Part Number 800-1106-01
Revision: A of 1st May 1984
For: Sun System Release 1.1

System Summary
and
Change Notes
for the
Sun Workstation
Release 1.1 of May 1984

This document is a summary of the Sun System Release 1.1, and a description of changes made relative to previous major releases. This document is in three major parts:

Part I contains a description of changes to the system between Sun System Release 1.0 (November 1983) and Sun System Release 1.1 (April 1984).

Part II contains a description of changes to the system between Sun System Release 0.4 (May 1983) and Sun System Release 1.0 (November 1983).

Part III is a summary of the differences between the Sun system and the Berkeley 4.2 Software Distribution. Sun has embarked upon a long-term program to improve upon the system. To this end, there are inevitable discrepancies betwen the Berkeley Software Distribution and the Sun version of this software. Part III describes those differences that are likely to have substantial impact.

Sun Microsystems, Inc.,
2550 Garcia Avenue
Mountain View
California 94043
(415) 960-1300

Table of Contents

List of Tables

Introduction

Welcome to the Sun Workstation and its operating software based on the UNIX† operating system.

New Customers Please Note

that these notes provide a description of the major features of this release and changes since the previous release. If you have just received your first shipment of a Sun Workstation, the first manual you should read is the *System Manager's Manual*, which contains a detailed explanation of how to set up the Sun Workstation and how to install the operating software on it. The Sun software is based upon the system known as 4.2 BSD — a version of the UNIX operating system for the DEC VAX machine family, with enhancements from the University of California at Berkeley. If your knowledge of the UNIX system is sketchy or nonexistent, we recommend reading the manual entitled *Beginner's Guide to the Sun System* and finding your way through the various manuals by following the interesting pointers from there on.

Existing Customers Please Note

that if you already have a Sun System, these notes provide a description of the major feaures of this release and changes since the previous release.

This document is in three major parts:

Part I contains a description of changes to the system between Sun System Release 1.0 (November 1983) and Sun System Release 1.1 (April 1984).

Part II contains a description of changes to the system between Sun System Release 0.4 (May 1983) and Sun System Release 1.0 (November 1983).

Part III is a summary of the differences between the Sun system and the Berkeley 4.2 Software Distribution. Sun has embarked upon a long-term program to improve upon the system. To this end, there are inevitable discrepancies betwen the Berkeley Software Distribution and the Sun version of this software. Part III describes those differences that are likely to have substantial impact.

Note that if you are upgrading from a 0.4 system to a 1.1 system, you should be able to go straight there without necessarily taking a path through the 1.0 system — this is especially true of some of the changes to the **SunCore** graphics interfaces.

In parts I and II of this document, we cover three topics of interest to you:

New or Changed Utilities

describes how the utilities (user-accessible programs) have changed in terms of new utilities,

† UNIX is a trademark of Bell Laboratories.

Sun Workstation, and the combination of Sun with a numeric suffix are trademarks of Sun Microsystems, Inc.

changes to the user interface, or utilities that have been deleted.

New or Changed System Interfaces
describes changes to kernel calls and library routines which necessitate changes to application code or recompilation.

New or Changed Manuals
describes new or changed documentation.

Part I — Changes From Release 1.0 to Release 1.1

This part of the document contains a description of changes to the system between Sun System Release 1.0 (November 1983) and Sun System Release 1.1 (April 1984).

1. Bugs and Features that should be Bugs

1.1. UNIX Domain Sockets

UNIX domain sockets are known to be riddled with bugs. Sun Microsystems has no immediate plans to fix the bugs at this time.

Customers are encouraged to use Internet domain sockets to avoid problems.

1.2. Limitations on UDP Data Size

Valid size of a UDP data packet is between 1 and 2032 bytes. This is a *feature*.

If the size of the data packet is greater than 2032 bytes but less than 2048 bytes, the *sendto* system call returns an OK status, but nothing is sent. This is a *bug*.

1.3. Limitations on UDP Broadcast Size

The maximum size of a UDP broadcast packet is 1472 bytes.

1.4. ICMP Bugs

ICMP is hopelessly eaten by bugs. Sun Microsystems is planning to clean up the bugs in the future.

2. Changes to the Operating System Software

Release 1.1 of the Sun system is based on the final 4.2 Berkeley Software Distribution for the VAX.

2.1. CPU PROMS

Screen output:
Bugs in the screen output driver were fixed. They caused the cursor to disappear and Tektronix 4014 partial emulation mode (on the model 100, 100U, 150, and 150U) to fail. Character insert/delete is now supported on the Model 120 and 170; the termcap entries for all monochrome screens are now the same. Watchdog resets now cause the frame buffer to be remapped, preventing the screen image from freezing.

Keyboard input:
PROM keyboard support is simpler, since the Unix kernel now handles the keyboard while it is up. The PROM no longer deals with cursor or function keys. It generates control codes and Caps Lock by and-ing the unshifted keys, rather than with separate tables.

Power-up and bootstrap:
The workstation's Ethernet address is now printed in the power-up banner. Ethernet bootstrap code now deals with the 'ec' (3Com Ethernet) or 'ie' (Intel/Sun Ethernet) devices, rather than with 'nd'. The bootstrap code no longer works on Interphase 2880 or 2181 disk controllers (this was never a supported product).

Commands:
The Trace and Breakpoint commands were deleted due to lack of space. The 'K0' command now causes a 68000 'reset' instruction, rather than resetting the stack pointer. (The stack pointer can be reset in the usual way by modifying register A7.)

New device support:
Sun-2 CPU board, Sun-2 video and keyboard (Model 120/170), SCSI disk and tape.

Power-up and bootstrap:
Better diagnostics including LED values for failures, and readable error messages. Improved watchdog handling allows Unix to catch watchdogs. Improved boot interface allows Unix to specify booted device and file, and reboot without clearing memory.

Commands:
New 68010 registers accessible in 'r' command. Transparent mode (Sun acts as terminal over serial port) removed.

2.2. MAXUSERS Changed from 2 Users to 4 Users

The MAXUSERS field in the GENERIC configuration file has been changed from 2 users to 4 users. If you are running a single-user workstation, or a workstation with only one Megabyte of memory, it is important to generate your own kernel with the maximum number of users adjusted to suit your needs. For information on configuring and building a new kernel, see the section on *Kernel Configuration* in the chapter entitled *Installing UNIX for the First Time* in the *System Manager's Manual for the Sun Workstation*.

2.3. Limitations on Program Text, Data, and Stack Size

Release 1.1 now supports a maximum text size of a program beyond 1.5 Megabytes. The limit on the text size is now dependent on the amount of swap space available, up to a max of 6144K bytes. The text and data limits depend on how much swap space is configured. This table

shows approximate sizes (based on calculation):

Swap space	*Unit of allocation*	*Max Text*	*Max Data*
< 6 Meg	256 sectors	1536K	1776K
< 8 Meg	512 sectors	3072K	3312K
< 12 Meg	1024 sectors	6144K	6128K
< 16 Meg	2048 sectors	6144K	11248K
>= 16 Meg	4096 sectors	6144K	16320K

The stack segment size is initially limited to 512K bytes by the system but may be as large as the data segment size if it is increased with the setrlimit system call. At no time may the sum of the text, data, and stack segment sizes be greater than 16320K.

2.4. New Software Drivers

- Software driver for the Systech parallel interface board, an output-only DMA device. The device has one channel for Versatec devices and one channel for Centronics devices, with an optional long lines interface for Versatec devices.
- Software driver for the Systech MTI card providing 8 (MTI-800) or 16 (MTI-1600) serial communication lines with modem control. Each line behaves as described in *tty*(4). Input and output for each line may independently be set to run at any of 16 speeds.
- Software driver for the Sun-2 Ethernet board.
- Software driver for the Ikon board — a Versatec interface driver exists in the generic kernel.

Also please note that the driver for the Central Data Octal Serial Interface Board is in the generic kernel but is not a Sun supported product. Here is a list of things we are adding to the 1.1 release as supported software (with the exception of the octal driver):

2.4.1. Support for 9600 Baud

Here is the status of the operating system's support for 9600 Baud input and output:

input The operating system supports sustained input at 9600 Baud on *one* CPU-board serial port *provided* that the other end of the communications line supports flow control

output The operating system can concurrently support output at 9600 Baud through *one* CPU-board serial port.

2.5. Changes to Sendmail

The *sendmail.cf* file has gone. Now you must choose either */usr/lib/sendmail.domain.cf* or */usr/lib/sendmail.generic.cf* to construct your configuration file for mail routing. See the *sendmail* installation procedures in the *System Manager's Manual*.

2.6. New Version of Network News

The news system supplied with this release is based on the Usenet 2.10.1 version of the news system, including many bug fixes.

2.7. Change to Mail Interface

The *mail* interface has a significant change in the way the **reply** command works:

How the **reply** *command used to work*

Previously there were two **reply** commands: **reply** (lower-case **r**), and **Reply** (upper-case **R**). **reply** (with a lower-case **r**) would reply to everyone who got the original message and **Reply** (with an upper-case **R**) would reply only to the sender.

How the **reply** *command works now*

In the new version of *Mail* their behavior is reversed. That is, a **reply** (lower-case **r**) replies to the sender only, and a **Reply** (upper-case **R**) replies to everyone who got the original message!

However, if you set the **replyall** variable in your *.mailrc* file, they will behave as before.

There are also two new commands, namely **replyall** and **replysender**. **replyall** always replies to all names in the original message, and **replysender** always replies only to the sender, regardless of the setting of the **replyall** variable.

Note that setting the **replyall** option in */usr/lib/Mail.rc* will make the default be the same as the previous version of *mail*.

2.8. Keyboard-related include files moved

The files */usr/include/sun/kbd.h* and */usr/include/sun/kbio.h* have been moved into the */usr/include/sundev* directory.

2.9. Keyboard-related ioctl interface changed

Ioctl calls from the *kbio.h* interface should now be directed to */dev/kbd* instead of */dev/console*.

2.10. Changes to rdate

The Release 1.1 version of *rdate* is incompatible with the Release 1.0 version.

3. Applications Software Changes in Release 1.1

3.1. Versatec Printing Software

This release includes the Versatec support software, namely:

vtroff runs *troff*(1) sending its output through various programs to produce typeset output on a raster plotter such as a Benson-Varian or a Versatec.

vfontinfo displays information about fonts in the UNIX format.

vwidth translates from the width information stored in the *vfont* style format to the format expected by *troff*.

The 'man' pages for *vtroff*, *vfontinfo*, and *vwidth* can be found as an addendum to this document. Note that the 'man' pages did not get into the Section 1 manual pages in time for the release.

3.1.1. Changes to vfont Formats

The *vfont* format has been changed so that fonts are stored in local (MC68000) byte order instead of VAX byte order as they were previously.

To assist customers in adapting to this new format, a new *vswap*(1) utility exists for swapping the byte order in font files.

3.2. Screen Dump, Screen Load, and Raster Replicate Programs

Three new programs exist related to the Sun workstation's framebuffer:

screendump
reads out the contents of the console frame buffer (*/dev/fb*) on any model of Sun Workstation and outputs the display image in Sun standard rasterfile format (see */usr/include/rasterfile.h*) on the standard output.

screenload
accepts input in Sun standard rasterfile format (see */usr/include/rasterfile.h*) and attempts to display the input on the appropriate monitor (monochrome or color).

rastrepl
reads a file in rasterfile format (see */usr/include/rasterfile.h*) on the standard input, replicates each bit in both the *x* and *y* directions, and writes the resulting rasterfile to standard output.

The 'man' pages for *screendump*, *screenload*, and *rastrepl* can be found as an addendum to this document. Note that the 'man' pages did not get into the Section 1 manual pages in time for the release.

4. Programming Language Support in Release 1.1

4.1. DBX — Symbolic Debug Package

The *dbx* symbolic debug package was considerably enhanced for the 1.1 release. The specific areas of improvement are:

- *Dbx* works with FORTRAN programs
- New *dbx* **when** command which executes a specified set of *dbx* commands when a given condition is satisfied (such as a function being called).

- New *dbx* command **cont at** *line number*
- Type-ahead is allowed
- Addresses are now printed in hexadecimal instead of decimal
- One can now call routines not compiled with the **-g** compiler option (the number and types of parameters are not checked)
- **print p**, where *p* is a function pointer, prints the name of the function as well as the hexadecimal value
- The **help** command takes command name arguments
- various bug fixes

4.2. FORTRAN Programming Language

- FORTRAN now has a single-precision mathematical library (sqrt, sin, cos, etc.)
- FORTRAN logical unit-numbers can now be arbitrary nonnegative integers — unit numbers are no longer constrained to lie in the range 0 thru 19.
- The FORTRAN library has a new routine called *getfd* which returns the file descriptor of an external unit number if the unit is connected.

4.3. Pascal Programming Language

Pascal has had a major bug fixed in the code generated for **case** statements.

5. Changes in the 1.1 Release of the SunCore Graphics Package

Here are the major changes to the SunCore graphics package for release 1.1:

Pascal language interfaces
to the SunCore library are described in a new appendix.

Higher performance
floating-point library now exists which uses the hardware floating-point unit.

Multiple view surfaces
per physical device for devices in the window system and/or multiple physical devices of a single type. This added flexibility has required a change in view surface names (see below).

New view surface type
namely *cgpizwindd*, has been added for color windows.

5.1. View Surface Names

All SunCore routines which have a view surface argument must use the view surface structure defined in Appendix B of the SunCore Programmer's Reference Manual. This requires a small change at the beginning of a SunCore applications program where the view surface structure is declared, and may require changes to in the calls to SunCore functions which take a view surface argument. The *get_view_surface* function has been added to SunCore to simplify using the new view surface structure. See Appendix B of the *SunCore Programmer's Reference Manual*.

5.2. Other Changes to SunCore

The COP routines, low-level rasterOp functions for the Sun-1 color display — have been replaced by pixrect routines.

It is no longer possible to stop *initialize_view_surface* clearing the viewsurface.

The *get_mouse_state* function has acquired two new arguments to specify which device class and device number are generating the input.

The *set_zbuffer_cut* function now has an additional argument to specify a view surface.

The *hue* and *style* arguments to the *set_shading_parameters* function are now integers instead of floating-point numbers.

6. Changes in the 1.1 Release of SunWindows

This section describes changes, deletions, and additions to SunWindows from release 1.0 to release 1.1. The main differences in 1.1 center around adding support for multiple screens and color displays.

SunWindows includes sources for window application programs in */usr/suntool/**, the suntool library */usr/lib/libsuntool.a*, the subwindow library */usr/lib/libsunwindow.a*, and the pixrect library */usr/lib/libpixrect.a*. For *suntools*(1) user information, refer to the *User's Manual for the Sun Workstation*. For more detailed SunWindows programming information, refer to the reference section of the *Programmer's Reference Manual for SunWindows* .

This release also includes a *Programmer's Guide for SunWindows*, which provides motivational material and a collection of examples for programmers meeting the windo system for the first time.

6.1. Upgrading from 1.0 to 1.1 SunWindows

1.0 programs must be recompiled to run in 1.1.

6.2. User Interface Differences — Changes to /usr/suntool

6.2.1. Additions to the User Interface

Additions to the user interface are:

suntools

Now takes an extensive argument list to control the environment of the window system. This includes indicating color, which screen, inversion, and so on. See OPTIONS in *suntools*(1) in the *User's Manual for the Sun Workstation* for details.

spheresdemo

Now produces multiple colored spheres. A **–g** command line argument produces varying shades of gray spheres. These grays may not appear gray until the cursor is positioned in the window in which the spheres are being drawn.

jumpdemo

Now produces colored vectors. A **–c** command line argument causes the vectors to sparkle

via colormap rotation.

System Fonts
More fonts are available for use as the DEFAULT_FONT. See */usr/suntool/fixedwidthfonts/** and *suntools*(1) in the *User's Manual for the Sun Workstation*.

Exiting suntools
Typing ^D followed by a ^Q to the Root Window exits *suntools*.

6.2.2. New Programs

adjacentscreens(1)
Tells the window system the physical relationship of screens.

lockscreen(1)
Puts a "lock" on and hides the current window context so logging out is no longer necessary.

perfmon(1)
A graphic performance monitor.

6.3. Suntool Library

The changes to the suntool library involve several changes to the option subwindow interface, making graphics subwindows more robust, and simplifying window management utilities.

6.3.1. Option Subwindow Changes

Three routines now take different arguments, return different values, and/or behave differently than they used to. These are:

optsw_text
Takes a fifth argument, the address of a notify procedure, exactly as for the other item-creation routines. The notify procedure is called whenever the value of the text item is changed, except by a call to *optsw_setvalue*. It will be called with handles for the option subwindow and the item which changed. *Optsw_getvalue* should be used to actually retrieve the new value. This parameter to *optsw_text* may be NULL to indicate "no notification."

optsw_getvalue
Behaves differently for text items; its second (destination) argument should now be a pointer to a struct *string_buf*, as defined in *optionsw.h*. This protects against the case where the value of the item is longer than the client's buffer. In such a case, the buffer is filled, and the max count is returned; no terminating NULL is written in the client's buffer. A subsequent call to *optsw_getvalue* for that item will return the next fragment, until the whole value has been reported. A terminating NULL is written in the buffer when there is room for it, and a subsequent call to *optsw_getvalue* will start anew at the beginning of value.

optsw_setplace
Has had its arguments changed to be parallel with *optsw_getplace*. It third argument is now a pointer to a struct *item_place*, instead of the struct *rect* pointer it used to take; the struct contains a rect, and four boolean bit flags indicating that a value is to be fixed for that item.

6.3.2. Option Subwindow Deletions

The *struct opt_item* is no longer defined in a public header file. Routines which used to return a pointer to such a struct (all of the item-creation routines, for instance) now return an opaque pointer (*caddr_t*). Routines which took a pointer to such a struct as an argument now accept the opaque pointer. Inquiry and manipulation functions are provided to support access to the items without commitment to their internal representation.

6.3.3. Option Subwindow Additions

Two new structs are defined, one for *optsw_getplace* and *optsw_setplace*, one for *optsw_getvalue* on text items:

struct item_place
: Encodes the information about an item's size and location which the client may see and modify. A pointer to such a struct is passed to *optsw_getplace* (which fills it in) and *optsw_setplace*, which uses it to establish an item's location, size, and willingness to change.

struct string_buf
: Provides a counted buffer for text items' values to be stored into. Limit should be the size of the buffer on a call to *optsw_getvalue*.

The following new routines are also provided:

optsw_getcaret(osw)
: Returns an item handle for the item which currently has the caret in *osw*, or NULL if there is no text item in *osw*.

optsw_setcaret(osw, ip)
: Makes the optionsw text item referred to by *ip* be the one which has the caret in the indicated optionsw.

optsw_getfont(osw)
: Returns a pointer to the struct *pixfont* which is currently being used by the optionsw.

optsw_getplace(osw, ip, place)
: Stores into the *item_place* struct pointed to by *place* a description of the size, position, and fixedness of the item indicated.

optsw_nextitem(osw, ip)
: Given an item in an optionsw, returns a handle for the next item in sequence.

optsw_removeitems(osw, ip, count, reformat)
: Removes at most *count* items from *osw*, making them inaccessible to the user, but not destroying them.

optsw_restoreitems(osw, ip, count, reformat)
: Restores at most *count* items in *osw*, starting at the item indicated by *ip*; returns the number restored.

6.3.4. Graphic Subwindow Changes

Graphic subwindow changes to the interface are:

gfxsw_setinputmask
: Should be called instead of *win_setinputmask*. This call takes additional arguments as well.

6.3.5. Graphic Subwindow Deletions

The graphics subwindow procedure *gfxsw_cleanup* was removed from the interface because it is now obsolete due to the new implementation of the graphic subwindows. Graphics subwindows now use blanket windows instead of the old window takeover mechanism. You can instead call *gfxsw_done*, or do nothing at all, from SIGINT and SIGHUP handling routines.

6.3.6. Graphic Subwindow Additions

Graphic subwindow additions to the interface are:

gfxsw_catchsigwinch
Catches and handles SIGWINCH.

gfxsw_catchsigtstp
Catches and handles SIGTSTP.

gfxsw_catchsigcont
Catches and handles SIGCONT.

gfxsw_notusingmouse
May be called if your program doesn't use the mouse; this is optional.

gfxsw_inputinterrupts
A substitute utility for tty process control while using the window input mechanism.

6.3.7. Window Management Deletions

Window management interface deletions include:

Wmgr_changelevelonly
Removed in favor of the similar procedure *wmgr_changelevel.*

Wmgr_changestate
Was removed in favor of the similar procedures *wmgr_open* and *wmgr_close.*

Wmgr_setupmenus
Removed in favor of *wmgr_setupmenu.* The interface no longer supports *wmgr_rootmenu* (moved into client code, see the *suntools.c* source), thus, the change of plurality.

6.3.8. Window Management Additions

Window management additions to the interface are:

wmgr_open, wmgr_close, wmgr_move, wmgr_stretch, wmgr_top, wmgr_bottom, wmgr_refreshwindow Are the highest level window management routines and correspond exactly to tool menu operations.

wmgr_changerect
Provides finer control of moving and stretching user interaction.

wmgr_confirm
A standard confirmation utility.

wmgr_handletoolmenuitem
Switch to call top level window management routines based on *wmgr_toolmenu* menu item chosen.

wmgr_setrectalloc and *wmgr_getrectalloc*
Global storage of next default window position.

6.4. Sunwindow Library

The changes to the suntool library center around keeping up with the pixrect library changes by providing a pixwin operation to match each pixrect operation and cleaning up the interface to screen structures.

6.4.1. Changes to the Interface

The screen struct was completely overhauled to accommodate color and multiple screens. However, the *scr_rect* was left untouched and is the field that high level clients most often use. Therefore, no source changes should probably be required by most programs.

Win_screennew
Now has a different calling sequence. It now takes a struct screen pointer and returns a window file descriptor. It used to take a window file descriptor and and struct screen pointer.

Win_screenpositions
Was renamed *win_setscreenpositions*.

6.4.2. Deletions from the Interface

With the advent of blanket windows, using *win_setowner* to temporarily change ownership of windows is no longer recommended.

6.4.3. Additions to the Interface

The pixwin additions that correspond to the equivalent pixrect additions are:

pw_region
Pixwin region operation.

pw_ttext
Pixwin transparent text operation.

pw_batchrop
Pixwin batchrop operation.

pw_stencil
Pixwin stencil operation.

pw_putattributes and *pw_getattributes*
Pixwin attributes control.

pw_putcolormap and *pw_getcolormap*
Pixwin colormap control.

The pixwin additions that extend pixrect functionality are:

pw_setcmsname and *pw_getcmsname*
Pixwin colormap segment name access.

pw_preparesurface
Pixwin surface preparation (colormap segment related).

pw_cyclecolormap
Pixwin colormap utility.

The screen-related additions are:

win_getscreenpositions
Retrieve neighbors of the window's screen.

win_setkbd and *win_setms*
Change keyboard and mouse devices used by the screen.

win_initscreenfromargv
Standard command line to screen specification parser.

These are the additions related to the new blanket window mechanism:

win_insertblanket
Insert window into display tree and treat as a "blanket" window (one that always covers its parent).

win_removeblanket
Remove blanket window from display tree.

win_isblanket
Check 'is a window a blanket window?'

6.5. Pixrect Library

Pixrects features a slightly modified interface to support color pixrects. Also, the font format has changed.

6.5.1. Changes to the Interface

A new frame buffer naming convention now exists:

/dev/fb
The default frame buffer for a machine. This replaces */dev/console* which has other tty-related functions, and */dev/bw0* which no longer exists.

Frame Buffer Naming Convention
The general naming convention for a frame buffer follows the form */dev/CTU* in which:

C is either "bw" (for monochrome displays) or "cg" (for color displays).

T is the type of the display, such as "one" or "two" for Sun-1 or Sun-2 respectively.

U is the unit number starting from 0, indicating which specific frame buffer.

Some examples of frame buffer names are: */dev/bwone0*, */dev/bwtwo0* and */dev/cgone0*.

The font format used in *pf_open* has been changed. The old format was in VAX byte order. The new format is in Motorola 68000 byte order (reversed from the VAX). You can tell if a font is in the new format by using the *file*(1) program on the font file in question. The font file should be listed as "vfont definition". The program *vswap*(1) converts a font file from the old format to the new.

The format of the *mpr_data* data structure (the internal memory pixrect data format) has changed slightly. The **int** *md_primary* field has been split into **short** *md_primary* and **short**

md_flags. The overall length of ***mpr_data*** remains the same.

6.5.2. Additions to the Interface

pr_stencil
Provides spatial masking of the destination pixrect for control of the areas of the destination pixrect to be painted by the source pixrect.

pr_putcolormap and ***pr_getcolormap***
Provides a unified colormap and reversevideo interface for both color and monochrome pixrects.

pr_putattributes and ***pr_getattributes***
Provides access to a bitplane mask which specifies the modifiable bits in destination pixrect pixels.

pf_ttext
Uses character bitmap as a stencil through which the specified color is squirted, hence background shows through around the characters.

7. Utility Packages

7.1. New MS Macro Package

A new version of the –ms macro package exists. This new version fixes several bugs in the old –ms macro package and adds some new features. See *Editing and Text Processing on the Sun Workstation*. The old version of the –ms macro package is still available as –msold.

8. Documentation

Here are the major highlights of the documentation changes for this release:

8.1. System Manager's Manuals

System Manager's Manual

- Model 170 setup and installation guide and hardware information.
- Setup notes for the Sun-2 Ethernet Board.
- Notes on installing expansion disk subsystems.
- Notes on installing USENET.
- Revised installation procedures for class C network addressing.

8.2. Beginner's Guide to the Sun Workstation

New section
on how to use SunWindows — the window-based user interface.

New descriptions
on *mail* and *news*.

New quick reference pages
for *mail* and *news*.

8.3. Editing and Text Processing on the Sun Workstation

A description of the new version of the **–ms** macro package has been folded into the paper on using the **–ms** macro package for formatting documents. Much new explanatory material on paragraph styles was added to this document and some typographical terms were explained better.

8.4. Sun Windows Manuals

- Additions, changes, and deletions to user interface, option subwindow, graphic subwindow, and window manager.
- Changes to subwindow library to accommodate color and multiple screens.
- Changes to the pixrect library to support color pixrects.
- Included new "Changes" supplement.

SunWindows Programmer's Guide — An Introduction to the Sun Window System
A new tutorial guide for programming tools and applications in SunWindows.

8.5. SunCore Programmer's Reference Manual

Here are the major changes to the **SunCore** Programmer's Reference Manual for release 1.1:

Pascal Language Interfaces
to the **SunCore** library are described in a new appendix.

8.6. Programming Tools for the Sun Workstation

In *Programming Tools for the Sun Workstation* there is a new chapter on *Programming the Shells*. This information got dropped out of previous releases of the Sun documentation.

Part II — Changes From Release 0.4 to Release 1.0

This part of the document contains a description of changes to the system between Sun System Release 0.4 (May 1983) and Sun System Release 1.0 (November 1983).

9. Changes to the Operating System Software

Release 1.0 of the Sun system was based on a preliminary version of the final Berkeley Software Distribution for the VAX.

9.1. CPU PROMS

The Revision L PROMS for the Model 100 had a bug which stopped the 4014 emulation from working.

9.2. New Software Drivers

- Software driver for the Xylogics-450 SMD disk controller.

10. Programming Language Support in Release 1.0

10.1. DBX — Symbolic Debug Package

The *dbx* symbolic debug package was added to the 1.0 release. At that time, *dbx* did not support FORTRAN programs.

10.2. Changes to the C Compiler

Floating Point
: support was added via the **-fsky** option to compile code for the SKY floating-point board, and the **-fsingle** option to use single-precision arithmetic in computations involving **float** numbers.

Interface
: between compiled programs and the floating-point library was changed.

Support for dbx
was added in release 1.0. In addition, the **–G** option was changed to the **–g** (lower-case g) option to generate symbols for *dbx*, and the old **–g** option was changed to the **–go** (old g option) to generate more symbols for *adb* (actually for *sdb*).

Alternate Compiler Phases
when using the **–t** option without the **–B** option: the alternate path prefix was */usr/new* but it was documented as */usr/c*. The alternate path is now */usr/new/*.

11. Changes to the SunCore Graphics Package

This section provides guidelines for the efficient conversion of existing **SunCore** application programs to the Sun 1.0 software release.

The Sun implementation of the ACM CORE specification has not changed significantly between release 0.4 and release 1.0. Most application programs should prove fairly simple to convert. The difficulty of converting an existing program depends largely on how much the program depends on specific hardware features. Dependencies tend to center around the Sun-1 frame buffer and direct use of the bitmap and RasterOp capabilities.

11.1. Changes in SunCore for Sun 1.0

The following are the major changes in the **SunCore** graphics package between release 0.4 and release 1.0:

- **SunCore** view surface names have been changed.
- *await_keyboard* returns with the *input_string* null- terminated "after" the newline character instead of before the newline character, so you can distinguish between a timeout and a newline-terminated string.
- Bitmap framebuffer RasterOp functions have been replaced by pixrect operations.

11.2. View Surface Names

The 1.0 SunCore implements a more uniform surface naming scheme than the older versions of SunCore. Black and white devices are referred to as *bw***X**, where **X** is the level of the board used. Color devices are similarly *cg***X**.

Currently there are Sun-1 and Sun-2 black and white surfaces, in addition to a black and white window surface, and a Sun-1 color surface available. View surface names are as follows:

Table 1: SunCore View Surface Names

View Surface	*SurfaceName*
Sun-1 Black and White	bw1dd
Sun-2 Black and White	bw2dd
Sun-1 Color	cg1dd
Window Black and White	pixwindd

The only change needed to convert device-independent **SunCore** programs to run under release 1.0 is an alteration in the view surface names.

11.3. Determining the Correct View Surface

A **SunCore** program which runs on the monochrome display may write to the raw screen or to a window. To help a program determine which black and white view surface to initialize, the WINDOW_ME environment variable may be examined. WINDOW_ME is normally defined only if the window system is being used. The following code is a sample routine for determining a specific view surface to be initialized:

```
int pixwindd();
int bw1dd();
static int (*vwsrfdd)();

if (getenv( "WINDOW_ME")) vwsrfdd = pixwindd;
        else vwsrfdd = bw1dd;    /* bw2dd for Sun 2 video */

if (initialize_view_surface( vwsrfdd, FALSE)) exit(0);
```

The correct way to determine which video board is being used is through the *ioctl* call FBIOGTYPE, which is defined in */usr/include/sun/fbio.h*.

11.4. Low-Level RasterOp Functions

In SunCore releases prior to release 1.0, SunCore provided support for low level RasterOps for the Sun-1 bitmap display via a group of ROP*zzz* functions documented in appendix B.1 of the SunCore manual. These ROP functions have been removed from **SunCore**. A new set of low-level device-independent routines called *pixrects* are now provided in a separate library. A full explanation of pixrect-level programming can be found in the *Programmer's Reference Manual for the Sun Window System.*

11.4.1. Correspondence of ROP Functions and Pixrect Functions

The following table provides a rough correspondence between ROPs and *pixrects* to help programmers convert existing code to the pixrect support. The ROP functions on the left should be replaced by the pixrect function on the right. The user of pixrect functions must include *.h* files from */usr/include/pixrect*.

Table 2: Correspondence Between ROP Functions and Pixrect Functions

ROP function	Pixrect function	ROP function	Pixrect function
_core_GXopen	pr_open	ROPssdr	pr_rop
_core_GXclose	pr_close	ROPssdl	pr_rop
ROPssur	pr_rop	ROPms	pr_rop
ROPssul	pr_rop	ROPdminv	pr_rop
ROPds	pr_rop	ROPdmclr	pr_rop
ROPcopysm	pr_rop	ROPvector	pr_vector
ROPdmset	pr_rop	ROPtext	pf_text pf_open pf_close pf_default

The *pr_rop* function can deduce from its arguments whether the source and/or destination are memory pixrects or display pixrects. It also knows which direction it must read a raster in order to copy from one place to another in a pixrect without overwriting the source. The RasterOp function itself — that is, XOR, OR, and so on, is an argument to the pixrect functions rather than a separate operation. Hence it is no longer necessary (or possible) to set the RasterOp via GXfunction = .

Rote conversion of ROP-based programs to pixrect-based programs is usually not very difficult, but only provides a level of functionality equal to that available under previous releases of the operating system. With a little extra work it is normally possible to use also the extensive device-independent graphics interface and the window manager provided by SunWindows to greatly extend the capabilities of the program. In addition, performance considerations may dictate some rethinking of programs, or at least reordering and regrouping operations, to take advantage of performance features in the new software.

11.5. Re-Compiling

Because of the device-independent nature of SunCore, converting SunCore programs involves only changing the names of the view surfaces, and then recompiling. Obviously, changes in sizes of view surfaces, changing from Sun-1 video to Sun-2 video for example, will affect programs that are not size-independent.

12. LEGS Graphics Package

The LEGS library is no longer provided as part of the Sun software release. LEGS programs should be converted to use either the **SunCore** or pixrects libraries. Programs which require device-independence and high-level graphics primitives should be converted to **SunCore**. Programs which require high speed, use 2-D device (integer) coordinates, and are restricted to vectors, rasters, and text, may be converted to pixrects (see the *Programmer's Reference Manual*

for the Sun Window System).

12.1. Device Independence

LEGS programs which are converted to **SunCore** will take advantage of the **SunCore's** device-independence by using the appropriate viewing parameters and directing output to a desired view surface. Pixrects provide some device-independence for display-oriented programs by using (0,0)-based coordinates, and operating transparently on the various display devices and memory. Programs desiring to run only on the raw display via pixrects can use the function *pr_open* on the defined device */dev/bw0*, which canonically represents the entire black and white display screen.

12.2. Specification of RasterOP Function in Pixrect Operations

Rather than having to set a single global variable for the type of RasterOp to be performed on the video buffer, pixrect calls have a RasterOp argument. While this may require the addition of parameters to some of the calls used in LEGS programs, the ease of use gained as a result should be worth it.

It is worth noting here that PIX_CLR is the equivalent of GXclear, and PIX_SET is the equivalent of GXset. In addition, functions such as *ras_write* translate almost directly into *pr_rop* calls.

12.3. Conversion of Offscreen Areas to Memory Pixrects

Many LEGS programs used the "offscreen" area of the video buffer to store unused images, or as sort of a "staging" area for images being prepared. These should be replaced by *memory pixrects*, which are not displayed, and can easily be copied onto a display area when needed.

12.4. Performance Considerations

A naive conversion of code from LEGS to *pixrects* can result in a loss of performance, but through some restructuring of the code to take advantage of the pixrect facilities, much of this loss can be regained. In addition, through judicious use of the PIX_DONTCLIP flag, some of the clipping overhead of the pixrect system can be avoided. While not directly applicable to *pixrect* applications, if applications take advantage of the *sunwindow* layer, locking is a good method for eliminating much of the overhead for *pixwin*-based programs running in SunWindows.

12.4.1. Converting ROP Functions to Pixrect Functions

The following table provides a rough correspondence between ROPs and *pixrects* to help programmers convert existing code to the pixrect support. The ROP functions on the left should be replaced by the pixrect function on the right. The user of pixrect functions must include *.h* files from */usr/include/pixrect*.

Table 3: Converting ROP Functions to Pixrect Functions

ROP function	Pixrect function	ROP function	Pixrect function
_core_GXopen	pr_open	ROPssdr	pr_rop
_core_GXclose	pr_close	ROPssdl	pr_rop
ROPssur	pr_rop	ROPms	pr_rop
ROPssul	pr_rop	ROPdminv	pr_rop
ROPds	pr_rop	ROPdmclr	pr_rop
ROPcopysm	pr_rop	ROPvector	pr_vector
ROPdmset	pr_rop	ROPtext	pf_text pf_open pf_close pf_default

The *pr_rop* function can deduce from its arguments whether the source and/or destination are memory pixrects or display pixrects. It also knows which direction it must read a raster in order to copy from one place to another in a pixrect without overwriting the source. The RasterOp function itself — that is, XOR, OR, and so on, is an argument to the pixrect functions rather than a separate operation. Hence it is no longer necessary (or possible) to set the RasterOp via GXfunction = .

Rote conversion of ROP-based programs to pixrect-based programs is usually not very difficult, but only provides a level of functionality equal to that available under previous releases of the operating system. With a little extra work it is normally possible to use also the extensive device-independent graphics interface and the window manager provided by SunWindows to greatly extend the capabilities of the program. In addition, performance considerations may dictate some rethinking of programs, or at least reordering and regrouping operations, to take advantage of performance features in the new software.

13. Documentation

These were the major highlights of the documentation changes between Release 0.4 and Release 1.0:

13.1. Beginner's Guide to the Sun Workstation

Rewritten for technical accuracy and usability.

13.2. Editing and Text Processing on the Sun Workstation

Rewritten for technical accuracy and usability.
New introductory material on using text editors and preparing documents.

13.3. Programmer's Reference Manual for SunWindows

Additions to pixrect creation, input handling, and tool facilities

Part III — Differences Between Sun Release 1.1 and Berkeley 4.2 BSD

Although the Sun System evolved from Berkeley 4.2, there are differences between the two systems as they diverge from each other. Some of the differences are the inevitable result of differences between the VAX and the Sun hardware. Other differences arise as Sun people fix bugs, enhance utilities, and redo the documentation for a different market. This part of the document contains a description of differences between the Sun Release 1.1 and the Berkeley 4.2 release.

14. New, Changed, and Deleted Utilities

14.1. New Utilities

The Sun system has the *suntools* window system for the benefit of users.

14.2. Changed Utilities

An **ls -l** command identifies a UNIX AF_DOMAIN socket with the letter 's', whereas an **ls -F** command identifies a UNIX AF_DOMAIN socket with a trailing '=' sign.

14.3. Deleted Utilities

User Contributed Software
portion of the 4.2 BSD release is not present in the Sun system.

efl — Extended FORTRAN Language — doesn't exist.

struct — turn FORTRAN programs into *Ratfor* programs — doesn't exist.

fed — font editor — command doesn't exist.

quotas command doesn't exist because the Sun system does not support disk quotas.

apply command was removed from the system in 1983. There is no replacement for *apply* other than writing a shell script.

sdb — symbolic debugger — is replaced by *dbx*(1).

style command was deleted because *style* is now part of 'Writer's Workbench' which requires a supplementary license.

diction	command was deleted because *diction* is now part of 'Writer's Workbench' which requires a supplementary license.
explain	command was deleted because *explain* is now part of 'Writer's Workbench' which requires a supplementary license.
learn	command is so far out of date and inappropriate for the Sun systen that we do not ship it.
apropos	was deleted — the same effect can be achieved with a **man -k** command.
xref	was deleted — the same effect can be achieved with a **ctags -x** command.
num	was deleted — the same effect can be achieved with a **cat -n** command.
print	was deleted — the same effect can be achieved with a **lpr -p** command.
see	was deleted — the same effect can be achieved with a **cat -v** command.
tra	was deleted — the same effect can be achieved with a **tail -f** command.
finger	command was deleted from the Sun system because it is very specific to the campus environment at Berkeley.

15. Changes to Documentation

The documentation is where things are changing the most. A major reason for the massive changes is that Sun's documentation is targeted towards a market consisting of people who are not only not UNIX experts, but also people who can't look at the source code as the final arbitration of how something should work. For the benefit of customers who haven't had our standard spiel on the documentation, here are some of the details.

The Sun System documentation has gone through major changes, and we thought we'd give you a brief summary of what's gone on and how this has affected the manuals you'll be seeing. Basically, we've tried to improve the utility of the existing material by radically re-organizing and updating it. We've added new material to try to keep up with development, and to remedy the fact that many aspects of the existing system were simply never described. In what follows, we'll begin with history and go on to current organization and contents.

15.1. Original Organization

In the beginning, all the UNIX documentation was in four volumes known as *The UNIX Programmer's Manual* Volumes 1, 2a, 2b, and 2c.

Volume 1 contained eight sections. Only sections 1, 6, and 7 were really relevant to users — the rest was relevant to programmers writing code, or system managers looking after the system. Volume 1 was divided into sections like this: 1: Commands — descriptions of the commands that a UNIX user uses on a daily basis; 2: System Calls — interfaces from running programs to the kernel; 3: Subroutines — access to the input-output system, the network library, the math library, the FORTRAN library, and miscellaneous libraries such as jobs and curses; 4: Device Descriptions — descriptions of the 'special files' that represent the interface from the device to the device driver; 5: File Formats — format of files such as *a.out*, */etc/hosts*, and so on; 6: Games; 7: Tables — descriptions of manuscript-formatting macros (like **-ms**), maps of certain character codes (like the ASCII character set), and a diagram of the file system hierarchy; 8: Maintenance Commands and Procedures — describing the utilities that the super-user normally uses to keep the system running.

Volume 2 (a, b, and c) contained various tutorial-style papers on various UNIX utilities: the text-editor, the mail system, programming tools, and so on.

It has always been obvious that the above organization simply wasn't working very well — Volume 1 was too big to fit in a single three-inch binder; Volume 2 was a random collection of material that was often out of date or irrelevant.

15.2. New Organization

Sun has 'unbundled' the whole thing. Instead of four huge three-ring binders, there are now twelve smaller ones. These new manuals are focused on functional application areas — you should be able to find what you need without leafing through reams of irrelevant paper. They have index tabs between the sections.

15.2.1. Reference Manuals

The original Volume 1 was broken out into three reference manuals, one for users, one for programmers, and one for system managers:

User's Manual
covers the facilities available to a Sun Workstation user (Section 1). It includes sections 1 (Commands), 6 (Games), and 7 (Tables) of the original Volume 1.

System Interface Manual
covers areas of interest to programmers. It contains sections 2 (System Calls), 3 (Library Functions), 4 (Special Files), and 5 (File Formats). The old section 3 is divided by functional area.

System Manager's Manual
contains all the information needed to install and configure the Sun Workstation, install the UNIX system, and run the UNIX system on a day-to-day basis. The System Manager's Manual includes the original section 8 (Maintenance Commands and Procedures). It also contains information on how to set up the mail system and networking facilities.

15.2.2. Supplementary Documentation

The original Volume 2 (2a, 2b, and 2c) has met with a similar fate. The manuals as they currently stand are:

Beginner's Guide to the Sun Workstation
A general overview and introduction to UNIX providing roadmaps and guides to the rest of the documentation, plus some tutorial introductions to the Sun system. It also contains a general guide to the communication facilities that UNIX offers you.

Editing and Text Processing on the Sun Workstation
documents the facilities that UNIX offers for massaging text files in various ways, and for processing documentation. Introduces the basic document layout facilities.

Programming Tools for the Sun Workstation
contains information of general interest to anyone using UNIX to write programs. Contains UNIX Programming, Lint, Make, SCCS, DC, BC, M4 Macro Processor, LEX, YACC, Assembler Manual.

SunCore Programmer's Reference Manual
is a complete reference description of Sun Microsystems' implementation of the ACM Core graphics package.

SunWindows Programmer's Reference Manual
is a complete reference description for programmers wishing to write tools to run in the Sun window system.

SunWindows Programmer's Guide
is a tutorial guide to writing tools for the Sun window system.

System Internals Manual
contains information relevant to highly-skilled programmers wishing to work closely with the system kernel. This manual contains papers on the network interfaces and protocols, adding device drivers to the system, and tuning the system.

FORTRAN and Pascal
Information specific to the FORTRAN and Pascal programming languages. Contains Fortran Reference Manual, Ratfor, Fortran System Interface Routines, Pascal User's Manual.

15.3. Manual Pages that were Moved

Several manual pages were moved out of section 1 into section 8 because it was felt that the everyday user wouldn't care about such esoterica and we didn't want the 'man' pages for such commands cluttering up the section 1 commands section. These commands include:

netstat show network status.

newaliases rebuild mail aliases database.

vmstat report virtual memory statistics.