number Nine PEPPER PRO1024ISA kernel driver facilitates accessing the Number Nine PEPPER PRO1024ISA board. It is a regular character-based driver that manages one PRO1024ISA board. It is used by programs that draw using the graphics modes of the PRO1024ISA.

CONFIGURATION

The *sdevice*(4) file **/etc/conf/sdevice.d/nnpi** describes the memory addresses of the PRO1024ISA board. The ninth and tenth fields in this file describe the starting and ending memory addresses. These fields are given in hexadecimal. The default *sdevice* file entry with an memory base address of C:C000 is:

```
nnpi  Y  1  0  0  0  0  0  cc000  ccfff
```

As an example, if you want to change the default *sdevice* to use an addresses of D:C000, then the following *sdevice* file is used:

```
nnpi  Y  1  0  0  0  0  0  dc000  dcfff
```

If you change the default *sdevice* entry, you will have to build a new kernel and boot it for these changes to become effective.

PROGRAMMING INTERFACE

The Number Nine PEPPER PRO1024ISA device is opened by a call to *open*. Currently, only minor device 0 is supported. If a non-zero minor device is specified, then an error will occur and *errno* will be set to ENOENT. A simple check is done to see if the PRO1024ISA board is in the system. If it is not, an error will result and *errno* will be set to ENXIO. A memory address clash may also cause this error. Only one user is allowed to open the PRO1024ISA board at a time, unless the subsequent user is the superuser. If a subsequent open is attempted by another user who is not the superuser, an error will result and *errno* will be set to EPERM.

Read and write interfaces are provided to access the memory of the Texas Instruments 34010 processor in the PRO1024ISA. Seek addresses are specified in bytes. Because addresses on a 34010 processor are given in bits, it is necessary to divide the bit address by 8 to get a byte address. The state of the 34010 is saved before each read and write, and restored when completed.

The following *ioctl* calls allow a user program to access the PRO1024ISA board directly:

NNPI_GET_INFO

This *ioctl* returns a structure containing the addresses of the board described in the section "Configuration" above. The structure returned is:

```
typedef struct  nnpi_info  {
  char   *nnpi_addr;      /* physical address */
  char   *nnpi_endaddr;   /* physical address */
  }  nnpi_info_t;
```

This structure is filled in with information in the ninth through tenth fields of the *sdevice* file. It is stored at the location specified by the argument that is passed.

NNPI_MAP_BOARD
: This *ioctl* allows the process to have direct access to the PRO1024ISA memory. The argument passed is the address of the memory to map to the PRO1024ISA board. It must be page (4096 byte) aligned and span a length specified by the *sdevice* file (which is also a multiple of 4096 bytes). The memory that had existed at the argument passed will be irretrievably deleted, and the current contents of the PRO1024ISA memory will be placed at those locations. One way of allocating the virtual memory in the user's address space that will be mapped to the PRO1024ISA memory is to call *malloc* requesting (*length* + 4096) bytes. Then, using the address that *malloc* returns, round it up to the next page (4K) boundary and use the result as the argument to this *ioctl*.

NNPI_UNMAP_BOARD
: This *ioctl* call unmaps the direct access to the PRO1024ISA memory. The map must have been set up with NNPI_MAP_BOARD by the current process. The argument passed should be the same one given to NNPI_MAP_BOARD. The memory where the PRO1024ISA memory had been becomes zeros.

FILES

/dev/nnpi

SEE ALSO

Xnnpi(1).
sdevice(4) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual*.

NAME

pty – pseudo terminal driver

DESCRIPTION

The *pty* driver provides support for a device-pair termed a *pseudo terminal*. A pseudo terminal is a pair of character devices, a *master* device and a *slave* device. The slave device provides an interface identical to that described in *tty*(4). However, whereas all other devices that provide the interface described in *tty*(4) have a hardware device of some sort behind them, the slave device has, instead, another process manipulating it through the master half of the pseudo terminal. That is, anything written on the master device is given to the slave device as input and anything written on the slave device is presented as input on the master device.

In configuring, if an optional "count" is given in the specification, that number of pseudo terminal pairs are configured; the default count is 16.

The following *ioctl* calls apply only to pseudo terminals:

TIOCPKT

Enable/disable *packet* mode. Packet mode is enabled by specifying (by reference) a nonzero parameter and disabled by specifying (by reference) a zero parameter. When applied to the master side of a pseudo terminal, each subsequent *read* from the terminal will return data written on the slave part of the pseudo terminal preceded by a zero byte (symbolically defined as TIOCPKT_DATA), or a single byte reflecting control status information. In the latter case, the byte is an inclusive-or of zero or more of the bits:

TIOCPKT_FLUSHREAD
: whenever the *read* queue for the terminal is flushed.

TIOCPKT_FLUSHWRITE
: whenever the *write* queue for the terminal is flushed.

TIOCPKT_STOP
: whenever output to the terminal is stopped with ^S.

TIOCPKT_START
: whenever output to the terminal is restarted.

TIOCPKT_DOSTOP
: whenever *t_stopc* is ^S and *t_startc* is ^Q.

TIOCPKT_NOSTOP
: whenever the start and stop characters are not ^S/^Q.

While this mode is in use, the presence of control status information to be read from the master side may be detected by a *select* for exceptional conditions.

This mode is used by *rlogin*(1C) and *rlogind*(1M) to implement a remote-echoed, locally ^S/^Q flow-controlled remote login with proper back-flushing of output; it can be used by other similar programs.

TIOCREMOTE

A mode for the master half of a pseudo terminal, independent of TIOCPKT. This mode causes input to the pseudo terminal to be flow controlled and not input edited (regardless of the terminal mode). Each *write* to the control terminal produces a record boundary for the process reading the terminal. In normal usage, a *write* of data is like the data typed as a line on the terminal; a write of 0 bytes is like typing an end-of-file character. TIOCREMOTE can be used when doing remote line editing in a window manager, or whenever flow-controlled input is required.

TIOCPTYCTL

Enable/disable *ptyctl* mode. In *ptyctl* mode, each buffer of data read from or written to the pseudo terminal will start with a flag byte. The flag byte identifies the rest of the buffer. A flag byte of data means the rest of the buffer contains user data. A flag byte other than data means the data buffer contains control information for the pseudo terminal.

Flag bytes for writing include:

pty_ctl3_cmd_user_data
The rest of the buffer contains user data.

pty_ctl3_cmd_char_echo
Control echoing of input characters. The data byte following the command byte must be either 1 (TRUE) or 0 (FALSE). If the data byte is 1, input characters will be echoed if, and only if, the ECHO flag is set on the the slave side of the pseudo terminal. If the data flag is 0, input characters will never be echoed. Character echoing will be true by default when the service is started.

pty_ctl3_cmd_break
A break should be sent to the line discipline.

pty_ctl3_cmd_first_unused
The highest command literal defined.

Flag bytes for reading include:

pty_ctl3_info_user_data
The rest of the buffer contains user data.

pty_ctl3_info_flush_input
All input to the pseudo terminal should be flushed.

pty_ctl3_info_change_ioctl
The slave side program has changed the state of the line discipline's ioctl information.

pty_ctl3_info_want_data
The slave side program has done a read requesting more data.

pty_ctl3_info_output_suspend
Output from the pseudo terminal has been suspended.

pty_ctl3_info_output_resumed
Output from the pseudo terminal has been resumed.

pty_ctl3_info_break
A break has been sent to output.

pty_ctl3_info_first_unused
Specifies the first unused info literal.

FILES

/dev/pty[p–r][0–9a–f]
master pseudo terminals
/dev/tty[p–r][0–9a–f]
slave pseudo terminals

NAME

pw – Pixelworks Clipper Graphics Controller driver

DESCRIPTION

The Pixelworks Clipper Graphics Controller (PCGC) kernel driver is used to access the card in Graphics mode. It is the responsibility of the application to control the mode of the board (CGA or Graphics) so that CGA commands are not sent to the board when it is in Graphics mode, and Graphics commands are not sent to the board when it is in CGA emulation mode.

Write Calls

The *write*(2) routine is used to send a display list to the graphics card. The display list must end with a DLEND opcode. The driver will send a DLRUN command and the address of the display list to the card. The card will then be in master mode, and it will execute the display list. The entire display list must be contained in one 4096-byte page that is aligned on a page boundary. The argument to the *write*(2) call is the virtual address of the display list to be executed.

Ioctl Calls

The following commands are used with the *ioctl*(2) call:

P_CLIPOUT
: This call is used to send one 16-bit word to the board. The argument to the *ioctl* call is the value of the word to be sent to the board.

P_CLIPIN
: This call gets one 16-bit word from the board. The argument to the *ioctl* call is the address where the 16-bit word will be put.

P_GRAFOUT
: This call causes the driver to transfer one or more words between a user-specified buffer and the graphics card. The argument to the *ioctl* call is the address of the *pw_ioctl* structure defined in *pw.h*:

```
struct pw_ioctl {
    char *addr;     /* address of graphics buffer */
    int data;       /* number of 16-bit words to transfer */
    int mode;       /*0 = output buffer to graphics card */
                    /*1 = get data from graphics card */
}
```

P_HRESET
: This call forces the driver to flush the I/O queue and to do a hardware reset of the card.

P_RTN_VAL
: This call retrieves the last interrupt reason code from the driver. The argument to the *ioctl* call is the address where the interrupt reason code will be located.

P_IO_INT
: This call informs the driver that the next driver command (either P_GRAFOUT or P_CLIPOUT) will generate an I/O

mode interrupt. I/O mode interrupts are caused by the *WAIT* and *blt* opcodes. If these opcodes are sent in I/O mode and the P_IO_INT call is not made, then the system is likely to hang because of subsequent accesses to the board while it is busy.

P_CGAON
: This command turns on CGA emulation if it has been enabled with dip switches on the board. Otherwise, this command does nothing.

P_CGAOFF
: This command turns off CGA emulation if it has been enabled with dip switches on the board and is currently enabled. Otherwise, this command does nothing.

P_RTN_SIG
: This command instructs the driver to send a signal when an interrupt is received by the driver. This allows the application to be informed when the card completes execution of a display list. The argument to the *ioctl* call is the signal to send when the interrupt is received (*signal*(2)).

P_DLRUN
: This command results in a display list being executed from the physical buffer previously allocated with P_MAPBUF. The argument to the *ioctl* call is the physical address of the display list to be executed. The call will return when the display list has completed execution.

P_DLRUN_NW
: This command results in a display list being executed from the physical buffer previously allocated with P_MAPBUF. The argument to the *ioctl* call is the physical address of the display list to be executed. The call will return after the display list has been sent to the board.

P_MAPBUF
: This command maps a chunk of the user's virtual address space to the physical memory buffer allocated by the driver at boot-up time. The argument to the *ioctl* call is the address of the *pw_ioctl* structure defined in *pw.h*.

```
struct pw_ioctl {
    char *addr;     /* virtual address of graphics buffer */
    int data;       /* number of bytes to map */
    int mode;       /* not used */
}
```

It is important that the virtual address be on a page boundary, and that the byte count be an even number of pages. The size of a page in bytes is defined to be NBPP, which is defined in **sys/immu.h**. The physical address of the buffer is passed back in the *pw_ioctl* structure:

```
struct pw_ioctl {
    char_t addr;    /* physical address of graphics buffer */
    int data;       /* not used */
    int mode;       /* not used */
}
```

If the number of bytes to map is set to −1, then the driver will map the entire physical buffer (up to 131072 bytes) and pass back the size of the buffer in the data field in the *pw_ioctl* structure.

P_FREEBUF
: This command unmaps the virtual address space mapped with the P_MAPBUF command.

FILES

/dev/pw0
/usr/include/pw.h
/usr/include/pw_opeds.h

SEE ALSO

Xpw(1),
close(2), ioctl(2), open(2), signal(2), write(2) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual*.

WARNINGS

Passing bad physical addresses or bad display lists to the controller can "hang" the system. Some commands sent to the board, such as large polygon fills and screen clears, take a long time to execute. Sending these commands with the P_GRAFOUT *ioctl* command can "hang" the system. It is advisable to send all display lists to the board with the P_DLRUN or P_DLRUN_NW *ioctl* commands.

NAME

rren2 – Renaissance Rendition II driver

DESCRIPTION

The Renaissance Rendition II kernel driver facilitates accessing the Renaissance Rendition II board. It is a regular character-based driver that manages one Rendition II board. It is used by programs that draw using the graphics modes of Rendition II.

CONFIGURATION

The *sdevice*(4) file **/etc/conf/sdevice.d/rren2** describes the I/O port addresses as well as the shared memory addresses. Sinced the memory and I/O addresses are not configurable, it should not be necessary to change the default **sdevice** file. The seventh and eighth fields in this file describe the starting and ending input/output port addresses. The ninth and tenth fields in this file describe the starting and ending shared memory addresses. The shared memory fields are not used for the *Xrren2* server for X11. All of these fields are given in hexadecimal. The default *sdevice* file entry with an I/O base address of 0290h is and shared memory address of 0d0000h is:

```
rren2 Y 1 0 0 0 290 29f d0000 d1fff
```

If you change the default *sdevice* entry, you will have to build a new kernel and boot it for these changes to become effective.

PROGRAMMING INTERFACE

The Rendition II device is opened by a call to *open*. Currently, only minor device 0 is supported. If a non-zero minor device is specified, then an error will occur and *errno* will be set to ENOENT. A simple check is done to see if the Rendition II board is in the system. If it is not, an error will result and *errno* will be set to ENXIO. An I/O address clash may also cause this error. Only one user is allowed to open the Rendition II board at a time, unless the subsequent user is the superuser. If a subsequent open is attempted by another user who is not the superuser, an error will result and *errno* will be set to EPERM.

Read and write interfaces are provided to access the memory of the Texas Instruments 34010 processor in the Rendition II. Seek addresses are specified in bytes. Because addresses on a 34010 processor are given in bits, it is necessary to divide the bit address by 8 to get a byte address. The state of the 34010 is saved before each read and write, and it is restored when they are completed.

The *ioctl* following calls allow a user program to access the Rendition II board directly:

RREN2_GET_INFO

This *ioctl* returns a structure containing the addresses of the board described in the section "Configuration" above. The structure returned is:

```
typedef struct  rren2_info  {
  short   rren2_port;       /* i/o port */
  short   rren2_endport;    /* i/o port */
  char    *rren2_sram;      /* physical address */
  char    *rren2_endsram;   /* physical address */
  }  rren2_info_t;
```

This structure is filled in with information in the seventh through tenth fields of the *sdevice* file. It is stored at the location specified by the argument that is passed.

RREN2_MAP_BOARD
: This *ioctl* allows the process to do in and out instructions at the I/O ports specified in the *sdevice* file. The argument passed is unused.

RREN2_UNMAP_BOARD
: This *ioctl* prevents the process from doing in and out instructions at the I/O ports specified in the *sdevice* file. The argument passed is unused.

RREN2_MAP_SRAM
: This *ioctl* allows the process to have direct access to the shared memory. The argument passed is the address of the memory to map to the Rendition II board. It must be page (4096 byte) aligned and span a length specified by the *sdevice* file (which is also a multiple of 4096 bytes). The memory that had existed at the argument passed will be irretrievably deleted, and the current contents of the shared memory will be placed at those locations. One way of allocating the virtual memory in the user's address space that will be mapped to the shared memory is to call *malloc* requesting (*length* + 4096) bytes. Then, using the address that *malloc* returns, round it up to the next page (4K) boundary and use the result as the argument to this *ioctl*.

RREN2_UNMAP_SRAM
: This *ioctl* call unmaps the direct access to the shared memory. The map must have been set up with RREN2_MAP_SRAM by the current process. The argument passed should be the same one given to RREN2_MAP_SRAM. The memory where the shared memory had been becomes zeros.

FILES

/dev/rren2

SEE ALSO

Xrren2(1).
sdevice(4) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual*.

NAME

sp200 – Spectre SP200 Display Controller driver

DESCRIPTION

The Spectre SP200 kernel driver facilitates accessing the Spectre SP200 board. It is a regular character-based driver that manages one SP200 board. It is used by programs that draw using the graphics modes of SP200.

CONFIGURATION

The *sdevice*(4) file **/etc/conf/sdevice.d/sp200** describes the I/O port addresses and the memory addresses of the SP200 board. The seventh and eighth fields in this file describe the starting and ending input/output port addresses. The ninth and tenth fields in this file describe the starting and ending memory addresses. The length of memory must be either 16K or 32K. All of these fields are given in hexadecimal. The default *sdevice* file entry is:

```
sp200 Y 1 0 0 0 2b0 2bf d0000 d7fff
```

As an example, if you want to change the default *sdevice* to use I/O ports of 200-20f, and only 16K of memory from CC0000-CFFFFF, then the following *sdevice* file would be used:

```
sp200 Y 1 0 0 0 200 20f cc000 cffff
```

If you change the default *sdevice* entry, you will have to build a new kernel and boot it for these changes to become effective.

PROGRAMMING INTERFACE

The Spectre SP200 device is opened by a call to *open*. Currently, only minor device 0 is supported. If a non-zero minor device is specified, then an error will occur and *errno* will be set to ENOENT. Only one user is allowed to open the SP200 board at a time, unless the subsequent user is the superuser. If a subsequent open is attempted by another user who is not the superuser, an error will result and *errno* will be set to EPERM.

Read and write interfaces are provided to access the memory of the Texas Instruments 34020 processor in the SP200. Seek addresses are specified in bytes. Because addresses on a 34020 processor are given in bits, it is necessary to divide the bit address by 8 to get a byte address. The state of the 34020 is *not* saved before each read and write and *not* restored when the read or write is completed. The kernel uses and does not restore segment 4, and the control register.

The following *ioctl* calls allow a user program to access the SP200 board directly:

SP200_GET_INFO

This *ioctl* returns a structure containing the addresses of the board described in the section "Configuration" above. The structure returned is:

```
typedef struct  sp200_info  {
  short   sp200_port;       /* i/o port */
  short   sp200_endport;    /* i/o port */
  char    *sp200_addr;      /* physical address */
  char    *sp200_endaddr;   /* physical address */
  }  sp200_info_t;
```

This structure is filled in with information in the seventh through tenth fields of the *sdevice* file. It is stored at the location specified by the argument that is passed.

SP200_MAP_BOARD

This *ioctl* allows the process to do in and out instructions at the I/O ports as well as allowing direct access to the memory specified in the *sdevice* file. The argument passed is the address of the memory to map to the SP200 board. It must be page (4096 byte) aligned and span a length specified by the *sdevice* file (which is also a multiple of 4096 bytes). The memory that had existed at the argument passed will be irretrievably deleted, and the current contents of the memory allocated to the SP200 will be placed at those locations. The user must store values in the memory base registers to use the SP200 memory. One way of allocating the virtual memory in the user's address space that will be mapped to the SP200 memory is to call *malloc* requesting (*length* + 4096) bytes. Then, using the address that *malloc* returns, round it up to the next page (4K) boundary and use the result as the argument to this *ioctl*.

SP200_UNMAP_BOARD

This *ioctl* call unmaps the direct access to the SP200 memory as well as preventing the process from doing in and out instructions at the I/O ports specified in the *sdevice* file. The map must have been set up with SP200_MAP_BOARD by the current process. The argument passed should be the same one given to SP200_MAP_BOARD. The memory where the SP200 memory had been becomes zeros.

FILES

/dev/sp200

SEE ALSO

Xsp200(1).
sdevice(4) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual*.

NAME

tisdb – Texas Instruments TMS34010 Software Development Board driver

DESCRIPTION

The Texas Instruments TMS34010 Software Development Board kernel driver facilitates accessing the Texas Instruments TMS34010 Software Development Board. It is a regular character-based driver that manages one SDB. It is used by programs that draw using the graphics modes of the SDB.

CONFIGURATION

The *sdevice*(4) file **/etc/conf/sdevice.d/tisdb** describes the memory addresses of the SDB. The ninth and tenth fields in this file describe the starting and ending memory addresses. These fields are given in hexadecimal. The default *sdevice* file entry with an memory base address of >C7000 is:

```
tisdb  Y   1   0   0   0   0   0   c7000  c7fff
```

If you want a memory base address of >E7000 instead, then the following *sdevice* entry should be used:

```
tisdb  Y   1   0   0   0   0   0   e7000  e7fff
```

If you change the default *sdevice* entry, you will have to build a new kernel and boot it for these changes to become effective.

NOTE: Both of the PALS given with the SDB generate a memory address that conflicts with the VGA.

PROGRAMMING INTERFACE

The Texas Instruments TMS34010 Software Development Board device is opened by a call to *open*. Currently, only minor device 0 is supported. If a non-zero minor device is specified, then an error will occur and *errno* will be set to ENOENT. A simple check is done to see if the SDB is in the system. If it is not, an error will result and *errno* will be set to ENXIO. A memory address clash may also cause this error. Only one user is allowed to open the SDB at a time, unless the subsequent user is the superuser. If a subsequent open is attempted by another user who is not the superuser, an error will result and *errno* will be set to EPERM.

Read and write interfaces are provided to access the memory of the Texas Instruments 34010 processor in the SDB. Seek addresses are specified in bytes. Because addresses on a 34010 processor are given in bits, it is necessary to divide the bit address by 8 to get a byte address. The state of the 34010 is saved before each read and write, and it is restored when they are completed.

The following *ioctl* calls allow a user program to access the SDB directly:

TISDB_GET_INFO

This *ioctl* returns a structure containing the addresses of the board described in the section "Configuration" above. The structure returned is:

```
typedef struct  tisdb_info  {
  char    *tisdb_addr;      /* physical address */
  char    *tisdb_endaddr;   /* physical address */
  }  tisdb_info_t;
```

This structure is filled in with information in the ninth through tenth fields of the *sdevice* file. It is stored at the location specified by the argument that is passed.

TISDB_MAP_BOARD

This *ioctl* allows the process to have direct access to the SDB memory. The argument passed is the address of the memory to map to the SDB. It must be page (4096 byte) aligned and span a length specified by the *sdevice* file (which is also a multiple of 4096 bytes). The memory that had existed at the argument passed will be irretrievably deleted, and the current contents of the SDB memory will be placed at those locations. One way of allocating the virtual memory in the user's address space that will be mapped to the SDB memory is to call *malloc* requesting (*length* + 4096) bytes. Then, using the address that *malloc* returns, round it up to the next page (4K) boundary and use the result as the argument to this *ioctl*.

TISDB_UNMAP_BOARD

This *ioctl* call unmaps the direct access to the SDB memory. The map must have been set up with TISDB_MAP_BOARD by the current process. The argument passed should be the same one given to TISDB_MAP_BOARD. The memory where the SDB memory had been becomes zeros.

FILES

/dev/tisdb

SEE ALSO

Xtisdb(1).
sdevice(4) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual*.

NAME

vdm – video memory map

SYNOPSIS

#include <sys/kd.h>
#include <sys/vdm.h>

DESCRIPTION

vdm is a pseudo-device driver used to map the memory of a bitmapped display into the virtual address space of a process. It is currently used only for the Cornerstone Video Controller X11 server (see *Xcvc*(1)), but the interface is general and may be used to map memory for other devices.

The driver supports only the single *ioctl* VDM_MAP to map a region of physical memory. The argument to the *ioctl* must be a pointer to a **struct kd_memloc** (see KDMAPDISP in *display*(7)). The **vaddr** and **length** elements of the structure must specify a valid address range within the process address space. (A process may use *malloc*(3) to allocate the area. This will not consume actual memory as long as the memory is not accessed until after it has been mapped.) The **physaddr** must be the location of the physical memory for the device. *vdm* will refuse to map to memory already in use by the kernel. The **iofig** element of the structure is ignored.

The effective user ID of the process must be **root** (the superuser) to use this device.

FILES

/dev/vdm

SEE ALSO

Xcvc(1).
malloc(3C) in the *INTERACTIVE SDS Guide and Programmer's Reference Manual*.

INTERACTIVE

DC00217-2Z