Introduction

The 1.4 release fixes various bugs and adds support for the Sun-2/160 workstation, otherwise it is compatible with prior releases in the 1.X software series. Existing 1.1 documentation applies to 1.4 except where superseded by the *Release 1.4 Manual*.

Release 1.4 is available in two forms, as a full release tape called 1.4, and as an update release tape called 1.4U. The 1.4U form of the release updates systems running the 1.1, 1.2, or 1.3 releases, bringing them up to the equivalent of the 1.4 full release. 1.4U does not update systems running 0.3, 0.4, or 1.0 releases (these releases are no longer supported).

The full release version of 1.4 is shipped with new workstations, and the update version, 1.4U, is shipped to customers with support agreements covering software. The 1.4U tape can also be ordered from Sun for $250 per licensed workstation. Call Connie Humphries, (415) 960–2764, for information.

Should you have software or hardware questions while installing 1.4, call (415) 960–3500. Have your system's model and serial numbers ready to give the dispatcher. If you have questions about your shipment (e.g., missing items), call your sales representative. Software bugs can be reported by sending a description electronically to sun/sunbugs, or by U.S. Mail to Technical Support MS–2/30, Sun Microsystems, 2550 Garcia Ave, Min View, CA, 94043.

New Workstations

Install the 1.4 full release according to the instructions in the *Release 1.4 Manual*, being careful to note the changes documented in the 1.4 errata pages.

Existing Workstations

On workstations running 1.1, 1.2 or 1.3, begin installing 1.4U by reading chapter 1 of the *Release 1.4 Manual* and proceeding with the installation steps in chapter 2. There is no need to install 1.2 or 1.3 before 1.4U (1.4U is a cumulative update).

If your workstation IS NOT running 1.1, 1.2 or 1.3, call Technical Support, (415) 960–3500, for additional information about installing 1.4.
Known Problems in 1.4

The following are some of the known problems in Release 1.4.

Determining the type of 85MB disk in a workstation.

On systems with 85MB disks, before running diag, you need to know the type of 85MB disk drive. The type of 85MB drive (Fujitsu, Micropolis, or Vertex) will be shown on a label behind the front cover of the machine. If no label is present, assume the disk is a Fujitsu.

Compatibility of 20MB and 45MB tapes.

Tapes written on the 20MB and 45MB tape drives are not guaranteed to be interchangeable. Generally, tapes written on 20MB drives can be read on 45MB drive when the tape is correctly tensioned. There is no assurance that tapes written on 45MB cartridge drives will read on 20MB drives.

To retension tapes, use the mt ret command. Alternatively, on machines with Rev Q or later proms, the b (boot) command will first retension a tape when 100 is added to the tape file number. For example, use b st(0,0,100) to retension the tape and boot file 0 from the st drive.

The Sun-2/50 and Sun-2/160 are more sensitive to Ethernet™ problems.

The Sun-2/50 and Sun-2/160 are designed to be used with level 1 Ethernet transceivers, like the 3Com™ transceivers supplied by Sun. Level 2 transceivers, like the DEC™ DELNI, may work when the cpu board jumper located next to the 24-pin MB502 chip by the transceiver cable connector is removed. Sun, however, does not support this.

usercore77.h misnamed as f77.

This affects the compilation of Fortran programs using SunCore™. Correct this with the command, mv /usr/include/f77 /usr/include/usercore77.h.

Corrections to the Release 1.4 Manual.

Systems with 1/4" should use the mt -f /dev/nrtape0 rew command before running 1.4_upgrade (page 2-5).

On page 2-7, the line tar xvpf /dev/nrtape0 { man | demo } should read tar xvpf /dev/nrtape0 { ./man | ./demo }.

If you are using 1/4" tape, skip section 2.6 and use appendix A for loading optional software.


A 'ghost' image of the window system may be left behind when SunWindows is exited. Fix this with the clear -c command. See the clear manual page in the Release 1.4 Manual.

zs serial port problems.

Various problems exist with the zs serial port driver.

Degaussing color monitors.

Keep magnetic tapes away from color monitors, particularly during degaussing.

Sun Workstation is a registered trademark of Sun Microsystems, Inc.
Sun-2, Sun-2/xxxx, SunCore, SunWindows, and Sun Microsystems are trademarks of Sun Microsystems, Inc.
Unix is a trademark of AT&T Bell Laboratories.
Ethernet is a trademark of Xerox Corporation.
3Com is a trademark of 3Com Corporation.
DEC is a trademark of Digital Equipment Corporation.
Credits and trademarks

Sun Workstation, Sun-1, and Sun-2 are trademarks of Sun Microsystems, Incorporated.

UNIX is a trademark of AT&T Bell Laboratories.

Copyright © 1984, 1985 by Sun Microsystems, Inc.

This publication is protected by Federal Copyright Law, with all rights reserved. No part of this publication may be reproduced, stored in a retrieval system, translated, transcribed, or transmitted, in any form, or by any means manual, electric, electronic, electro-magnetic, mechanical, chemical, optical, or otherwise, without prior explicit written permission from Sun Microsystems.
Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5th November 1984</td>
<td>First release of this 1.4 Manual.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added one erratum and notes on new version of <em>tip</em>.</td>
</tr>
<tr>
<td>B</td>
<td>22nd January 1985</td>
<td>Added new <em>diag</em> chapter, additional new man pages,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bug fixes for GENERIC, notes on new tape reorganization and new</td>
</tr>
<tr>
<td></td>
<td></td>
<td>installation procedures for optional software.</td>
</tr>
<tr>
<td>C</td>
<td>7th February 1985</td>
<td></td>
</tr>
</tbody>
</table>
Contents

Chapter 1 Introduction ................................................................. 1-1
Chapter 2 Installing the 1.4 Release .............................................. 2-1
Chapter 3 Using the New diag to Format and Label Disks .................. 3-1
Chapter 4 Changes from the 1.1 Release Software .......................... 4-1
Chapter 5 Changes from the 1.2 Release Software .......................... 5-1
Chapter 6 Changes from the 1.3 Release Software .......................... 6-1
Chapter 7 Errata Pages for 1.1 Manuals ...................................... 7-1
Chapter 8 Insert Pages for 1.1 Reference Manuals ......................... 8-1
Appendix A Distribution Tape Reorganization ............................... A-1
 Contents

Chapter 1 Introduction ....................................................... 1-1
  1.1. Summary of Contents .............................................. 1-1

Chapter 2 Installing the 1.4 Release ................................... 2-1
  2.1. What is on the Distribution Tape? ............................. 2-2
  2.2. Overview of the Installation Procedure ....................... 2-3
  2.3. Load the Tape .................................................... 2-4
  2.4. Extract 1.4 Installation Utility from Tape .................. 2-4
  2.5. Run the 1.4 Installation Utility ............................... 2-5
  2.6. Loading Optional Software from the Release Tape .......... 2-6
  2.7. Reconfigure your UNIX System Kernel .......................... 2-7
    2.7.1. Kernel Reconfiguration — an Annotated Copy of GENERIC 2-8
    2.7.2. Kernel Reconfiguration for Standalone Systems ........ 2-11
    2.7.3. Kernel Reconfiguration for Servers ....................... 2-13
  2.8. ‘Uninstalling’ the 1.4 Release .................................. 2-14

Chapter 3 Using the New diag to Format and Label Disks ........... 3-1
  3.1. Phase One: Specifying Hardware Configuration............... 3-1
  3.2. Phase Two: Formatting and Labelling the Disk ............... 3-3
    3.2.1. Formatting SCSI Disks .................................... 3-3
    3.2.2. Labelling SCSI Disks ..................................... 3-5
    3.2.3. Formatting SMD Disks .................................... 3-6
    3.2.4. Labelling SMD Disks ..................................... 3-7

Chapter 4 Changes from the 1.1 Release Software .................... 4-1
  4.1. Software Contents and Details of Changes ................... 4-1
    4.1.1. Window System ............................................ 4-1
    4.1.2. Assembler ............................................... 4-1
    4.1.3. C Compiler .............................................. 4-2
    4.1.4. FORTRAN Compiler ....................................... 4-3
    4.1.5. Diag .................................................... 4-3
    4.1.6. Sky Driver ............................................. 4-3
    4.1.7. SCSI Drivers ........................................... 4-3
    4.1.8. UART Driver (zs) ....................................... 4-4
    4.1.9. Sky Microcode .......................................... 4-4
    4.1.10. VpItdmp ............................................... 4-4
  4.2. Documentation Changes ......................................... 4-4
4.2.1. Diag ................................................................. 4-4
4.2.2. Sendmail Installation .......................................... 4-4
4.2.3. zsh(4) .............................................................. 4-4
4.2.4. /etc/skynit ....................................................... 4-5
4.2.5. Kernel Configuration .......................................... 4-5
4.2.6. Miscellaneous Errata in the System Manager’s Manual ..... 4-5
4.2.7. Miscellaneous Manual Pages ............................... 4-5

Chapter 5 Changes from the 1.2 Release Software ............... 5-1
  5.1. Software Contents and Details of Changes ................ 5-1
      5.1.1. Model 50 Kernel Support .............................. 5-1
      5.1.2. PROMs for the Sun-2/50 ............................... 5-1
  5.2. Documentation Changes ...................................... 5-1
      5.2.1. Sun-2/50 Hardware Manual ......................... 5-1

Chapter 6 Changes from the 1.3 Release Software ............... 6-1
  6.1. Software Contents and Details of Changes ............... 6-1
      6.1.1. Model 160 Kernel Support ............................ 6-1
      6.1.2. New Version of diag ................................ 6-1
      6.1.3. New Version of tip ................................... 6-1
      6.1.4. Hayes Support for tip and uucp .................... 6-2
      6.1.5. Tape Retensioning ................................... 6-3
      6.1.6. MAKEDEV Bug Fix ................................... 6-3
      6.1.7. Sky Board Error Messages ........................... 6-3
      6.1.8. Changes to the SunCore Graphics Package .......... 6-4
          6.1.8.1. SunCore — Added set_pick Function .......... 6-4
          6.1.8.2. SunCore — Additional Raster Fonts .......... 6-4
          6.1.8.3. SunCore — Sun/2-160 Support ................ 6-4
          6.1.8.4. SunCore — Speed Improvements ............... 6-4
          6.1.8.5. SunCore — Bug Fixes .......................... 6-5
      6.1.9. Changes to SunWindows .............................. 6-5
  6.2. Documentation Changes ..................................... 6-5
      6.2.2. Miscellaneous Manual Pages ....................... 6-6

Chapter 7 Errata Pages for 1.1 Manuals .......................... 7-1

Chapter 8 Insert Pages for 1.1 Reference Manuals .............. 8-1

Appendix A Distribution Tape Reorganization ................. A-1
  A.1. Loading Optional Software ................................. A-2
Chapter 1

Introduction

This document describes the content of the Sun UNIX 1.4 incremental software release; gives instructions for installing the 1.4U release tape on systems currently running the Sun UNIX Version 1.1, 1.2, or 1.3 releases; discusses, briefly, the major software changes between each of these releases (major bug fixes and enhancements); and, finally, includes documentation — errata pages and new insert pages for your 1.1 manuals — which reflects these major changes.

1.1. Summary of Contents

Chapter 1: Introduction
You’re reading it.

Chapter 2: Installing the 1.4 Release
Provides instructions for installing 1.4 on systems currently running the Sun 1.1, 1.2, or 1.3 Release software.

Chapter 3: Using the New diag to Format and Label Disks
Gives instructions for use of the new version of diag, the disk formatting and labeling utility, which is part of the 1.4 Release distribution. This new version supports several new SCSI 85 MByte drives (the Micropolis 1325, the Fujitsu M2243AS, and the Vertex V185), as well as providing enhanced diagnostic capabilities.

Chapter 4: Changes from the 1.1 Release Software
Discusses differences between Releases 1.1 and 1.2: fixes and enhancements. This chapter is identical to Chapter 4 of the 1.2 and 1.3 Manuals.

Chapter 5: Changes from the 1.2 Release Software
Discusses differences between Releases 1.2 and 1.3. This chapter is identical to Chapter 5 of the 1.3 Manual.

Chapter 6: Changes from the 1.3 Release Software
Discusses differences between Releases 1.3 and 1.4.

Chapter 7: Errata Pages for 1.1 System Manager’s Manuals
Contains errata pages for the Sun System Manager’s Manual for the Sun Workstation: Models 100U/150U and the System Manager’s Manual for the Sun Workstation: Models 180/170. Please read the pages from the manual(s) relevant to your system configuration, and note the errors in your 1.1 Manuals.

Chapter 8: Insert Pages for 1.1 Reference Manuals
Contains a few new pages which you can (copy if necessary and) insert into your 1.1 Sun Manuals.

Several of these pages were released with 1.2: as(1), cc(1), cpp(1), screendump(1), screenload(1), malloc(3), za(4S), acct(5), utmp(5), skversion(8), and dinfo(8). These pages either describe new facilities, have been revised, or were mistakenly omitted from the 1.1 manuals.
Several pages are new with this release: \texttt{bessel}(3F), \texttt{bit}(3F), \texttt{time}(3F), \texttt{rand}(3F), \texttt{ctime}(3F), \texttt{fdate}(3F), \texttt{stat}(3F), and \texttt{long}(3F). These pages were omitted from the FORTRAN Library Functions Section of the 1.1 \textit{FORTRAN and Pascal Manual}. A new \texttt{mem}(4S) page is included which describes changes made for Sun-2/160 support. New pages for \texttt{cgtwo}(4S), \texttt{clear}(1), \texttt{diag}(8S), \texttt{ic}(4S), \texttt{ftpusers}(5), \texttt{mt}(1), and \texttt{tip}(1C) reflecting software enhancements are also included.
Chapter 2

Installing the 1.4 Release

In this chapter, we give directions for installing the Sun 1.4U incremental release software on a Sun system currently running the 1.1, 1.2, or 1.3 Releases.

PLEASE NOTE: If you are installing a complete 1.4 distribution rather than an upgrade, do not use the instructions in this chapter for installation. You should, in general, follow the instructions given in the Installing UNIX for the First Time chapter of the 1.1 Release System Manager's Manual for the Sun Workstation. You should have received this manual with your distribution tape(s). Exceptions to the procedures given in that manual are:

1) You have no doubt noticed, if you have opened the manual, that the distribution has grown from two to three 1/4" tape cartridges. This affects only one part of the installation procedure: loading optional software (the online manual pages, demonstration programs, games, and vfont files) to your disk. Appendix A of this manual gives an outline of what is on your tapes, and new procedures for loading optional software.

2) Use the instructions in Chapter 3 of this manual, Using the New diag to Format and Label Disks, to prepare any new disks.

3) Use the procedures in the final sections of this chapter for kernel configuration (since they are more accurate than the description given in the 1.1 Manual because of updates made to the GENERIC system configuration file for Sun-2/160 support).

4) Finally, read through the chapters of this manual describing errata, since these will affect you.

The directions below assume you are working with the incremental release software on either one 1/4" or 1/2" tape, and support installation on

- Standalone machines with local tape and disk, which can read in the distribution tape via the local tape drive;
- Machines with a local disk, but no local tape drive, that are on a network. Such machines will use the tape drive on another machine (called remote_host or server_name in the procedures) to read the tape;
- Server machines with local tape and disk and with diskless clients, which will use the local drive to install both the server and its clients.
Before beginning installation, there are several important things you should be aware of:

- You must have at least 300 KBytes of disk space available on your root partition, and 800 KBytes available on your /usr partition to do this installation. If you wish to load the optional software included on the tape (manual pages and demonstration executables and source), allow another 4.2 MBytes on /usr (100KBytes for the pages, and 4.1 MBytes for the demos).

You can use the df(1) command to check your available space. For example:

```
cronin% df
Filesystem       kbytes  used  avail capacity Mounted on
/dev/xy0a        7437    5470   1223    82%   /
/dev/xy0b        148455 128709  4900    96%  /usr
/dev/xy0g        117327  66896  38698   63%  /usr/misc
[and so on]
```

- If you are installing a new disk, you must follow the directions in Chapter 3 of this document for formatting and labelling it.

- This maintenance release is for installation only on systems running the Sun Version 1.1, 1.2, or 1.3 Release software. This release is incremental in the sense that you need not reinstall the complete operating system. We strongly recommend at least scanning Chapters 4, 5, and 6, which describe deltas in the software, before installing this release, to see how it will affect your Sun Workstation(s) and your software.

- This release is intended for installation as a package; you must install the entire release. Sun will not provide direct support for users who wish to install selected portions of the release software.

- You can 'de-install' this release if you have to: a facility has been provided with the release for backing out changes. In most circumstances, you should not find this necessary; however, if you do, please inform Sun Microsystems Technical Support — we'd like to know what went wrong.

- You must build and install a new operating system kernel to complete the installation of the 1.4 release.

- We strongly advise a thorough back-up before beginning installation.

2.1. What is on the Distribution Tape?

Distribution of the 1.4U Release binaries is either on a 1/4" magnetic tape cartridge or a 1/2" nine-track tape. The tapes contain eight files, as follows:

File 1: Boot block.
A general-purpose boot program which knows how to boot from the various devices that can be attached to the Sun Workstation. The PROM monitor boots this general-purpose boot program.

File 2: Bootable diag program.
This is a new version of diag, the disk formatting and labelling program; this version has enhanced diagnostic capabilities and supports a wider range of disks. For more information on changes in this version, see Chapter 3, Using the New diag to Format and Label Disks.
2.2. Overview of the Installation Procedure

The object of this exercise is to load the Release 1.4 binaries from the magnetic tape onto your local or network disk subsystem.

The basic steps in installation are:

1. Load the tape.
2. Load the 1.4 installation utility.
3. If you are installing a server, halt any diskless clients.
4. Run the installation utility.
5. Optionally, use the tar(1) command to extract the manual pages and/or color demos.
6. Reconfigure your system kernel.

If you have performed an installation of a Sun system before, you will recall that we use several conventions in the procedures and examples to try and clarify things:

- What the system types at you is printed in **typewriter font like this**.
- What you type at the system is shown in **boldface typewriter font like this**. Everything shown in boldface should be typed exactly as it appears.
- Where parts of a command are shown in *italic text like this*, they refer to a variable which you have to substitute from a selection; it is up to you to make the proper substitution.

One very important variable in the examples is *tape*, which should be replaced with the UNIX device name for your tape:

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Tape Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>ar</td>
<td>Archive quarter-inch tape</td>
</tr>
<tr>
<td>mt</td>
<td>Nine-track magnetic tape</td>
</tr>
<tr>
<td>st</td>
<td>SCSI tape controller</td>
</tr>
</tbody>
</table>
Installing the 1.4 Release

For example, a common configuration would load from a quarter-inch magnetic tape cartridge via a SCSI tape controller. In this case, tape would be replaced by st (SCSI Tape) everywhere. Now, you are ready to begin the actual installation.

2.3. Load the Tape

PLEASE NOTE: If you are installing 1.4 on a network disk server, you must have a tape drive on the server machine.

PLEASE NOTE: We do not guarantee full compatibility between the 4-track 20 MByte and the 9-track 45 MByte drives.

Your chances of successfully reading tapes produced by a different type of drive are increased if you follow the manufacturer's instructions for drive maintenance: clean the drive heads after every use of a new tape and after every eight hours of use.

This release contains a new mt with retensioning capability which also helps solve compatibility problems. Once you are running the release, please use the retensioning capability before writing a tape. See the mt(1) page included in Chapter 8 of this manual for instructions.

In the case of the nine-track tape, load the tape onto the tape drive and put the drive on-line.

For a quarter-inch cartridge, hold the cartridge so that the little window with the word "SAFE" in it is at the top left-hand corner. To insert the tape, firmly push the cartridge into the slot in the front of the tape drive. The tape will suddenly go all the way in with a definite "click". The tape is now ready for use.

2.4. Extract 1.4 Installation Utility from Tape

When you have loaded the tape, become super-user and use the tar(1) command to extract the 1.4 installation utility.

- If you are using a local tape drive, do the following. Remember to replace tape with the appropriate device abbreviation for your tape (ar for the Archive drive, st for a SCSI tape drive, or mt for the nine-track tape):

```
  gaia% su
  Password: Type the root password here
  gaia# cd /usr/etc
  gaia# mt -f /dev/nxtape0 rew
  gaia# mt -f /dev/nxtape0 ftf 3
  gaia# tar xvpf /dev/nxtape0
  x 1.4_upgrade, 5075 bytes, 10 tape blocks
  gaia#
```

- If you are using a remote tape drive, do the following. Note that, since you are performing a remote process as super-user, the hostname of the local machine (which you are typing commands on) must be in the remote machine's ~/.hosts file to avoid permission problems. In addition, each machine must have an entry for the other (name and Internet
address) in its /etc/hosts file. Remember to replace tape with the appropriate device abbreviation for the remote tape drive you are using, to replace remote_host with the hostname of the machine this tape drive is attached to, and to replace block_size with 20b for a 1/2" tape or 128b for a 1/4" tape:

    gaia$ su
    Password: Type the root password here
    gaia$ cd /usr/etc
    gaia$ rsh remote_host mt -f /dev/nxrtape0 rew
    gaia$ rsh remote_host mt -f /dev/nxrtape0 fsf 3
    gaia$ rsh remote_host dd if=/dev/nxrtape0 bs=block_size of=1.4_upgrade, 5075 bytes, 10 tape blocks
    gaia$

If you get a "Broken pipe" message when you run the last command, you can ignore it.

2.5. Run the 1.4 Installation Utility

PLEASE NOTE: If you are installing 1.4 on a disk server, you must halt all diskless clients of this server before beginning the procedures in this section.

ALSO: Do not run SunWindows during the installation procedure.

Next, you run the 1.4 installation utility to extract the 1.4 files from the release tape. This process takes about 10 minutes.

If you are doing the installation on a server, note that the installation utility takes care of 1.4 installation on your diskless clients, but that it assumes a canonical form of the nd configuration file /etc/nd.local in order to do so. In particular, if there are lines in the server's /etc/nd.local for client partitions which are commented out (lines with a leading '#'), these clients will not have 1.4 installed on them. If you wish to have the 1.4 Release installed on these clients, you must remove the comment symbol from their lines in the file before performing installation on the server. If you do not do this during the initial install, you will have to run through the entire installation procedure again (on the server and all the clients) in order to bring the commented-out client partitions up to date.

Also, if your /etc/nd.local file has user... lines and/or ether... lines that refer to non-existent clients, comment out these lines before running the installation utility. Again, the utility will not run to completion if this is not done.

To run the utility, type the following command, using only one of each set of terms in braces. The first set of terms allows you to decide whether to install or back out the 1.4 binaries; the second set specifies your tape device; and the last designates your machine as a server, standalone with tape drive, or Workstation without a tape drive using a another machine's (server_name's) tape drive (the command is printed on two lines for formatting purposes only; type it as a single line):

    gaia$ 1.4_upgrade {install|uninstall} {ar|mt|st}
      {server|tapefull|tapeless server_name}
For example, the installation command and system response for a server machine with a half-inch tape drive would look like this:

cronin# 1.4_upgrade install at server
Beginning 1.4 install.
Extracting 1.4 object files.
x ./usr/lib/vpltdmp, 18432 bytes, 36 tape blocks
x ./usr/lib/f77pass1, 200704 bytes, 392 tape blocks
x ./usr/lib/libF77.a, 60602 bytes, 119 tape blocks
x ./usr/lib/libF77_p.a, 67284 bytes, 132 tape blocks
[ and so on... extraction takes about half an hour... ]
x ./sys/sys/init_systent.c, 8530 bytes, 17 tape blocks
x ./sys/sys/tty_conf.c, 2158 bytes, 5 tape blocks
x ./sys/sys/uipc_proto.c, 810 bytes, 2 tape blocks
x ./sys/machine symbolic link to sun
x ./boot, 23552 bytes, 46 tape blocks
Installing new bootable code on server.
Beginning 1.4 install on diskless clients.
Beginning 1.4 install on client client_1.
Completed 1.4 install on client client_1.
Beginning 1.4 install on client client_2.
Completed 1.4 install on client client_2.
Beginning 1.4 install on client client_3.
Completed 1.4 install on client client_3.
Beginning 1.4 install on client client_4.
Completed 1.4 install on client client_4.
Completed 1.4 install on diskless clients.
Installing new sky microcode.
Running ranlib on new libraries.
Applying patches to object code.
1.4 install completed.
You should now reconfigure and rebuild your kernel.

Note: if you are using a SCSI tape for the install, you may get a message like "/dev/nrst0 rewind 1 failed: I/O error" at the end of the install script. You can ignore it.

To install the release on my machine ('gaia'), which is a tapeless Workstation using the 1/4" SCSI tape drive on a machine named 'hal', the command line would be:

gaia# 1.4_upgrade install at tapeless hal

2.6. Loading Optional Software from the Release Tape

The seventh file on the upgrade tape contains new/revised manual pages and several new color demos (source and executables); you may load the pages and/or demos if you desire.

Manual pages take approximately 97 KBytes of storage. Pages are:

Also included in 1.2 and 1.3:

as(1), cc(1), cpp(1), screendump(1), screenload(1), malloc(3), zo(4S), acct(5), utmp(5), skyversion(8), and dkinfo(8).
New in 1.4:
  `clear(1)`, `diag(8S)`, `mt(1)`, `tip(1C)`, `bessel(3F)`, `bit(3F)`, `time(3F)`, `rand(3F)`, `etime(3F)`,
  `fdate(3F)`, `stat(3F)`, `long(3F)`, `cgtwo(4S)`, `ic(4S)`, `mem(4S)`, and `ftpusers(5)`.

Demos take approximately 4.1 MBytes of space. They are:

In `demo/src`:
  `draw.c`, `shaded.c`, `showmap.c`, `stringart.c`, `suncube.c`, `shademo.c`, `rotcube.c`, `molecule.c`,
  `cframedemo.c`, `demolib.h`, `Makefile`, `show.c`.

In `demo`:
  `MAPS/`, `DATA/`, `globeframes/`, `READ_ME`, `cframedemo`, `shaded`, `draw`, `molecule`, `shademo`,
  `showmap`, `stringart`, `suncube`, `COLORPIX`, `show`.

Use `tar(1)` to extract the pages and/or demos as follows. If you want to load the pages or the
demos (not both), use the appropriate command line argument to `tar`; if you omit
the argument(s), `tar` loads both the pages and demos. The complete load takes about 5 minutes.

For a machine with a local tape drive:

```
  gaias# cd /usr
  gaias# mt -f /dev/nrtape0 rew
  gaias# mt -f /dev/nrtape0 fsf 6
  gaias# tar xvpf /dev/nrtape0 { man | demo }
  gaias# mt -f /dev/nrtape0 rew
```

For a machine using a remote tape drive, type the following. Remember to replace `tape` with the
appropriate device abbreviation for the remote tape drive you are using, to replace `remote_host`
with the hostname of the machine this tape drive is attached to, and to replace `block_size` with
`20b` for a 1/2" tape or `128b` for a 1/4" tape:

```
  gaias# cd /usr
  gaias# rsh remote_host mt -f /dev/nrtape0 rew
  gaias# rsh remote_host mt -f /dev/nrtape0 fsf 6
  gaias# rsh remote_host dd if=/dev/nrtape0 bs=block_size | tar xvpBf - { man | demo }
  gaias# rsh remote_host mt -f /dev/nrtape0 rew
```

The `dd` command above will print the number of records read in and written out, for example:

```
  6 + 0 records in
  6 + 0 records out
```

This message may be interspersed with the `tar` output and, in any case, may be ignored. Also,
you can ignore the "Broken pipe" message delivered by the `tar`.

If you load the pages, and normally use `catman(8)` to create pre-formatted copies of your online
manual pages, then don't forget to re-issue the `catman` command for the new 1.4 pages. If you
are installing a server, you must do this before re-booting your clients.

### 2.7. Reconfigure your UNIX System Kernel

Finally, to complete the 1.4 Release installation, you must reconfigure your system kernel.
PLEASE NOTE: Substantive changes have been made to the device description lines in the
kernel configuration file to allow for Sun-2/50 and Sun-2/180 support; therefore all systems
must begin by editing the new /sys/conf/GENERIC file. Do not use your old system
configuration file.

If you are doing kernel configuration for the first time, you can use the procedures in the 1.1 Sys-
tem Manager's Manual to help you along. See the Chapter Installing UNIX for the First Time,
the sections beginning with Kernel Configuration.

If you have previously configured a kernel, you can use the following sections to guide you
through reconfiguration. The first subsection is an annotated copy of the new GENERIC file;
please read it carefully to make sure you are including the correct device description lines for
your system. The second subsection gives reconfiguration procedures for standalone machines,
and the third subsection addresses servers.

2.7.1. Kernel Reconfiguration — an Annotated Copy of GENERIC

The following pages provide an annotated copy of the GENERIC file shipped with this distribu-
tion. You can use the explanations of each line in the file to determine which lines should be
included in your own system configuration file.

#
#
# GENERIC SUN
#

machine  sun
    [ mandatory. ]

cpu      "SUN2"
    [ mandatory. ]

ident     GENERIC
    [ mandatory. If you use "GENERIC" as your system identifier, you may use the "swap generic" clause in the
"config" line below. If you customize the identifier to SYS_NAME, you must either include an "options GENERIC"
line, or specify at least the device where your root file system lives in place of "swap generic". For example, the
"config" line for a standard Sun-2 might read: "config vmunix root on xy". See General and Specific System
Description Lines, above, for information.Finally, if SYS_NAME contains both alpha and numeric characters (as in,
for example, SDST120), you must enclose the name in double quotes ("SDST120") or you will get a syntax error
when you run /etc/config. ]

timezone  8 dst
    [ mandatory. Specifies your timezone. Adjust value accordingly. ]

maxusers  4
    [ mandatory. Number may vary. For most systems, "2" is the proper value for maxusers. See the section
General System Description Lines, above, for information. ]

options  INET
    [ mandatory. Controls inclusion of Internet code — see inet(4). You must also include the "pseudo-device inet"
and "pseudo-device loop" lines below. ]

options  SYSACCT
    [ Controls inclusion of code to do process accounting — see acct(2) and acct(5). If you include this line, you must
also include the "pseudo-device sysacct" line below. ]

config  vmunix swap generic
    [ mandatory. Specify kernelname and configuration clauses. Please see Specific System Description Lines,]
above, for information.

pseudo-device  pty
[Pseudo-tty's. Needed for network or window system.]
pseudo-device  bk
[Berknet line discipline for high speed tty input — see bk(4).]
pseudo-device  sysacct
[See “options SYSACCT” line above.]
pseudo-device  inet
[mandatory. See “options INET” line above.]
pseudo-device  ether
[ARP code. Must include if using Ethernet — see arp(4).]
pseudo-device  loop
[mandatory. Software loop back network device driver — see lo(4). Must include with ‘options INET’.]
pseudo-device  nd
[Network disk. Necessary for servers and diskless clients, and for machines serving as remote hosts for remote installation — see nd(4).]
pseudo-device  win128
[Window system. Number indicates maximum windows. If you include this line, you must also include the “pseudo-device dtop”, “ms”, and “kb” lines just below.]
pseudo-device  dtop
[Maximum number of screens (‘desktops’). Required for window system.]
pseudo-device  ms
[Maximum number of mice. Required for window system — see ms(4).]
pseudo-device  kb3
[Maximum number of Sun keyboards. Required if using any Sun keyboard, and for the window system.]
pseudo-device  ings
[Sun MicroINGRES lock device.]
controller   mb0 at nexus ?
[mandatory. Main bus code.]
controller   ipco at mb0 csr all virt 0x0eb0040 priority 2
[1st Interphase SMD disk controller — see ip(4).]
controller   ipcl at mb0 csr all virt 0x0eb044 priority 2
[2nd Interphase controller.]
disk         ipo at ipco drive 0
[1st disk on 1st Interphase controller.]
disk         ip1 at ipco drive 1
[2nd disk on 1st Interphase controller.]
disk         ip2 at ipcl drive 0
[1st disk on 2nd Interphase controller.]
disk         ip3 at ipcl drive 1
[2nd disk on 2nd Interphase controller.]
controller   xyc0 at mb0 csr all virt 0x0eb040 priority 2 vector xynintr 72
[1st Xylogics SMD disk controller — see xy(4).]
controller   xycl at mb0 csr all virt 0x0eb048 priority 2 vector xynintr 73
[2nd Xylogics controller.]
disk         xyo at xyc0 drive 0
[1st disk on 1st Xylogics controller.]

Release 1.4 of February 1985
Installing the 1.4 Release

disk
  xy1 at xyc0 drive 1
  [2nd disk on 1st Xylogics controller.]
disk
  xy2 at xycl drive 0
  [1st disk on 2nd Xylogics controller.]
disk
  xy3 at xycl drive 1
  [2nd disk on 2nd Xylogics controller.]
controller
  sc0 at mb0 csr Ox80000 priority 2
  [1st SCSI controller on a Sun-2/120 or Sun-2/170.]
controller
  sc0 at mb0 csr vme busmem Ox200000 priority 2 vector scintr 64
  [1st SCSI controller on a Sun-2/160.]
disk
  sd0 at sc0 drive 0 flags 0
  [1st disk on 1st SCSI controller.]
disk
  sd1 at sc0 drive 1 flags 0
  [2nd disk on 1st SCSI controller.]
tape
  st0 at sc0 drive 32 flags 1
  [SCSI tape.]
controller
  scl at mb0 csr Ox84000 priority 2
  [2nd SCSI controller.]
disk
  sd2 at scl drive 0 flags 0
  [1st disk on 2nd SCSI controller.]
disk
  sd3 at scl drive 1 flags 0
  [2nd disk on 2nd SCSI controller.]
tape
  st1 at scl drive 32 flags 1
  [SCSI tape.]
device
  ropc0 at mb0 csr Oxeee000
  [mandatory. Raster Op chip — see ropc(4).]
device
  sky0 at mb0 csr Ox2000 priority 2
  [Sky Floating Point board in any Sun-1, Sun-2/120, or Sun-2/170.]
device
  sky0 at mb0 csr vme busio Ox8000 priority 2 vector skyintr 176
  [Sky Floating Point board in a Sun-2/50 or Sun-2/160.]
device
  zs0 at mb0 csr all virt Oxexec800 flags 3 priority 3
  [CPU serial I/O ports — see zs(4).]
device
  zs1 at mb0 csr all virt Oxexec000 flags Ox103 priority 3
  [Sun-2 Video Board ports. Required for Sun-2 keyboard and mouse.]
device
  zs2 at mb0 csr Ox80800 flags 3 priority 3
  [1st two serial I/O ports on 1st SCSI Board.]  
device
  zs3 at mb0 csr Ox81000 flags 3 priority 3
  [2nd two serial I/O ports on 1st SCSI Board.]  
device
  zs4 at mb0 csr Ox84800 flags 3 priority 3
  [1st two serial I/O ports on 2nd SCSI Board.]  
device
  zs5 at mb0 csr Ox85000 flags 3 priority 3
  [2nd two serial I/O ports on 2nd SCSI Board.]  
device
  mt10 at mb0 csr all virt Oxeeb0620 flags Oxffff priority 4 vector mtlintr 136
  [Systech terminal MUX — see mt(4).]
device
  ie0 at mb0 csr Ox88000 priority 3
  [1st Sun-2 Ethernet Controller on a Sun-2/120 or Sun-2/170.]
device
  ie0 at mb0 csr vme virt Ox0ee3000 priority 3

Release 1.4 of February 1985
2.7.2. Kernel Reconfiguration for Standalone Systems

For standalone machines, proceed as follows.

1. Change directory to /sys/conf, and copy the GENERIC file to produce your own system configuration file. We'll call this latter file SYS_NAME:
 gaia# cd /sys/conf
 gaia# cp GENERIC SYS_NAME
 gaia# chmod +w SYS_NAME

2. Create the ./SYS_NAME directory (if you haven’t already) to contain the kernel image. Remember: since the system build utility /etc/config places its output files here, the directory must have the same name as your system configuration file:

 gaia# mkdir ./SYS_NAME

3. Edit /sys/conf/SYS_NAME to reflect your system configuration.

As you look at the file, you’ll notice several new lines and various changes from the 1.1 version of GENERIC. Use the annotated copy of GENERIC provided in the previous section for an explanation of these changes. Make sure you are including the proper device description lines for your system.

4. Still in the /sys/conf directory, run /etc/config. Then change directory to the new configuration directory, and make the new system (remember to substitute your actual system image name for SYS_NAME):

 gaia# /etc/config SYS_NAME
 gaia# cd ../SYS_NAME
 gaia# make depend
 [ lots of output ]
 gaia# make
 [ lots of output ]

5. Now you can save your old kernel, install your new one, and try everything out:

 gaia# mv /vmunix /vmunix.old
 gaia# cp vmunix /vmunix
 gaia# /etc/halt
    The system goes through the halt sequence, then
    the monitor displays its prompt, at which point you
    can boot the system:
    > b vmunix
    The system boots up multi-user, and then
    you can try things out.
 gaia#

6. If the system appears to work, this completes installation. If the new kernel doesn’t seem to be functioning properly, boot /vmunix.old, copy it back to /vmunix, and go about fixing your new kernel:

 gaia# /etc/halt
 > b vmunix.old -s
 gaia# mv /vmunix /vmunix.oops
 gaia# cp /vmunix.old /vmunix
 gaia# "D [ Brings the system up multi-user ]
 gaia#
2.7.3. Kernel Reconfiguration for Servers

For server machines, proceed as follows.

1. Change directory to /sys/conf, and copy the GENERIC file to produce your own system
   configuration file. We'll call this latter file SYS_NAME:

   gaia# cd /sys/conf
   gaia# cp GENERIC SYS_NAME
   gaia# chmod +w SYS_NAME

2. Create the ..SYS_NAME directory (if you haven't already) to contain the kernel image.
   Remember: since the system build utility /etc/config places its output files here, the directory
   must have the same name as your system configuration file:

   gaia# mkdir ..SYS_NAME

3. Edit /sys/conf/SYS_NAME to reflect your system configuration.

   As you look at the file, you'll notice several new lines and various changes from the 1.1 ver-
   sion of GENERIC. Use the annotated copy of GENERIC provided above for an explanation
   of these changes. Make sure you are including the proper device description lines for your
   system.

4. Still in the /sys/conf directory, run /etc/config. Then change directory to the new
   configuration directory, and make the new system (remember to substitute your actual sys-
   tem image name for SYS_NAME):

   gaia# /etc/config SYS_NAME
   gaia# cd ..SYS_NAME
   gaia# make depend
   [ lots of output ]
   gaia# make
   [ lots of output ]

5. If you have a specially configured client kernel, it can be reconfigured now as well:

   gaia# cd /sys/conf
   gaia# cp GENERIC CLIENT_KERNEL_NAME
   gaia# chmod +w CLIENT_KERNEL_NAME
   [ Edit CLIENT_KERNEL_NAME to reflect all clients' systems.
     Be especially careful with the device description lines, given above. ]
   gaia# mkdir ..CLIENT_KERNEL_NAME
   gaia# /etc/config CLIENT_KERNEL_NAME
   gaia# cd ..CLIENT_KERNEL_NAME
   gaia# make depend
   [ lots of output ]
   gaia# make
   [ lots of output ]

6. Now you can position yourself in the directory which has the server's kernel in it, save your
   server's old kernel, install your new one, and try everything out:
Installing the 1.4 Release

7. Next, install the appropriate client kernel in `/pub`.
   - If you reconfigured a special client kernel (in Step 5 above), copy it into `/pub`:
     
     ```
     gaia# cd /sys/CLIENT KERNEL NAME  [or wherever your client kernel is]
     gaia# cp vmunix /pub/vmunix
     ```
   - Otherwise place a copy of your server's kernel (if appropriate) in `/pub`:
     
     ```
     gaia# cp /vmunix /pub/vmunix
     ```

8. If everything appears to work, you can finish by rebooting each of your clients. See the final step in the standalone instructions above if you have problems with your kernel.

2.8. "Uninstalling" the 1.4 Release

If you run into problems while running 1.4, you can back out the changes\(^1\) by using the "uninstall" option of the installation utility. You then reconfigure your kernel, and can run as a 1.1 Sun system again. Proceed as follows.

**PLEASE NOTE:** If you are 'uninstalling' a network disk server, you **must** halt all diskless clients before proceeding.

1. Load the release tape, as described in the normal installation procedure above.
2. To run the 'uninstall' option of the utility, type the following command, using only one of each set of terms in braces. The first set of terms specifies your tape device; and the last designates your machine as a server, standalone with tape drive, or Workstation on a network using a another machine's (server_name's) tape drive:

   ```
   gaia% su
   Password: Type the root password here
   gaia# cd /usr/etc
   gaia# 1.4_upgrade uninstall {ar|mt|st}
   {server|tapefull|tapeless server_name}
   ```

\(^1\) **NOTE:** Optional software is not backed out during the uninstall.
This takes about 25 minutes.

3. Reconfigure your kernel as described in the section just above.
Chapter 3

Using the New diag to Format and Label Disks

The second file on the 1.4 Release tape contains a new version of diag, the disk formatting/labelling program. This version of diag supports several new SCSI 85 MByte drives (the Micropolis 1325, the Fujitsu M2243AS, and the Vertex V185), provides improved disk diagnostics and bad block handling, and supports Rev. C of the Xylogics 450 Disk Controller PROM's.

If you have a new disk and you are installing a complete release, you must use this version of diag during the standard UNIX installation procedure for disk initialization (the phase titled Using the Diag Utility in the System Manager's Manual). This means that you will be using your 1.1 Release tape and 1.1 System Manager's Manual to perform the first part of UNIX installation, then switch to the 1.4 Release tape and this chapter for disk formatting and labelling, and then return to the 1.1 tape and documentation to complete UNIX installation. Finally, when you have installed 1.1, you will switch back to the 1.4 tape to perform 1.4 installation, following the instructions in the earlier chapters of this document. Instructions follow.

1. Begin UNIX installation with your 1.1 Release tape, following the procedures in the System Manager's Manual chapter Installing UNIX for the First Time. Load the tape and the bootstrap program. If you are using a 100U/150U Manual for Release 1.1, this means you should complete installation through section 4.3.3 (Loading the Bootstrap Program); if you are using a 120/170 Manual for Release 1.1, go through section 3.3.3.

2. When you have completed this section, you should see the bootstrap program's prompt:

   Boot:

   Now switch to your 1.4 Release tape (first cartridge for 1/4" distributions).

3. Boot diag from your 1.4 tape by typing the following command (replacing tape with the appropriate device abbreviation for your controller):

   Boot: tape(0,0,1)
   Size: some_number+some_number+some_number bytes [ varies with release ]

3.1. Phase One: Specifying Hardware Configuration

When diag starts up, it displays a sign on message, and begins its first series of queries about your hardware configuration. Specifically, diag asks what sort of disk controller(s) and disk drive(s) you have. The answers to these questions 'configure' diag to work with this specific hardware.

Release 1.4 of February 1985
1. The first thing `diag` wants to know is the type of disk controller to use:

   Version `sees version_number and date`
   Disk Initialization and Diagnosis
   When asked if you are sure, respond with 'y' or 'Y'

   specify controller:
   0 - Interphase SMD-2180
   1 - Xylogics 440 (prom set 926)
   2 - Xylogics 450
   3 - Adaptec ACB 4000 - SCSI
   which one? `type the number for your controller type`

2. `diag`'s next question is what address this controller occupies on the main system bus. Unless you have an unusual controller configuration\(^2\), you can take the address from this table:

   Table 3-1: Default Addresses for Disk Controllers

<table>
<thead>
<tr>
<th>Controller Type</th>
<th>Address (hex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interphase SMD-2180</td>
<td>40  48</td>
</tr>
<tr>
<td>Xylogics 450/440</td>
<td>ee40 ee48</td>
</tr>
<tr>
<td>Adaptec ACB 4000-SCSI for Sun-2/120 or 170</td>
<td>80000 84000</td>
</tr>
<tr>
<td>Adaptec ACB 4000-SCSI for Sun-2/160</td>
<td>ee2800</td>
</tr>
</tbody>
</table>

   When you give `diag` your controller's address, it echoes the address back to you:

   Specify controller address on the mainbus (in hex): `address`
   Device address: `address you entered`

3. If your controller interfaces to the SCSI bus (Adaptec, for example), `diag` asks for the address of the controller on the SCSI bus as well. The correct address is 0 for the first (or only) SCSI disk controller; 1 is the target number for the second:

   Which target? `SCSI bus address`

4. Next, `diag` wants to know the unit number of the disk on this controller which you want to label and format. The correct unit number is 0 for the first (or only) drive on this controller; 1 is the unit number for the second:

\(^2\) If you have one Xylogics SMD Controller and one Interphase SMD Controller in your system, you cannot use the default addresses (the Interphase Controller sees only four bytes). The Interphase must be configured to be at address 48, and the Xylogics at ee40. The Xylogics does not need to be the first controller specified. If both controllers are the same type, use the defaults. See the Hardware Configuration and Expansion chapter in the System Manager's Manual for the 100U/150U for Interphase Board configuration; board configuration procedures for other controllers are in the Hardware Installation Manual for the appropriate machine model.
5. Now diag wants to know the type of disk drive you are working with. Diag displays a menu of the different disks it knows about, and asks you to specify your disk type. There are two menus, one for SMD disk drives, such as the Fujitsu Eagle, and one for SCSI drives like the Micropolis; the menu corresponding to your controller-type comes up. Let's assume you're formatting a SCSI disk:

Specify drive:
0 - Micropolis 1304
1 - Micropolis 1325
2 - Maxtor XT-1050
3 - Fujitsu-M2243AS
4 - Vertex V185
5 - Other

which one? type the number for your drive type

ncyl number acyl number nhead number nssect number interleave number status:
[ ...status information... ]

When you select the disk drive you wish to operate on from the menu, diag displays a table of physical data about that disk, which includes the number of cylinders, number of alternate cylinders, number of heads, and number of sectors per track, and then displays drive-specific status information.

At this point, diag knows all it needs to know about the controller and the disk you are using, and displays its prompt:

diag>

You can now begin the second phase of the diag procedures, formatting and labelling the disk.

3.2. Phase Two: Formatting and Labelling the Disk

Next, you use diag's disk formatting function to prepare and format new disk(s). After formatting, the disk must be labelled. The label subcommand is used to write a default partition map label to the disk.

There are two distinct disk formatting and labelling paths: one for disks controlled by SCSI disk controllers (Adaptec, for example), and one for disks controlled by SMD controllers (Xylogics, for example). Follow the procedures which are appropriate for your system configuration.

3.2.1. Formatting SCSI Disks

1. Begin by typing the format command. This calls up the SCSI formatting subprogram to diag, and you'll see its prompt:

diag> format

SCSI format.

format>
The SCSI format subprogram has various utilities which allow you to prepare and format a new disk. If you'd like to see a list of its capabilities, ask for help by typing ?:

```
format> ?

SCSI Format Subcommands:
f: format disk
r: read defect list from disk
p: print defect list from disk
a: add defect to list
d: delete defect from list
s: surface analysis
t: translate block no. to cyl/hd/bfl
q: quit format
```

2. Your first task is to type in the disk defect list supplied with your disk drive.

The location of this list depends on the type of workstation you have. Usually, the list is taped to the front or top of your pedestal. If you don't see it, or have misplaced it during unpacking, you should be able to find a second copy of the list taped to the disk drive housing inside the pedestal (remove the two Phillips screws from the top rear of the enclosure, slide the top of the beige metal housing off, and check).

To enter the defect list, type `a` (for 'add defect to list') to the SCSI format prompt, and then type the appropriate data for each subsequent prompt, taking this data from your defect listing:

```
format> a
cylinder? number
head? number
bytes from index? number
```

If you are dealing with a Micropolis drive, the defect map that comes with the disk groups defects by head; cylinder and bytes from index numbers appear in the CYL and BI columns. If you are formatting a Maxtor drive, the numbers appear in the CYL, H, and BYTE columns respectively.

3. Add an entry for each defect listed. When you're through, respond `p` (for 'print defect list') to the format prompt. This displays the information you have entered. Make sure this listing matches the defect listing which came with your drive.

4. If you need to make corrections to the data you have entered, use the 'add defect' (type `a`) and 'delete defect' (type `d`) sub-commands.

5. When you have verified the defect listing, go on to format the disk with the format (f) sub-command. After you type the sub-command, the system warns you that formatting a disk destroys all information which might already be stored on that disk, and asks for confirmation before going ahead and doing the job:

```
format> format
format/verify, DESTROYS ALL DISK DATA!
are you sure? y
```

The formatting process takes about three minutes. At the end of this process, you should get a "Defect list written on disk" message. If you get any other message ("SCSI reset", for example), the formatting process did not succeed, and the defect list has not been recorded on the disk. You **must** format the disk again.
6. When you have successfully formatted the disk, do a surface analysis of it using the surface analysis (s) sub-command. This analyzes the entire disk and adds any defects found to the disk's defect list. For a new disk, three surface analysis passes should be done:

    format> s
    Number of surface analysis passes (3 is usual)? 3

Three passes take between a half-hour and an hour to complete. When everything's done, you'll get a message to that effect.

7. If the surface analysis turns up bad sectors, the system will tell you how many bad sectors were found, and tell you to re-format the disk:

    Surface analysis complete
    some_number bad sectors found
    Use the 'f' command to format the disk.
    format>

Before continuing, you must reformat a disk with bad sectors.

8. When no bad sectors have been found by the surface analysis, you have successfully formatted the disk. Now you may go on to label the disk. This process is described below.

3.2.2. Labelling SCSI Disks

A disk must be labelled after it has been formatted. The disk label records information about how the disk is divided into partitions for such things as paging space and file systems.

1. To label your disk, type the label command in response to diag's prompt. When you give the command to diag, it asks if you want to use the logical partition map that is 'built in' to the program, and then asks for confirmation before proceeding. Default partitioning for Sun-supplied SCSI disks is outlined in the following table.

<table>
<thead>
<tr>
<th>SCSI Disk</th>
<th>Raw</th>
<th>&quot;a&quot;</th>
<th>&quot;b&quot;</th>
<th>&quot;c&quot;</th>
<th>&quot;d&quot;</th>
<th>&quot;e&quot;</th>
<th>&quot;f&quot;</th>
<th>&quot;g&quot;</th>
<th>&quot;h&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Micropolis 1304</td>
<td>50</td>
<td>8.1</td>
<td>8.4</td>
<td>43.1</td>
<td>unused</td>
<td>unused</td>
<td>unused</td>
<td>26.5</td>
<td>unused</td>
</tr>
<tr>
<td>Micropolis 1325</td>
<td>85</td>
<td>8.1</td>
<td>17.1</td>
<td>71.2</td>
<td>unused</td>
<td>unused</td>
<td>unused</td>
<td>45.9</td>
<td>unused</td>
</tr>
<tr>
<td>Maxtor XT-1050</td>
<td>50</td>
<td>8.1</td>
<td>8.4</td>
<td>44.4</td>
<td>unused</td>
<td>unused</td>
<td>unused</td>
<td>27.9</td>
<td>unused</td>
</tr>
<tr>
<td>Fujitsu M2243AS</td>
<td>86</td>
<td>8.1</td>
<td>17.1</td>
<td>72.0</td>
<td>unused</td>
<td>unused</td>
<td>unused</td>
<td>46.7</td>
<td>unused</td>
</tr>
<tr>
<td>Vertex V185</td>
<td>85</td>
<td>8.1</td>
<td>17.1</td>
<td>70.9</td>
<td>unused</td>
<td>unused</td>
<td>unused</td>
<td>45.5</td>
<td>unused</td>
</tr>
</tbody>
</table>

If you confirm, diag will partition your disk according to these default maps.

---

Note that in all discussions of disk partitioning, numbers are approximate, since formatted capacity depends on the type of controller being used with the drive. Also, note that a 'Megabyte', as far as numbers given in discussions of disk capacity, is defined as one million bytes.

If these default maps are inadequate for your system, you can use diag's partition command at this point to design your own labels. See diag(8S) for information.
Using the New diag to Format and Label Disks

diag> label
label this disk...
OK to use logical partition map 'your disk type'? y
Are you sure you want to write? y

2. After labelling the disk, diag automatically verifies the label it has just written. As an example, the verify for a Fujitsu M2322 might look like this:

[ This is an example only; do not enter this information. ]

verify label
id: <Fujitsu-M2322 cyl 821 ait 2 hd 10 sec 32 interleave 1>
Partition a: starting cyl=0, # blocks=15884 [ #'s vary with disk ]
Partition b: starting cyl=50, # blocks=33440
Partition c: starting cyl=0, # blocks=262720
Partition g: starting cyl=155, # blocks=213120

diag>

3. This completes the formatting and labelling process for a single SCSI disk. If you have a second disk drive on your controller, or two controllers and two or more drives, you may now get back to the beginning of diag's first phase by responding to the diag prompt with the command diag:

diag> diag

Note that you must complete both phases of diag for each of your disks. Be careful each time in the first phase of diag to respond with correct values when you are prompted for: controller mainbus address, controller type, disk type, and disk unit number.

4. If you have only one disk, or if you are done formatting and labelling all of your disks, you are ready to continue with the next phase of installation. Get back to the bootstrap program by typing the q (quit) command to diag, and you'll see the boot program's prompt again:

diag> q

Boot:

5. Now switch back to your 1.1 tape, and continue with installation using the procedures in your 1.1 manual. If you are using a 100U/150U Manual for Release 1.1, you are ready to start section 4.3.5, Loading the Mini UNIX System; if you are using a 120/170 Manual for Release 1.1, begin with section 3.3.5.

When you have completed installation for the 1.1 Release, follow the procedures in Chapters 1 and 2 of this document for installing the incremental 1.4 Release.

3.2.3. Formatting SMD Disks

1. Begin by typing the format command. Diag warns you that formatting a disk destroys all information which might already be stored on that disk, and asks for confirmation before going ahead and doing the job:
Using the New `diag` to Format and Label Disks

```
    diag> format
    format/verify, DESTROYS ALL DISK DATA!
    are you sure? y
```

2. Now, `diag` asks for the number of surface analysis passes to do on the disk. The surface analysis phase of the formatting is for detecting bad spots on the disk and mapping them to alternate places. When formatting a disk for the first time, five surface analysis passes is a reasonable number. You can use just one surface analysis pass if you are reformatting a disk that has been formatted before and is known not to have any bad spots:

```
    # of surface analysis passes (5 recommended)? 5
```

The complete format phase (5 passes) takes about two hours for a Fujitsu M2322, and about five hours for an Eagle. It is well worth taking the time to do this when formatting a new disk.

After surface analysis, `diag` formats the disk, and displays the current cylinder number as it formats each cylinder.

At the end of the format process, `diag` displays a message telling you that it has finished; then it tells you to use the label command to label the disk:

```
    cyl 588 format complete - 0 bad sector(s)
    Use the label command to label the disk
    diag>
```

Labelling the disk is the next step.

### 3.2.4. Labelling SMD Disks

A disk must be labelled after it has been formatted. The disk label records information about how the disk is divided into partitions for such things as paging space and file systems.

To label your disk, you use the `label` subcommand. When you give the command to `diag`, it asks if you want to use the logical partition map that is ‘built in’ to the program, and then asks for confirmation before proceeding\(^6\). Default partitioning for Sun-supplied SMD disks is outlined in the following table\(^6\):

---

\(^6\) If these default maps are inadequate for your system, you can use `diag`'s partition command at this point to design your own labels. See `diag(8S)` for information.

\(^6\) Note that in all discussions of disk partitioning, numbers are approximate, since formatted capacity depends on the type of controller being used with the drive. Also, note that a 'Megabyte', as far as numbers given in discussions of disk capacity, is defined as one million bytes.
Table 3-3: Default Partition Sizes for SMD Disk Subsystems

<table>
<thead>
<tr>
<th>SMD Disk</th>
<th>Raw</th>
<th>“a”</th>
<th>“b”</th>
<th>“c”</th>
<th>“d”</th>
<th>“e”</th>
<th>“f”</th>
<th>“g”</th>
<th>“h”</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fujitsu 2312K (8”)</td>
<td>84</td>
<td>8.1</td>
<td>17.1</td>
<td>67.3</td>
<td>unused</td>
<td>unused</td>
<td>unused</td>
<td>42.0</td>
<td>unused</td>
</tr>
<tr>
<td>Fujitsu 2284 (14”)</td>
<td>169</td>
<td>8.1</td>
<td>17.1</td>
<td>134.5</td>
<td>unused</td>
<td>unused</td>
<td>unused</td>
<td>109.1</td>
<td>unused</td>
</tr>
<tr>
<td>Fujitsu 2322 (8”)</td>
<td>168</td>
<td>8.1</td>
<td>17.1</td>
<td>134.5</td>
<td>unused</td>
<td>unused</td>
<td>unused</td>
<td>109.1</td>
<td>unused</td>
</tr>
<tr>
<td>Fujitsu 2351 Eagle</td>
<td>474</td>
<td>8.1</td>
<td>17.1</td>
<td>395.7</td>
<td>unused</td>
<td>unused</td>
<td>unused</td>
<td>369.8</td>
<td>unused</td>
</tr>
</tbody>
</table>

1. Type the label subcommand in response to diag’s prompt:
   ```
diag> label
label this disk...
OK to use logical partition map 'your disk type'? y
Are you sure you want to write? y
```

2. After labelling the disk, diag automatically verifies the label it has just written. As an example, the verify for a Fujitsu M2322 might look like this:
   ```
   [ This is an example only; do not enter this information. ]
   verify label
   id: <Fujitsu-M2322 cyl 821 alt 2 hd 10 sec 32 interleave 1>
   Partition a: starting cyl=0, # blocks=15884 [ #’s vary with disk ]
   Partition b: starting cyl=50, # blocks=33440
   Partition c: starting cyl=0, # blocks=262720
   Partition g: starting cyl=155, # blocks=213120
   diag>
   ```

3. This completes the formatting and labelling process for a single disk. If you have a second disk drive on your controller, or two controllers and two or more drives, you may now get back to the beginning of diag’s first phase by responding to the diag prompt with the command diag:
   ```
diag> diag
   ```

   Note that you must complete both phases of diag for each of your disks. Be careful each time in the first phase of diag to respond with correct values when you are prompted for: controller mainbus address, controller type, disk type, and disk unit number.

4. If you have only one disk, or if you are done formatting and labelling all of your disks, you are ready to continue with the next phase of installation. Get back to the bootstrap program by typing the q (quit) command to diag, and the bootstrap displays its prompt sign again:
   ```
diag> q
Boot:
```

5. Now switch back to your 1.1 tape and continue installation from your 1.1 manual. If you are using a 100U/150U Manual for Release 1.1, you are ready to start section 4.3.5, Loading the Mini UNIX System; if you are using a 120/170 Manual for Release 1.1, begin with
section 3.3.5.

When you have completed installation for the 1.1 Release, follow the procedures in Chapters 1 and 2 of this document for installing the incremental 1.4 Release.
Chapter 4

Changes from the 1.1 Release Software

This chapter briefly describes the contents of the 1.2 Release Software, and details changes that have been made since the 1.1 Release. User programs may need to be re-linked or re-compiled to take advantage of these changes.

4.1. Software Contents and Details of Changes

4.1.1. Window System

- Pw_region has been rewritten to handle retained pixwins. Also, a problem with locking the wrong section of the screen during pixwin operations using a region has been corrected.
- Pw_stencil has been corrected to reset some global state that previously left a time bomb in the code that could be tripped over later by a pw_rop or pw_replrop call.
- Pw_get has been corrected to return the proper value when called using a pixwin that is not retained.
- The optsw_setvalue call for enumerated and boolean items (optsw_enum.c and optsw_bool.c) now correctly maintains the correspondence between internal and display data.
- /etc/utmp is now reset appropriately when Exit is invoked in the Root Manager menu.
- lockscreen can now finish the Show Desktop command when lockscreen had been invoked by the super-user, rather than locking the machine.
- Win_screendestroy has been corrected to fix a problem which caused users to be logged off when exiting suntools.

4.1.2. Assembler

- The 50 character name limit has been removed; the new limit is 512 characters.
4.1.3. C Compiler

- The *atof* bug in the compiler and C library, which caused erroneous answers for very large or very small numbers, has been patched. A script is provided with the release tape to patch the C Library; the script is run as part of normal interim release installation.
- Code put out by the compiler to do loops like
  
  ```c
  for ( i=0; i<10; i++ )
  ```
  
  was marked with the incorrect line number (for dbx). This is repaired. The problem with marking the switch-statement selection code with the line number of the “switch(<exp>)” has also been fixed.
- Unsigned constant folding now works (better). This fixes the longstanding:
  
  ```c
  #define MAXINT (int)((unsigned)-1)<<1
  ```
  
  We now try a little harder to recover from certain syntax errors. This avoids the most common crash-&-burn mode error.
- The compiler now uses the “.skip” assembler directive, rather than many “.long 0” directives. This speeds up compilation of lines like:
  
  ```c
  static int x[100000] = { 1, 2, 3 };
  ```
  
  - A bug in accessing bit fields has been fixed.
  - Several changes have been made to avoid the ‘running-out-of-register’ problem often seen in FORTRAN compilations when compiling complicated expressions with all registers busy.
  - The bug with *unsigned char’s* and *unsigned short’s* being cast to *unsigned int* in cases like:
    
    ```c
    unsigned char x ( int a, b ){
      foo( (unsigned short)( a-b ) ); /* 1 */
      return( (unsigned char)( a+b ) ); /* 2 */
    }
    ```
    
    where the implicit conversion in parameter passing and value returning was done incorrectly has been fixed.
- The problem with floating-point relations in which one or both sides have complicated addresses (like subscripting) has been fixed.
- A fragment like:
  
  ```c
  { register float *fp; foo(*fp++); }
  ```
  
  could previously not be compiled, as the float-double conversion code could not recognize this addressing mode. This has been fixed.
- The ‘test’ in the fragment:
  
  ```c
  { double d; if (d) ....
  ```
  
  previously did the wrong thing for positive denormalized numbers with no significant bits in the first word. This has been fixed.
- The bug which caused ‘register float’ variables to generate bad code has been repaired.
• A bug which caused integer and floating point comparisons to be done in the wrong mode has been fixed. For example, the following:

```c
int a = 1;
double f = 1.1;
if (a < f) {
    printf("less\n");
} else {
    printf("more\n");
}
```

previously printed "more" instead of "less".

### 4.1.4. FORTRAN Compiler

• Single precision reals in PARAMETER statements are now actually set to the first word of a double precision version of the value. This fixes the problem with real parameters containing an erroneous value.

• Previously, if a program with 'do' loops nested more than 6 deep was compiled without the -O flag, the compilation would start using Address Registers as loop indices, causing the resulting code to loop infinitely. This has been fixed.

• A bug which caused 'f77 -o prog.f' to load a null file into prog.f has been fixed.

• Changes have been made to the FORTRAN library to make use of new intrinsics in the revised Sky Microcode.

• A call on an external is now recognized as defining its arguments in code motion.

• In evaluating 'x = f(x) **2' with -O set, f is no longer called twice.

### 4.1.5. Diag

• A revised version of diag is provided with this release. This new version supports Revision C Xylogics 450 PROMs and also has improved diagnostics. See Chapter 3 of this document for details and installation instructions.

### 4.1.6. Sky Driver

• A new Sky device driver that fixes problems with Sky interrupt handling is provided with this release.

### 4.1.7. SCSI Drivers

• New SCSI disk and tape drivers, and also a new SCSI boot block, are provided with this release. This fixes problems which occurred when a SCSI board was in a system that also had a 1/2" tape drive.
4.1.9. **Sky Microcode**

- A new version of the Sky microcode that fixes several problems with the previous version is provided in this release.
- A new `skyversion(8)` command is included in this release. `skyversion` reports the Sky version number of the microcode.

4.1.10. **Vpltdmp**

- Has been fixed to compute the number of bytes in a plot correctly.

4.2. **Documentation Changes**

4.2.1. **Diag**

- Supplemental `diag` documentation for use with the revised version of `diag` on the release tape is provided with in this document. Chapter 3 gives instructions for users with new disks, who must use this version to initialize their disks.

4.2.2. **Sendmail Installation**

- Users must now install the Mail system before coming up multi-user during UNIX installation, rather waiting until UNIX installation is complete. Instructions are given in the errata to the *System Manager's Manual*, see Chapter 7 of this document.
4.2.3. `ze(4)`

- Now includes information on remote dial-in/dial-out. See the `ze(4)` page in Chapter 8 of this document.

4.2.4. `/etc/skyinit`

- If you include a Sky Board in your Sun system, and you reboot before doing a ‘MAKEDEV sky’, you must either reboot again or run the `/etc/skyinit` program after doing the ‘MAKEDEV sky’ to load in the Sky Microcode. Documentation is included in the errata to the `System Manager's Manual`; see Chapter 7 of this document.

4.2.5. Kernel Configuration

- The pseudo-device ether must be included in the kernel configuration file. See the errata to the `System Manager's Manual`, Chapter 7 of this document.

- If the kernel configuration file contains sc it must also contain the corresponding st AND sd entries, whether or not the system contains both a SCSI disk and a SCSI tape. See the errata to the `System Manager's Manual`, Chapter 7 of this document.

- If you edit `/sys/h/param.h` you need to touch `/sys/KERNEL/param.c`. See the errata to the `System Manager's Manual`, Chapter 7 of this document.

4.2.6. Miscellaneous Errata in the System Manager's Manual

- Various errors in the `System Manager's Manual` have been fixed, such as some of the command-line syntax for remote installation, and the minimum size of `/pub`. See Chapter 7 of this document.

4.2.7. Miscellaneous Manual Pages

- See Chapter 8 of this document: `as(1), cc(1), cpp(1), screendump(1), screenload(1), malloc(3), ze(4S), acct(5), utmp(5), skyversion(8), and dinfo(8)`. These pages either describe new facilities, have been revised for this release, or were mistakenly omitted from the 1.1 manuals.
Chapter 5

Changes from the 1.2 Release Software

The following indicate changes made since the 1.2 Release to provide support for the Sun-2/50. User programs may need to be re-linked or re-compiled to take advantage of these changes.

5.1. Software Contents and Details of Changes

5.1.1. Model 50 Kernel Support

- Numerous changes have been made to the UNIX kernel to add support for the Sun-2/50.

5.1.2. PROMs for the Sun-2/50

- A new PROM set providing Sun-2/50 support is available.

5.2. Documentation Changes

5.2.1. Sun-2/50 Hardware Manual

- A version of the *Hardware Installation Manual for the Sun-2/50 Desktop SunStation* (Part Number: 800-1143-01) is provided with this release. Please use this manual when setting up any Sun-2/50's you receive with this release, and send us your comments!
Chapter 6

Changes from the 1.3 Release Software

The following indicate changes made since the 1.3 Release. Primarily, these changes were made to provide support for the Sun-2/160. User programs may need to be re-linked or re-compiled to take advantage of these changes.

6.1. Software Contents and Details of Changes

6.1.1. Model 160 Kernel Support

Numerous changes have been made to the UNIX kernel to add support for the Sun-2/160.

6.1.2. New Version of diag

diag now supports various 85 MByte SCSI disks (the Micropolis 1325, the Fujitsu M2243AS, and the Vertex V185). See Chapter 3 of this manual for instructions on using diag to initialize new disks. The diag(8S) page is also included in Chapter 8, and on the distribution tape.

6.1.3. New Version of tip

The new version of tip provided with 1.4 fixes many bugs related to tandem mode flow control; tip and uucp also now support the Hayes SmartModem 1200 (see below). Other changes are:

- The .tiprc and phones files now allow comments (#).
- The force and raise characters were previously set to ^P and ^A; they are now not set by default.
- Default parity is now 'none'.
- Halfduplex is another name for localecho.

A new tip(1C) manual page is included in Chapter 8, and on the distribution tape.
6.1.4. **Hayes Support for tip and uucp**

*Tip* and *uucp* now support the Hayes SmartModem 1200. Many modems are compatible with the Hayes, such as the Ventel EC1200-31 and EC1200-32 (aka 1200PLUS). Such modems often claim to be "AT" compatible or use the "AT" command set.

**Switch Settings**

The proper switch settings for the Hayes SmartModem 1200 for use with *tip* and *uucp* are listed below. For other Hayes compatible modems the switches should be set similarly — see your modem reference manual.

<table>
<thead>
<tr>
<th>Switch</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>UP</td>
<td>DO NOT force DTR true</td>
</tr>
<tr>
<td>2</td>
<td>DOWN</td>
<td>respond with digit result codes, NOT English words</td>
</tr>
<tr>
<td>3</td>
<td>DOWN</td>
<td>result codes WILL be sent</td>
</tr>
<tr>
<td>4</td>
<td>DOWN</td>
<td>DO NOT echo characters in command state</td>
</tr>
<tr>
<td>5</td>
<td>UP</td>
<td>DO answer incoming calls</td>
</tr>
<tr>
<td>6</td>
<td>UP</td>
<td>DO NOT force Carrier Detect true</td>
</tr>
<tr>
<td>7</td>
<td>UP</td>
<td>for connection to RJ11 modular jack</td>
</tr>
<tr>
<td>8</td>
<td>DOWN</td>
<td>enable command recognition</td>
</tr>
</tbody>
</table>

**Tip Support**

To use a Hayes modem with *tip*, the modem type ("at" attribute in `/etc/remote` file) should be specified as "hayes" or "at". The phone number ("pn" attribute) can contain any valid dial commands, see your modem manual for details. The most common dial commands are:

- 0-9 dial that number
- , pause for 2 seconds to wait for secondary dial tone
- P switch to pulse dialing
- T switch to tone dialing

By default *tip* uses tone dialing. Start the phone number with "P" to use pulse dialing.

To allow parameters of the dialer to be set, start the phone number with an 'S' to start a "Set" command. For example, to change the dial tone wait time, specify the phone number as:

```plaintext
:pn=S6=10DT4085551212:
```

Note that the Dial command must be specified explicitly.

**NOTE:** *Tip* can cycle through a set of phone numbers, trying each one until a connection is made. Previously it was common to separate the phone numbers with a comma, although this feature was never documented. Now, since comma is a valid dial character, phone numbers must be separated with a '|'. For example:

```plaintext
:pn=4085551212|4085551213|4085551214:
```
This is a potential compatibility problem and even affects people who are not using Hayes modems. They may have to change their `/etc/remote` file.

**Uucp Support**

To use a Hayes (or AT) compatible modem with `uucp` the modem type should be specified as ACUHAYES (or ACUAT). As with `tip`, the phone number can contain any valid dial commands and `uucp` defaults to tone dialing. However, `uucp` uses any alphabetic prefix of a phone number to look up a translation in the `L-devices` file. Therefore, to start a phone number with a letter insert a `"-"` before the letter. For example, to switch to pulse dialing, specify the phone number as

```
-P4085551212
```

An `L.sys` line that uses a Hayes modem might look like:

```
sun Any ACUHAYES 1200 -P9,4085551212 login:-EOT-login: uucp ssword:
whatever
```

To use the default of tone dialing simply omit the `"-P"`.

As with `tip`, `uucp` also allows `Set` commands in the phone number. Start the phone number with `"-S"` and don't forget to explicitly specify the `"D"` dial command.

### 6.1.5. Tape Retensioning

Quarter-inch tapes controlled by Archive (ar) or SCSI (st) controllers now may be retensioned. See the `mt(1)` page included in Chapter 8 for details.

### 6.1.6. MAKEDEV Bug Fix

MAKEDEV is now more rigorous about checking its arguments.

### 6.1.7. Sky Board Error Messages

Several customers have requested more information on the Sky Board error messages.

The Sky Floating Point Board does not use interrupts. Therefore, if the status register of the board indicates that interrupts are enabled or indicated, the board is in an error condition.

The Sky board status register looks something like this

```
[ ] [ ] [ ] [ ] | [ ] | [ ] | [ ]
```

<table>
<thead>
<tr>
<th>Other Functions</th>
<th>Interrupts</th>
<th>Programmed</th>
<th>Illegal Transfer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enabled</td>
<td>Interrupt</td>
<td>Interrupt</td>
<td></td>
</tr>
</tbody>
</table>

Since interrupts are not used on the board, the bottom three bits of the status register, the ones dealing with interrupts, should always be zero. Any deviation from this state causes the error messages below.
skyintr: Sky board unrecognized status, status = 0x%xx

This message indicates that one of the two interrupt bits in the status register is set and
interrupt enable is not. This is caused by a Sky Board initialization problem due to a design
flaw. The system is fine, but you'll probably continue to see the message.

The "status = 0x%xx" part of the message displays the value of the status register with four
hexadecimal digits preceded by a 0x to indicate that the digits are in hex. This is to help in
evaluating the error condition.

skyintr: Sky board interrupt enabled, status = 0x%xx

This message means that interrupt enabled bit in the status register is set, as it should not
be. The software attempts to clear this bit after indicating the error. If this message contin-
ues to appear, your system is probably hung.

6.1.8. Changes to the SunCore Graphics Package

This section is a summary of changes to the SunCore graphics package for the 1.4 Release.

6.1.8.1. SunCore — Added set_pick Function

The size of a square pick aperture can be specified so that the application program can set the
sensitivity of the PICK device.

6.1.8.2. SunCore — Additional Raster Fonts

There are now six raster (STRING precision) fonts available for SunCore.

6.1.8.3. SunCore — Sun/2-160 Support

SunCore can now run on /dev/cgtwo (Sun-2 color board) as well as all previously supported
display surfaces. A new view surface, cg2dd, has been added for the Sun-2/160 color board as a
raw device.

6.1.8.4. SunCore — Speed Improvements

Picking has been speeded up between 20% (for a display file of a few untransformed segments of
very few primitives) and 80% (for a display file of many fully transformed segments of many
primitives).

Transformations have been speeded up by 10% - 20%.

Other various improvements have been made in SunCore and the underlying libraries and com-
piilers, so a general (5% - 10%) improvement in performance should be seen in SunCore.
6.1.8.5. SunCore — Bug Fixes

- Calling the text() function with a space character as part of the text string, character_precision set to STRING, and font set to SPECIAL, previously generated a segmentation fault — this problem has been fixed.

- Calling the set_font() function with character_precision set to STRING, would previously use the default font the first time the function was called instead of using the requested font — this problem has been fixed.

- The actual width of fat vectors (line with line_width > 0) are now the same independent of the displayed slope of the line.

- Markers are now centered at the current position.

- set_zbuffer_cut now reports an error if the surface does not support hidden surfaces.

- All error messages are now sent to the standard error file.

- Calling the terminate_core() function while there were still many retained segments used to take a long time — this problem has been fixed.

- Calling the set_rdc_2() function on a raw bus2 surface no longer produces a segmentation fault when the z and y axes are not full scale. This problem also manifested itself by producing strange results from size_raster().

- Every line segment created by polyline() can now be picked.

- text() no longer produces a floating-point exception when the projection is perspective.

- The machine no longer hangs after a call to polygon_abss_3().

- Single pixel-high polygons no longer generate a segmentation fault.

- Horizontal edges of a polygon now create the same scan line segments independent of the directional sense of the polygon.

- The SKY version of the SunCore library now has a get_view_surface() function.

- The Pascal and FORTRAN wrappers for get_Mouse_state() now have the correct number of arguments.

- The problem of dragged rasters becoming invisible has been fixed.

- set_pickid() now sets the pickid for all subsequently created primitives in a segment, not just for the next primitive.

6.1.9. Changes to SunWindows

A new pixrect driver using cg2_ prefixes is available for the Sun-2 color board (/dev/cgtwo0). A variety of bugs have been fixed in the pixrect code.

6.2. Documentation Changes

A version of the Hardware Installation Manual for the Sun-2/160 Desktop SunStation (Part Number: 800-1144-01) is provided with this release. Please use this manual when setting up any Sun-2/160's you receive with this release, and send us your comments!

6.2.2. Miscellaneous Manual Pages

See Chapter 8 of this document. bessel(3F), bit(3F), time(3F), rand(3F), etime(3F), jdate(3F), stat(3F), and long(3F) are included here, as they were omitted from the FORTRAN Library Functions Section of the 1.1 FORTRAN and Pascal Manual. mem(4S) is also included; it has been revised to reflect VME bus support. Other pages include clear(1), diag(8S), mt(1), tip(1C), cgtwo(4S), ic(4S), and ftpusers(5).
Chapter 7

Errata Pages for 1.1 Manuals

The following pages list errata from the 1.1 System Manager's Manual for the Sun Workstation — Models 100U/150U (Part Number: 800-1109-01) and the System Manager's Manual for the Sun Workstation — Models 120/170 (Part Number: 800-1110-01). Note that there is one set for each manual.
Errata in the 1.1 Release of the System Manager's Manual for the Sun Workstation — Models 120/170

<table>
<thead>
<tr>
<th>Page(s)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>na</td>
<td>Kernel Configuration. In Section 3.3.12, the documentation says that the pseudo-device ether line is not needed in the configuration file for Workstations without Ethernet interfaces. Due to a bug in the current release, however, this line is currently mandatory for all systems, whether or not they have an Ethernet Board. If the line is left out, your subsequent make will fail with &quot;undefined _arpioctl error code 1&quot; and the resulting kernel won't boot.</td>
</tr>
<tr>
<td>na</td>
<td>Kernel Configuration. In Section 3.3.12, there is an undocumented restriction when configuring a kernel for systems with SCSI disk and/or SCSI tape. Along with the controller sc0 line, the kernel configuration file must include BOTH the disk sd0 AND tape st0 lines, whether or not the SCSI disk and SCSI tape are actually present in the system.</td>
</tr>
<tr>
<td>na</td>
<td>Kernel Configuration. If you edit the /sys/h/param.h file to tune your kernel configuration, you must touch(1) sys/KERNEL/param.c immediately afterwards for the changes to come into effect.</td>
</tr>
<tr>
<td>na</td>
<td>vfont files. The vfont files were included in the 1.1 Release. They are in the same tar file as the demos, manuals, and games. To load the files, see Section 4.14.</td>
</tr>
<tr>
<td>na</td>
<td>Sky Board. If you include a Sky Board in your Sun system, and you reboot before running the command 'MAKEDEV sky', you must either reboot again or run the /etc/skyinit program after doing the 'MAKEDEV sky' to load in the Sky Microcode. Proceed as follows:</td>
</tr>
<tr>
<td>3-16, 3-21, 3-24, 3-25, 3-27, 3-30</td>
<td>In Section 3.3.10, Using Setup to Configure Your System, the minimum and default sizes of the /pub partition as it is allocated by the setup program during first time UNIX installation was misquoted: the documentation says minimum /pub is 17 MBytes and default is 20 MBytes; minimum is in fact 20 MBytes, and default is in fact 24 MBytes.</td>
</tr>
<tr>
<td>3-32</td>
<td>Immediately before Booting the Full UNIX System (Section 3.3.11), you must install the Mail System as described in Chapter 5, System Set-up and Operation. Complete AT LEAST Sections 5.2, 5.2.1, 5.2.2, and 5.2.2.1 of Mail set-up. If you do not install /usr/lib/sendmail.cf before coming up multi-user, sendmail will try to run without the necessary files, and will leave one qf... and one af... file in the root directory.</td>
</tr>
<tr>
<td>Page(s)</td>
<td>Comments</td>
</tr>
<tr>
<td>--------</td>
<td>----------</td>
</tr>
<tr>
<td>5-10</td>
<td>Section 5.2.2.2, <em>Telling Sendmail your Domain Name</em>, includes a procedure for using the “CV” line in <code>sendmail.cf</code> for local mail routing. This feature does not currently work (bug).</td>
</tr>
<tr>
<td>6-2, 6-3</td>
<td>Section 6.1 (<em>Step 1: What to Save</em>) of the <em>Upgrading System Software</em> Chapter gives incorrect command syntax for <code>tar</code> file extraction from machines without local tape drives. Text for numbered step 1 reads: ``` # cd / # tar cfb - block_size dev/rtape# .??* dev/MAKEDEV.local etc lib usr/include usr/lib</td>
</tr>
<tr>
<td>3-7, 3-8, 3-12, 3-33, 4-5</td>
<td>A new version of <em>diag</em> with enhanced diagnostic and bad-block-handling capabilities is being shipped with the 1.3 distribution software. If you are installing a system with a new disk, use the instructions in Chapter 3 of this document with your 1.1 manual during installation; they will take care of the changes to noted pages.</td>
</tr>
<tr>
<td>4-4</td>
<td>There are two syntax errors in the section 4.3, <em>Setting up the Remote Host</em>. The lines given for <code>nd.local</code> are: <code>user 0 0 /dev/disk0g -1 -1 -1 son</code> They SHOULD be: <code>user 0 0 /dev/disk0g 0 -1 -1 son</code> Secondly, the <code>MAKEDEV</code> command reads: <code># MAKEDEV nd</code> It SHOULD read: <code># MAKEDEV nd10</code></td>
</tr>
<tr>
<td>Page(s)</td>
<td>Comments</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
</tr>
</tbody>
</table>
| 5-16 | There are three major items relating to UUCP. First, on page 5-16, the syntax for step 6 of UUCP installation is incorrect. Text for step 6 SHOULD read: Create the appropriate device for your modem with the following series of commands:  
# cd /dev  
# mknod cuao c 12 128  
# chmod 600 cuao  
# chown uucp cuao  
# mv ttya ttyd0  
The third line of the original series of commands has been deleted. Note that this command series supersedes the series printed in the Technical Support Bulletin Issue 1984-4 of 23 July 1984 (item 4.2), in which the final line designated a `ln` command rather than a `mv`.  
Secondly, `uust` may fail because it can’t open files due to permission problems. This can be fixed with the following command:  
# chmod 711 /usr/spool/uucp/{C.,D.,D.*systemname}  
Thirdly, one important point omitted from the discussion of setting up your modem for both dial-out and dial-in on the same serial port (Section 5.4.1), is that the flags bit in the kernel corresponding to the serial port you’re trying to set up has to be zero. This enables hardware carrier detect, so the Sun can tell when someone dials in or hangs up. If bit i of flags is set to 1, this tells the kernel that line i should be treated as hard-wired with carrier always present. In the case of device `/dev/serial0`, bit 0 (the 1’s bit) represents `ttya` and bit 1 (the 2’s bit) represents `ttys`. The default value of flags for `/dev/serial0` in the GENERIC kernel is 3, indicating software carrier for both ports a and b. To permit hardware carrier on `ttya`, flags should be changed to 2 or 0. You must also have the modem cabled correctly to the workstation: you need a cable with pins 1 through 8 and pin 20 all wired straight through. A full 25-pin ribbon cable also will work.  
Another omission from this discussion concerns Ven-Tel modems. If you’re using the Ven-Tel MD-212 modem for both dial-in and dial-out on the same serial port, you must have “WECO PROMs” in the modem (the term “WECO” refers to “Western Electric COmpany”). The default firmware that comes with the Ven-Tel modem is non-WECO because it disables some of the the standard modem control signals. You can obtain the WECO PROMs by contacting Ven-Tel in Santa Clara, CA, (408)727-5721.  
The foregoing discussion only applies if you are setting up a modem for both dial-in and dial-out on the same serial port. If you only want to dial-out, you should skip step 7 of the instructions in Section 5.4.1 (which tells you to edit `/etc/ttya` to enable logins on `/dev/ttyd0`). The last paragraph of this section describes the procedure for setting up the line only to receive calls.  
Page for `dinfo(8)` was not included in the 1.1 Manual. |
<p>| (8)23a | |</p>
<table>
<thead>
<tr>
<th>Page(s)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>7-18, 7-19</td>
<td>In Section 7.11 on <em>Asynchronous Serial Ports</em>, the SCSI I/O channels and their connectors were mismatched. The final text on page 7-18 <strong>SHOULD</strong> read: “On the SCSI Board, I/O channels C and D appear on connector <strong>J2</strong> (SIO-S2 and SIO-S3 respectively); channels E and F appear on 50-pin connector <strong>J1</strong> (SIO-S0 and SIOS-1 respectively).” On page 7-19, the tables for J1 and J2 are reversed.</td>
</tr>
</tbody>
</table>
## Errata in the 1.1 Release

of the

System Manager's Manual for the Sun Workstation — Models 100U/150U

<table>
<thead>
<tr>
<th>Page(s)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>na</td>
<td>Kernel Configuration. In Section 4.3.12, the documentation says that the <strong>pseudo-device ether</strong> line is not needed in the configuration file for Workstations without Ethernet interfaces. Due to a bug in the current release, however, this line is currently mandatory for all systems, whether or not they have an Ethernet Board. If the line is left out, your subsequent <code>make</code> will fail with &quot;undefined _arpoctl error code 1&quot; and the resulting kernel won't boot.</td>
</tr>
<tr>
<td>na</td>
<td>Kernel Configuration. If you edit the <code>/sys/h/param.h</code> file to tune your kernel configuration, you must <code>touch(1)</code> <code>/sys/KERNEL/param.c</code> immediately afterwards for the changes to come into effect.</td>
</tr>
<tr>
<td>na</td>
<td><code>vfont</code> files. The <code>vfont</code> files were included in the 1.1 Release. They are in the same <code>tar</code> file as the demos, manuals, and games. To load the files, see Section 4.3.13.</td>
</tr>
<tr>
<td>na</td>
<td>Sky Board. If you include a Sky Board in your Sun system, and you reboot before running the command <code>MAKEDEV sky</code>, you must either reboot again or run the <code>/etc/skyinit</code> program after doing the <code>MAKEDEV sky</code> to load in the Sky Microcode. Proceed as follows:</td>
</tr>
<tr>
<td></td>
<td><code># cd /dev</code></td>
</tr>
<tr>
<td></td>
<td><code># MAKEDEV sky</code></td>
</tr>
<tr>
<td></td>
<td><code># cd /etc</code></td>
</tr>
<tr>
<td></td>
<td><code># skyinit</code></td>
</tr>
<tr>
<td>4-15, 4-19, 4-22, 4-23, 4-25, 4-28</td>
<td>[message to the effect that z # words of microcode loaded]</td>
</tr>
<tr>
<td>4-30</td>
<td>In Section 4.3.10, <em>Using Setup to Configure Your System</em>, the minimum and default sizes of the <code>/pub</code> partition as it is allocated by the <code>setup</code> program during first time UNIX installation was misquoted: the documentation says minimum <code>/pub</code> is 17 MBytes and default is 20 MBytes; minimum is in fact 20 MBytes, and default is in fact 24 MBytes.</td>
</tr>
<tr>
<td>6-10</td>
<td>Immediately before <em>Booting the Full UNIX System</em> (Section 4.3.11), you must install the Mail System as described in Chapter 5, <em>System Set-up and Operation</em>. Complete AT LEAST Sections 6.2, 6.2.1, 6.2.2, and 6.2.2.1 of Mail set-up. If you do not install <code>/usr/lib/sendmail.cf</code> before coming up multi-user, <code>sendmail</code> will try to run without the necessary files, and will leave one <code>af</code>... and one <code>af</code>... file in the root directory.</td>
</tr>
<tr>
<td></td>
<td>Section 6.2.2.2, <em>Telling Sendmail your Domain Name</em>, includes a procedure for using the &quot;CV&quot; line in <code>sendmail.cf</code> for local mail routing. This feature does not currently work (bug).</td>
</tr>
<tr>
<td>Page(s)</td>
<td>Comments</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
</tr>
</tbody>
</table>
| 7-2, 7-3 | Section 7.1 *(Step 1: What to Save)* of the *Upgrading System Software* Chapter gives incorrect command syntax for *tar* file extraction from machines without local tape drives. Text for numbered step 1 reads:  
   # cd /  
   # tar cf - block_size dev/rtape# . ??* dev/MAKEDEV.local  
   etc lib usr/includeusr/lib | rsh remote_host  
   dd of=/dev/tape# obs=block_size b  
   It SHOULD read:  
   # cd /  
   # tar cf - block_size . ??* dev/MAKEDEV.local  
   etc lib usr/includeusr/lib | rsh remote_host  
   dd of=/dev/tape# obs=block_size b  
   Text for numbered step 2 reads:  
   # tar cf - block_size dev/rtape#usr/{spool,local,  
   usera, userb, userc, userd...} |  
   rsh remote_host dd of=/dev/tape# obs=block_size b  
   It SHOULD read:  
   # tar cf - block_size usr/{spool,local,  
   usera, userb, userc, userd...} |  
   rsh remote_host dd of=/dev/tape# obs=block_size b  
   In both cases, the "dev/rtape#" before the pipeline is unnecessary. |
| 4-7, 4-8, 4-10, 4-30, 5-5 | A new version of *diag* with enhanced diagnostic and bad-block-handling capabilities is being shipped with the 1.3 distribution software. If you are installing a system with a new disk, use the instructions in Chapter 3 of this document with your 1.1 manual during installation; they will take care of the changes to noted pages. |
| 5-4 | There are two syntax errors in the section 5.3, *Setting up the Remote Host*. The lines given for *nd.local* are:  
   user 0 0 /dev/disk0g -1 -1 -1  
   son  
   They SHOULD be:  
   user 0 0 /dev/disk0g 0 -1 -1  
   son  
   Secondly, the *MAKEDEV* command reads:  
   # MAKEDEV nd  
   It SHOULD read:  
   # MAKEDEV nd10 |
<table>
<thead>
<tr>
<th>Page(s)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>6-16</td>
<td>There are three major items relating to UUCP. First, on page 6-16, the syntax for step 6 of UUCP installation is incorrect. Text for step 6 SHOULD read: Create the appropriate device for your modem with the following series of commands:</td>
</tr>
</tbody>
</table>
|         |     # cd /dev  
|         |     # mknod cuao c 12 128  
|         |     # chmod 600 cuao  
|         |     # chown uucp cuao  
|         |     # mv ttysa ttyd0  
|         | The third line of the original series of commands has been deleted. Note that this command series supersedes the series printed in the Technical Support Bulletin Issue 1984-4 of 23 July 1984 (item 4.2), in which the final line designated a ln command rather than a mv.  
|         | Secondly, uucp may fail because it can’t open files due to permission problems. This can be fixed with the following command: |
|         |     # chmod 711 /usr/spool/uucp/{C,D,D.systemname}  
|         | Thirdly, one important point omitted from the discussion of setting up your modem for both dial-out and dial-in on the same serial port (Section 6.4.1), is that the flags bit in the kernel corresponding to the serial port you’re trying to set up has to be zero. This enables hardware carrier detect, so the Sun can tell when someone dials in or hangs up. If bit i of flags is set to 1, this tells the kernel that line i should be treated as hard-wired with carrier always present. In the case of device za0, bit 0 (the 1’s bit) represents ttys and bit 1 (the 2’s bit) represents ttyb. The default value of flags for za0 in the GENERIC kernel is 3, indicating software carrier for both ports a and b. To permit hardware carrier on ttys, flags should be changed to 2 or 0. You must also have the modem cabled correctly to the workstation: you need a cable with pins 1 through 8 and pin 20 all wired straight through. A full 25-pin ribbon cable also will work.  
|         | Another omission from this discussion concerns Ven-Tel modems. If you’re using the Ven-Tel MD-212 modem for both dial-in and dial-out on the same serial port, you must have “WECO PROMs” in the modem (the term “WECO” refers to “Western Electric Company”). The default firmware that comes with the Ven-Tel modem is non-WECO because it disables some of the the standard modem control signals. You can obtain the WECO PROMs by contacting Ven-Tel in Santa Clara, CA, (408)727-5721.  
|         | The foregoing discussion only applies if you are setting up a modem for both dial-in and dial-out on the same serial port. If you only want to dial-out, you should skip step 7 of the instructions in Section 6.4.1 (which tells you to edit /etc/ttys to enable logins on /dev/ttyd0). The last paragraph of this section describes the procedure for setting up the line only to receive calls. |

(8)23a Page for dinfo(8) was not included in the 1.1 Manual.
Chapter 8

Insert Pages for 1.1 Reference Manuals

The following pages are reference manual pages — that is, manual pages from the sections numbered 1 through 8 of the 1.1 Release *User's Manual for the Sun Workstation* (Part Number: 800-1107-01), the *System Interface Manual for the Sun Workstation* (Part Number: 800-1108-01), *System Manager's Manual for the Sun Workstation* (Part Numbers: 800-1109-01 [100U/150U] and 800-1110-01 [120/170]), and *FORTRAN and Pascal for the Sun Workstation* (Part Number: 800-1114-01) — which have been written since the 1.1 Release, have been revised since 1.1, or which were unintentionally not included in the 1.1 manuals.

How you handle the pages is up to you. We recommend inserting them in your 1.1 manuals (copying them if necessary); they are numbered accordingly.

**New Pages**

- ` screendump(1)` — dump frame buffer image
- ` screenload(1)` — restore frame buffer image
- ` cgtwo(4S)` — Sun-2 color graphics interface
- ` ic(4S)` — Sun-2 10 Mb/s Ethernet interface
- ` ftptusers(5)` — list of users prohibited by ftp
- ` skyversion(8)` — print the SKYFFP board microcode version number
- ` dkinfo(8)` — report information about a disk’s geometry and partitioning

**Revised Pages**

- ` as(1)` — mc68000 assembler
  [Note that the 50 character name limit has extended to 512 characters.]
- ` cc(1)` — C compiler
- ` clear(1)` — clear screen
  [Includes new -c option for clearing color map.]
- ` cpp(1)` — C language preprocessor
- ` mt(1)` — magnetic tape manipulating program
  [Includes new retensioning option.]
- ` tip(1C)` — connect to a remote system
  [Revised to reflect bug fixes; updated]
malloc(3) — memory allocator
[The page included in the 1.1 manual described a non-existent routine.]

mem(4S) — main memory and bus I/O space
[Now includes description of special files for VME bus support.]

zs(4S) — zilog 8530 SCC serial communications driver
[Now includes information on dial-in/dial-out support on the same line.]

diag(8S) — stand-alone disk initialization and diagnosis
[Revised to reflect additional capabilities of new version.]

Omitted Pages

acct(5) — execution accounting file
utmp(5) — login records
bessel(3F) — bessel functions
bit(3F) — setbit functions
time(3F) — return system time
rand(3F) — return random values
etime(3F) — return elapsed execution time
fdate(3F) — return date and time in an ASCII string
stat(3F) — get file status
long(3F) — integer object conversion
NAME
screendump — dump frame buffer image

SYNOPSIS
screendump | -c | -f display |

DESCRIPTION
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.

By default, screendump attempts first to output the contents of the console frame buffer. If no console frame buffer is found, screendump attempts to output the contents of a color frame buffer. The -c option selects a color frame buffer directly. Heuristics are applied to locate the color frame buffer the user is most likely interested in. The -f option explicitly sets the name of the desired frame buffer device.

The utility program screenload displays Sun standard rasterfiles on an appropriate Sun Workstation monitor. Output filters exist for printing standard monochrome rasterfiles on Versatec V-80 electrostatic plotters.

OPTIONS
-c Dump a color frame buffer without trying the console frame buffer.
-f display Use display as the device name of the display.

EXAMPLES
tutorial% screendump >save.this.image
writes the current contents of the console frame buffer into the file save.this.image.

tutorial% screendump -f /dev/cgone0 >save.color.image
writes the current contents of the frame buffer /dev/cgone0 into the file save.color.image. This has the same effect as the -c option for the most common display configuration.

tutorial% screendump | lpr -Pversatec -v
sends a rasterfile containing the current frame buffer to the lineprinter, selecting the printer named "versatec" and the "v" output filter (see /etc/printcap).

FILES
/dev/fb Default name of console display frame buffer.
/dev/cgone0 Default name of the Sun-1 color display frame buffer.

SEE ALSO
screenload(1), lpr(1)
NAME
screenload — restore frame buffer image

SYNOPSIS
    screenload [ -d ] [ -f display ] [ -l index ] [ -b ] [ -g ] [ -w ] [ -h# [### [...]] ] file

DESCRIPTION
    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). Screenload
    normally displays rasterfiles on the console display, but displays them on some heuristically deter-
    mined color display if it finds no console display. Screenload displays color rasterfiles only on a
    color monitor. Screenload is able to display monochrome rasters on a color display.

    If the dimensions of the image contained in the input data are smaller than the actual resolution
    of the display (e.g. loading a 1024-by-800 Sun-1 image on a 1152-by-900 Sun-2 display), screen-
    load centers the image on the actual workstation screen and fills the border area with solid black
    (by default). Various options may be used to change the fill pattern.

    If the dimensions of the image contained in the input data are larger than the actual resolution
    of the display, screenload clips the right and bottom edges of the input image.

    If there is an optional filename argument, screenload reads its input data from that file. If no
    filename argument is given, screenload reads its input data from the standard input.

    The utility program screendump creates Sun standard rasterfiles from the display on a Sun
    Workstation monitor.

OPTIONS
    -d    Verify that display dimensions and input data image dimensions match, and print warn-
          ing messages if they do not.

    -f display
          Use display as the name of the frame buffer device.

    -l index
          If the image is smaller than the display, use index (0 <= index < 256) as the index value
          into the color map for the foreground color of the border fill pattern. The default index
          value is 255.

    -b    If the input data dimensions are smaller than the display dimensions, fill the border with
          a pattern of solid ones (default). On a monochrome display this results in a black border;
          on a color display the color map value selected by the -l option determines the border
          color.

    -g    If the input data dimensions are smaller than the display dimensions, fill the border with
          a pattern of "desktop grey". On a monochrome display this results in a border matching
          the background pattern used by the SunWindows system; on a color display the color
          map value selected by the -l option determines the foreground border color, though the
          pattern is the same as on a monochrome display.

    -w    If the input data dimensions are smaller than the display dimensions, fill the border with
          a pattern of solid zeros. On a monochrome display this results in a white border; on a
          color display the color map value at index 0 determines the border color.

    -h#   If the input data dimensions are smaller than the display dimensions, fill the border with
          the bit pattern described by the following # 16-bit hexadecimal constants. Note that a
          "1" bit is black and a "0" bit is white on the monochrome display; on a color display the
          color map value selected by the -l option determines the border foreground color. The
          number of hex constants in the pattern is limited to 16.
EXAMPLES

tutorial% screenload saved.display.image
reloads the raster image contained in the file saved.display.image on the display type indicated by
the rasterfile header in that file.

tutorial% screenload -f /dev/cgone0 monochrome.image
reloads the raster image in the file monochrome.image on the frame buffer device /dev/cgone0.

tutorial% screenload -h1 ffff sun_1.saved.image
is equivalent to the -b option (fill border with black), while

tutorial% screenload -h4 8888 8888 2222 2222 sun_1.saved.image
is equivalent to the -g option (fill border with desktop grey).

FILES
/dev/fb        Default name of console display frame buffer
/dev/cgone0    Default name of SUN-1 color display frame buffer

SEE ALSO
screendump(1)
NAME
cgtwo — Sun-2 color graphics interface

SYNOPSIS
cgtwo at mb0 csr vme busmem 0x400000 priority 3

DESCRIPTION
The cgtwo interface provides access to the Sun-2 color graphics controller board, which is normally supplied with a 19" 60 Hz non-interlaced color monitor. It provides the standard frame buffer interface as defined in fbio(4S).

The hardware consumes 4 megabytes of VME bus address space. The board starts at standard address 0x400000. The board must be configured for interrupt level 3.

FILES
/dev/cgtwo[0-9]

SEE ALSO
mmap(2), fbio(4S)
NAME
ie — Sun-2 10 Mb/s Ethernet interface

SYNOPSIS
device ie0 at mb0 csr 0x88000 priority 3
device ie0 at mb0 csr vme virt 0xeee3000 priority 3

DESCRIPTION
The ie interface provides access to a 10 Mb/s Ethernet network through a Sun-2 controller. For a general description of network interfaces see if(4N).

Of the synopsis lines above, the first line specifies the first Sun-2 Ethernet controller on a Sun-2/120 or Sun-2/170; the second line specifies the first Sun-2 Ethernet controller on a Sun-2/50 or Sun-2/160.

The remainder of this information will be provided at a later time.
NAME
ftpusers — list of users prohibited by ftp

SYNOPSIS
/usr/etc/ftpusers

DESCRIPTION
ftpusers contains a list of users who cannot access this system using the ftp(1) program.
ftpusers contains one user name per line.

SEE ALSO
ftp(1), ftpd(8C)
NAME
skyversion — print the SKYFP board microcode version number

SYNOPSIS
/usr/etc/skyversion

DESCRIPTION
skyversion obtains from the SKYFP board the Sky version number of the microcode currently
loaded and prints the result on the standard output.

DIAGNOSTICS
The Sky version number operation code used to implement this command is not available for
microcode releases earlier than Sky release 3.00. The result in this case is unpredictable and is
either a nonmeaningful version number or a message indicating that no version number is avail-
able. Meaningful version numbers are of the form n.dd where n ≥ 3.
NAME
dkinfo — report information about a disk's geometry and partitioning

SYNOPSIS
/etc/dkinfo disk[partition]

DESCRIPTION
Dkinfo gives the total number of cylinders, heads, and sectors or tracks on the specified disk, and gives this information along with the starting cylinder for the specified partition. If no partition is specified on the command line, dkinfo reports on all partitions.

The disk specification here is a disk name of the form xzn, where x is the controller device abbreviation (ip, xy, etc.) and n is the disk number. The partition specification is simply the letter used to identify that partition in the standard UNIX nomenclature. For example, '/etc/dkinfo xy0' reports on the first disk in a system controlled by a Xylogics controller; '/etc/dkinfo xy0g' reports on the seventh partition of such a disk.

You must be superuser to run dkinfo.

EXAMPLE
A request for information on my local disk, an 84 MByte disk controlled by a Xylogics 450 controller, might look like this:

# /etc/dkinfo xy0
xy0: Xylogics 450 controller at addr ee40, unit # 0
586 cylinders 7 heads 32 sectors/track
a: 15884 sectors (70 cyls, 6 tracks, 12 sectors)
   starting cylinder 0
b: 33440 sectors (149 cyls, 2 tracks)
   starting cylinder 71
c: 131284 sectors (586 cyls)
   starting cylinder 0
d: No such device or address
e: No such device or address
f: No such device or address
g: 81760 sectors (365 cyls)
   starting cylinder 221
h: No such device or address
#

SEE ALSO
dk(4), diag(8)
NAME
as - mc68000 assembler

SYNOPSIS
as | -d2 | -J | -L | -R | -o objfile | file

DESCRIPTION
As translates assembly code in the named file into executable object code in the specified objfile.
All undefined symbols in the assembly are treated as global.
The output of the assembly is left in the file objfile. If the -o flag is omitted, then file a.out is used.

OPTIONS
-d2 specifies that instruction offsets which involve forward or external references, and which have sizes unspecified in the assembly language are two bytes long. The default is four bytes. See also -J.
-L Save defined labels beginning with an 'L', which are normally discarded to save space in the resultant symbol table. The compilers generate such temporary labels.
-J Use short (pc-relative) branches to resolve jump's and jsr's to externals. This is for compact programs which cannot use the -d2 flag because of large program relocation.
-R Make initialized data segments read-only by concatenating them to the text segments. This eliminates the need to run editor scripts on assembly code to make initialized data read-only and shared.

FILES
/tmp/as* default temporary file

SEE ALSO
ld(1), nm(1), adb(1), dbx(1), a.out(5)

BUGS
Should assemble standard input with no arguments.
The Pascal compiler (pc(1)) qualifies a nested procedure name by chaining the names of the enclosing procedures. This sometimes results in names long enough to abort the assembler, which currently limits identifiers to 512 characters.
NAME
cc - C compiler

SYNOPSIS
-efsingle | -fsky | -S | -E | -C | -o output | -D name=def] 
-D name | -U name | -I dir | -B string | -t p012 | file ...

DESCRIPTION
Cc is the UNIX C compiler which translates programs written in the C programming language
into executable load modules, or into relocatable binary programs for subsequent loading with
the ld(1) linker.

OPTIONS
The following options are interpreted by cc. See ld(1) for load-time options.

-c Compile only: suppress the loading phase of the compilation, and force an object file to
be produced even if only one program is compiled.

-g Have the compiler produce additional symbol table information for dbx(1). Also pass
the -lg flag to ld(1).

-go Have the compiler produce additional symbol table information in an older format
which is used by the adb(1) debugger. Also, pass the -lg flag to ld(1).

-w Suppress warning messages.

-p Produce profiling code to count the number of times each routine is called. If loading
takes place, replace the standard startup routine by one that automatically calls moni-
tor(3) and use a special profiling library in lieu of the standard C library. When the
program is run, the file mon.out is created. An execution profile can then be generated
by use of prof(1).

-pg Produce profiling code in the manner of -p, but invokes a run-time recording mechan-
ism that keeps more extensive statistics and produces a gmon.out file at normal termina-
tion. gprof(1) generates an execution profile.

-O [optflags]
Use the object code optimizer to improve the generated code. If optflags appears, it is
included in the command line used to run the optimizer. This can be used to pass it
option flags.

-A [asmflags]
Pass asmflags to as in its command line. For example, -A -J can be used to pass the
-J option, which causes the assembler to use short PC-relative instructions for subrout-
tine calls.

-R Passed on to as, making initialized variables shared and read-only.

-efsingle
Use single-precision arithmetic in computations involving only float numbers — that is,
do not convert everything to double which is the default. Note that floating-point
parameters are still converted to double-precision, and functions which return values
still return double-precision values. Certain programs run much faster using this option,
but be aware that some significance can be lost due to lower precision intermediate
values.

-fsky Generate code which assumes the presence of a SKY floating-point processor board.
Programs compiled with this option can only be run in systems that have a SKY board
installed. Programs compiled without the -fsky option will use the SKY board if it is
present, but won't run as fast as they would if the -fsky option were used. If any part
of a program is compiled using the -fsky option, you must also use this option when
linking with the cc command, since a different set of startup routines is used.

-S Compile the named C programs, and leave the assembler-language output on corresponding files suffixed `.s`.

-E Run only the C preprocessor on the named C programs, and send the result to the standard output.

-C Prevent the C preprocessor from removing comments.

-o output
Name the final output file output. If this option is used, the file a.out is left undisturbed.

-Dname=def
-Dname
Define name to the preprocessor, as if by `#define`. If no definition is given, the name is defined as "1".

-Uname
Remove any initial definition of name.

-Idir `<#include` files whose names do not begin with `/` are always sought first in the directory of the file argument, then in directories named in -I options, then in the /usr/include directory.

-Bstring
Find substitute compiler passes in the files named string with the suffixes cpp, ccom and c2. If string is empty, use a standard backup version.

-t[p012]
Find only the designated compiler passes in the files whose names are constructed by a -B option. In the absence of a -B option, the string is taken to be /usr/new/. The letter/number combinations that can be specified for the -t option have the meanings:

```
 p  cpp — the C preprocessor.
 0  ccom — both phases of the C compiler, but not the optimizer.
 1  Ignored in this system — this option would be for the second phase of a two-phase compiler but in the Sun system, ccom includes both phases.
 2  c2 — the object code optimizer.
```

In addition to the many options, cc accepts several types of files. Files whose names end with `.c` are taken to be C source programs; they are compiled, and each resulting object program is left in the current directory, with the same name as the source file, except that the `.c` suffix is replaced by `.o`. In the same way, files whose names end with `.s` are taken to be assembly source programs and are assembled, producing a `.o` file.

Other arguments are taken to be loader option arguments, object programs, or libraries of object programs. Unless -c, -S, or -E is specified, these programs and libraries, together with the results of any compilations or assemblies specified, are loaded (in the order given) to produce an executable program named a.out. The name a.out can be overridden with the loader's -o name option.

If a single C program is compiled and loaded all at once, the intermediate `.o` file is deleted.

FILES

```
file.c  C source file
file.s  assembler source file
file.o  object file
file.a  library of object files
a.out  executable output file
```
/	mp/ctm  temporary
/lib/cpp    preprocessor
/lib/ccom   compiler
/lib/c2     optional optimizer
/lib/crt0.o runtime startoff
/lib/mcrt0.o startoff for profiling
/usr/lib/gcrt0.o startoff for gprof-profiling
/lib/fcrt0.o SKY runtime startoff
/lib/fmcrt0.o SKY startoff for profiling
/usr/lib/fgcrt0.o SKY startoff for gprof-profiling
/lib/libc.a standard library, see intro(3)
/usr/lib/libc_p.a profiling library, see intro(3)
/usr/include standard directory for '#include' files
mon.out     file produced for analysis by prof(1)
gmon.out    file produced for analysis by gprof(1)

SEE ALSO
UNIX Programming in Programming Tools for the Sun Workstation
monitor(3), prof(1), gprof(1), adb(1), ar(1), ld(1), dbx(1), as(1), diff(1), cpp(1)

DIAGNOSTICS
The diagnostics produced by C itself are intended to be self-explanatory. Occasional obscure
messages may be produced by the preprocessor, assembler, or loader.

BUGS
Options which are identical to cc options cannot be passed through to the assembler, optimizer,
or loader.
NAME
clear — clear screen

SYNOPSIS
clear

DESCRIPTION
Clear clears your screen if this is possible. It looks in the environment for the terminal type and then in /etc/termcap to figure out how to clear the screen.

OPTIONS
-c  Clear the color map on color screens.

FILES
/etc/termcap  terminal capability data base
NAME

cpp — the C language preprocessor

SYNOPSIS

```
/lib/cpp [-P -C -Uname -Dname -Dname=def -Idir ] [ifile ofile ]
```

DESCRIPTION

Cpp is the C language preprocessor which is invoked as the first pass of any C compilation using
the cc(1) command (cpp may optionally be invoked as the first pass of a FORTRAN 77 or Pascal
compilation — see f77(1) or pc(1)). Thus the output of cpp is designed to be in a form accept-
able as input to the next pass of the C compiler. The preferred way to invoke cpp is through the
cc(1) command. See m4(1) for a general macro processor.

Cpp optionally accepts two file names as arguments. ifile and ofile are respectively the input
and output for the preprocessor. They default to standard input and standard output if not sup-
plied.

OPTIONS

- -P Preprocess the input without producing the line control information used by the next
pass of the C compiler.

- -C Pass all comments (except those which appear on cpp directive lines) through the prepro-
cessor. By default, cpp strips C-style comments.

- -Uname

Remove any initial definition of name, where name is a reserved symbol that is
predefined by the particular preprocessor. The current list of these possibly reserved
symbols includes:

- operating system: ibm, gcos, os, tss, unix
- hardware: interdata, pdp11, u370, u3b, vax, mc68000

UNIX System variant: RES, RT, sun

- -Dname

Define name as 1 (one). This is the same as if a -Dname=1 option had appeared on the
cpp command line, or as if a #define name 1 line had appeared in the source file that
cpp is processing.

- -Dname=def

Define name as if by a #define directive. This is the same as if a #define name def
line had appeared in the source file that cpp is processing.

- -Idir

Change the algorithm for searching for #include files whose names do not begin with /
to look in dir before looking in the directories on the standard list. Thus, #include files
whose names are enclosed in "" will be searched for first in the directory of the ifile
argument, then in directories named in -I options, and last in directories on a standard
list. For #include files whose names are enclosed in <>, the directory of the ifile
argument is not searched. See the section entitled Details of the CPP Preprocessor, below,
for exact details of the search order.

- -R Allow recursive macros.

CPP DIRECTIVES

All cpp directives start with lines begun by #. The directives are:

- #define name token-string

Replace subsequent instances of name with token-string.

- #define name( arg, ..., arg) token-string

Notice that there can be no space between name and the '. Replace subsequent
instances of name followed by a '(', a list of comma-separated tokens, and a ')' by token-
string where each occurrence of an arg in the token-string is replaced by the
corresponding token in the comma-separated list.

```c
#undef name
```
Forget the definition of `name` (if any) from now on.

```c
#include "filename"
#include <filename>
```
Include at this point the contents of `filename` (which is then run through `cpp`). When the `<filename>` notation is used, `filename` is only searched for in the standard places. See the `-I` option above for more detail.

```c
#line integer-constant "filename"
```
Generate line control information for the next pass of the C compiler. `Integer-constant` is interpreted as the line number of the next line and `filename` is interpreted as the file where it comes from. If "filename" is not given, the current file name is unchanged.

```c
#endif
```
Ends a section of lines begun by a test directive (`#if`, `#ifdef`, or `#ifndef`). Each test directive must have a matching `#endif`.

```c
#ifdef name
```
The lines following will appear in the output if and only if `name` has been the subject of a previous `#define` without being the subject of an intervening `#undef`.

```c
#ifndef name
```
The lines following will not appear in the output if and only if `name` has been the subject of a previous `#define` without being the subject of an intervening `#undef`.

```c
#if constant-expression
```
Lines following will appear in the output if and only if the `constant-expression` evaluates to nonzero. All binary non-assignment C operators, the `?:` operator, the unary `-`, `,`, and `*` operators are all legal in `constant-expression`. The precedence of the operators is the same as defined by the C language. There is also a unary operator `defined`, which can be used in `constant-expression` in these two forms: `defined ( name )` or `defined name`. This allows the effect of `#ifdef` and `#ifndef` in a `#if` directive. Only these operators, integer constants, and names which are known by `cpp` should be used in `constant-expression`. In particular, the `elseof` operator is not available.

```c
#else
```
Reverses for the following lines the notion of the test directive currently in effect. So if lines previous to this directive are ignored, the following lines will appear in the output, and vice versa.

The test directives and corresponding `#else` directives can be nested.

**DETAILS OF THE C PREPROCESSOR**

*Directory search order* for `#include` files is:

1. the directory of the file which contains the `#include` request (that is, `#include` is relative to the file being scanned when the request is made)
2. the directories specified by `-I` options, in left-to-right order.
3. the standard directory(s) (`/usr/include` for the Sun system).

**Special Names:** Two special names are understood by `cpp`. The name `__LINE__` is defined as the current line number (a decimal integer) as known by `cpp`, and `__FILE__` is defined as the current file name (a C string) as known by `cpp`. They can be used anywhere (including in macros) just as any other defined name.

A `newline` terminates a character constant or quoted string.
An escaped newline (that is, a backslash immediately followed by a newline) may be used in the body of a `#define` statement to continue the definition onto the next line. The escaped newline is not included in the macro body.

Comments are removed (unless the `-C` option is used on the command line). Comments are also ignored, except that a comment terminates a token.

Macro formal parameters are recognized in `#define` bodies even inside character constants and quoted strings. The output from:

```
#define foo(a) "\a"
foo(bar)
```

is the seven characters `\bar` (space, single-quote, escape character, b, a, r, single-quote). Macro names are not recognized inside character constants or quoted strings during the regular scan. Thus:

```
#define foo bar
printf("foo");
```

does not expand 'foo' in the second line, because it is inside a quoted string which is not part of a `#define` macro definition.

Macros are not expanded while processing a `#define` or `#undef`. Thus:

```
#define foo bletch
#define bar foo
#undef foo
bar
```

produces 'foo'. The token appearing immediately after a `#ifdef` or `#ifndef` is not expanded.

Macros are not expanded during the scan which determines the actual parameters to another macro call. Thus:

```
#define reverse(first,second)second first
#define greeting hello
reverse(greeting,
#define greeting goodbye
)
```

produces 'goodbye' (and warns about the redefinition of 'greeting').

Incompatibility: The slash `/` in `a=*/b` is interpreted as the first character of the pair `*/` which introduces a comment, rather than as the second character of the divide-and-replace operator `=*/`. This incompatibility reflects the change in the C language which made `a=*/b` the preferred way to write such a statement if the meaning `a=a/ *b` is intended.

FILES

```
/usr/include
```

standard directory for `#include` files

SEE ALSO

cc(1), m4(1).

DIAGNOSTICS

The error messages produced by `cpp` are intended to be self-explanatory. The line number and filename where the error occurred are printed along with the diagnostic.

NOTES

When newline characters were found in argument lists for macros to be expanded, previous versions of `cpp` put out the newlines as they were found and expanded. The current version of `cpp` replaces these newlines with blanks to alleviate problems that the previous versions had when this occurred.

Sun Release 1.4  Last change: 9 August 1984
NAME

mt — magnetic tape manipulating program

SYNOPSIS

mt [ -f tapename ] command [ count ]

DESCRIPTION

mt sends commands to a magnetic tape drive. If tapename is not specified, the environment variable TAPE is used; if TAPE does not exist, mt uses the device /dev/rmt12. Note that tapename must reference a raw (not block) tape device. By default mt performs the requested operation once. Operations may be performed multiple times by specifying count.

The available commands are listed below. Only as many characters as are required to uniquely identify a command need be specified.

  eof, weof
      Write count end-of-file marks at the current position on the tape.

  fsf
      Forward space count files.

  fcr
      Forward space count records.

  bsf
      Back space count files.

  bcr
      Back space count records.

For the following commands, count is ignored:

  rewind
      Rewind the tape.

  offline, rewoffl
      Rewind the tape and place the tape unit off-line.

  status
      Print status information about the tape unit.

  retension
      Retension the tape.

  erase
      Erase the entire tape.

mt returns a 0 exit status when the operation(s) were successful, 1 if the command was unrecognized, and 2 if an operation failed.

FILES

/dev/rmt* Raw magnetic tape interface
/dev/rar* Raw Archive cartridge tape interface
/dev/rts* Raw SCSI tape interface

SEE ALSO

mtio(4), dd(1), ioctl(2), environ(5)

BUGS

Not all devices support all commands. For example, ar(4s) and et(4s) currently do not support the far, bsf, or bar commands; but they are the only ones that currently support the retension and rewind commands.
NAME

tip, cu — connect to a remote system

SYNOPSIS

```
tip  [-v] [-speed] system-name
tip  [-v] [-speed] phone-number
cu   phone-number [-t] [-a speed] [-a acu] [-l line] [-#]
```

DESCRIPTION

Tip and cu establish a full-duplex connection to another machine, giving the appearance of being
logged in directly on the remote computer. It goes without saying that you must have a account
on the machine (or equivalent) to which you wish to connect. The preferred interface is tip. The
cu interface is included for those people attached to the 'call UNIX' command of the version 7
UNIX system. This manual page describes only tip.

When tip starts up it reads commands from the file .tiprc in your home directory. If you use the
-v option on the tip command line, tip displays these commands as it executes them. See the
discussion on variables later on.

Typed characters are normally transmitted directly to the remote machine (which does the echoing
as well).

A tilde ('~') appearing as the first character of a line is an escape signal which directs tip to per-
form some special action. Tip recognizes the following escape sequences:

```
```

-D

Drop the connection and exit (you may still be logged in on the remote machine).

c [name]

Change directory to name (no argument implies change to your home directory).

-!

Escape to a shell (exiting the shell returns you to tip).

->

Copy file from local to remote.

<

Copy file from remote to local.

-p from | to |

Send a file to a remote UNIX host. When you use the put command, the remote UNIX sys-
tem runs the command string

```
cat > to
```

while tip sends it the from file. If the to file isn't specified, the from file name is used. This
command is actually a UNIX specific version of the '->' command.

-t from | to |

Take a file from a remote UNIX host. As in the put command the to file defaults to the
from file name if it isn't specified. The remote host executes the command string

```
cat > from; echo ~A
```
to send the file to tip.

-| Pipe the output from a remote command to a local UNIX process. The command string
sent to the local UNIX system is processed by the shell.

-C

Connect a program to the remote machine. The command string sent to the program is
processed by the shell. The program inherits file descriptors 0 as remote line input, 1 as
remote line output, and 2 as tty standard error.

-# Send a BREAK to the remote system. For systems which don't support the necessary ioctl
call the break is simulated by a sequence of line speed changes and DEL characters.

-s Set a variable (see the discussion below).

-z Stop tip (only available when run under the C-Shell).
"? Get a summary of the tilde escapes

Copying files requires some cooperation on the part of the remote host. When a "-> or "<- escape is used to send a file, tip prompts for a file name (to be transmitted or received) and a command to be sent to the remote system, in case the file is being transferred from the remote system. The default end of transmission string for transferring a file from the local system to the remote is specified as the 'oe' parameter in the remote(5) file, but may be changed by the set command. While tip is transferring a file the number of lines transferred will be continuously displayed on the screen. A file transfer may be aborted with an interrupt. An example of the dialogue used to transfer files is given below (input typed by the user is shown in bold face).

arpa% tip monet
[connected]
...(assume we are talking to another UNIX system)...
uchmonet login: sam
Password:
monet% cat > sylvester.c
"-> Filename: sylvester.c
32 lines transferred in 1 minute 3 seconds
monet%
monet% "< Filename: reply.c
List command for remote host: cat reply.c
65 lines transferred in 2 minutes
monet%
...(or, equivalently)...
monet% "p sylvester.c
...(actually echoes as '/put sylvester.c')...
32 lines transferred in 1 minute 3 seconds
monet%
monet% "t reply.c
...(actually echoes as '/fake reply.c')...
65 lines transferred in 2 minutes
monet%
...(to print a file locally)...
monet% "L Local command: pr -h sylvester.c | lpr
List command for remote host: cat sylvester.c
monet% "-D
[EOT]
...(back on the local system)...

The remote(5) file contains the definitions for remote systems known by tip; refer to the remote manual page for a full description. Each system has a default baud rate with which to establish a connection. If this value is not suitable, the baud rate to be used may be specified on the command line, for example:

tip -300 mds

When tip establishes a connection it sends out a connection message to the remote system. The default value for this string may be found in the remote file.

At any time that tip prompts for an argument (for example, during setup of a file transfer) the line typed may be edited with the standard erase and kill characters. A null line in response to a prompt, or an interrupt, aborts the dialogue and returns you to the remote machine.

When tip attempts to connect to a remote system, it opens the associated device with an exclusive-open ioctl(2) call. Thus only one user at a time may access a device. This is to prevent
multiple processes from sampling the terminal line. In addition, *tip* honors the locking protocol used by *uucp(1C)*.

AUTO-CALL UNITS

*Tip* may be used to dial up remote systems using a number of auto-call unit’s (ACU’s). When the remote system description contains the ‘du’ attribute, *tip* uses the call-unit (‘cu’), ACU type (‘at’), and phone numbers (‘pn’) supplied. Normally *tip* displays verbose messages as it dials. See *remote(5)* for details of the remote host specification.

Depending on the type of auto-dialer being used to establish a connection the remote host may have garbage characters sent to it upon connection. The user should never assume that the first characters typed to the foreign host are the first ones presented to it. The recommended practice is to immediately type a ‘kill’ character upon establishing a connection (most UNIX systems support ‘@’ as the initial kill character).

*Tip* currently supports the Ventel MD-212+ autodialer modem and the Hayes SmartModem 1200.

REMOTE HOST DESCRIPTIONS

Descriptions of remote hosts are normally located in the system-wide file *etc/remote*. However, a user may maintain personal description files (and phone numbers) by defining and exporting the REMOTE shell variable. The *remote* file must be readable by *tip*, but a secondary file describing phone numbers may be maintained readable only by the user. This secondary phone number file is *etc/phones*, unless the shell variable PHONES is defined and exported. As described in *remote(5)*, the *phones* file is read when the host description’s phone number(s) capability is an ‘@’. The phone number file contains lines of the form:

```
    system-name    phone-number
```

Each phone number found for a system is tried until either a connection is established, or an end of file is reached. Phone numbers are constructed from ‘0123456789-==*’ where the ‘-’ and ‘*’ are used to indicate a second dial tone should be waited for (ACU dependent).

TIP INTERNAL VARIABLES

*Tip* maintains a set of variables which are used in normal operation. Some of these variables are read-only to normal users (root is allowed to change anything of interest). Variables may be displayed and set through the ‘‘s’’ escape. The syntax for variables is patterned after *vi(1)* and *mail(1)*. Supplying ‘all’ as an argument to the set command displays all variables that the user can read. Alternatively, the user may request display of a particular variable by attaching a ‘!’ to the end. For example ‘escape?’ displays the current escape character.

Variables are numeric, string, character, or Boolean values. Boolean variables are set merely by specifying their name. They may be reset by prepending a ‘!’ to the name. Other variable types are set by appending an ‘=’ and the value. The entire assignment must not have any blanks in it. A single set command may be used to interrogate as well as set a number of variables.

Variables may be initialized at run time by placing set commands (without the ‘‘s’’ prefix) in a *.tiprc file in one’s home directory. The ‘-w’ option makes *tip* display the sets as they are made. Comments preceded by a ‘#’ sign can appear in the *.tiprc file.

Finally, the variable names must either be completely specified or an abbreviation may be given. The following list describes those variables known to *tip*, their abbreviations (surrounded by brackets), and their default values. Those variables initialized from the remote file are marked with a ‘*’. A mode is given for each variable — capitalization indicates the read or write capability is given only to the super-user.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Mode</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[be]autify</td>
<td>bool</td>
<td>rw</td>
<td>true</td>
<td>discard unprintables when scripting</td>
</tr>
<tr>
<td>Command</td>
<td>Type</td>
<td>Value</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
<td>-------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>baudrate</td>
<td>num</td>
<td>rw</td>
<td>connection baud rate</td>
<td></td>
</tr>
<tr>
<td>chardelay</td>
<td>num</td>
<td>rw</td>
<td>character delay for file transfers to remote (cl)</td>
<td></td>
</tr>
<tr>
<td>connect</td>
<td>num</td>
<td>rw</td>
<td>timeout (seconds) when establishing connection</td>
<td></td>
</tr>
<tr>
<td>discon</td>
<td>str</td>
<td>rw</td>
<td>string to send to disconnect (di)</td>
<td></td>
</tr>
<tr>
<td>echocheck</td>
<td>bool</td>
<td>rw</td>
<td>char's signifying EOT from the remote host</td>
<td></td>
</tr>
<tr>
<td>eolread</td>
<td>str</td>
<td>rw</td>
<td>string sent for EOT</td>
<td></td>
</tr>
<tr>
<td>eofwrite</td>
<td>str</td>
<td>rw</td>
<td>end of line indicators</td>
<td></td>
</tr>
<tr>
<td>col</td>
<td>str</td>
<td>rw</td>
<td>command prefix character</td>
<td></td>
</tr>
<tr>
<td>escape</td>
<td>char</td>
<td>rw</td>
<td>echo check timeout (et)</td>
<td></td>
</tr>
<tr>
<td>exceptions</td>
<td>str</td>
<td>rw</td>
<td>char's not discarded due to beautification</td>
<td></td>
</tr>
<tr>
<td>force</td>
<td>char</td>
<td>rw</td>
<td>force character</td>
<td></td>
</tr>
<tr>
<td>framesize</td>
<td>num</td>
<td>rw</td>
<td>size of buffering between writes on reception</td>
<td></td>
</tr>
<tr>
<td>halfduplex</td>
<td>bool</td>
<td>rw</td>
<td>falsehost is half duplex — do local echo (hd)</td>
<td></td>
</tr>
<tr>
<td>host</td>
<td>str</td>
<td>r</td>
<td>name of host connected to</td>
<td></td>
</tr>
<tr>
<td>linelimit</td>
<td>num</td>
<td>rw</td>
<td>line delay for transfers to remote (dl)</td>
<td></td>
</tr>
<tr>
<td>localschook</td>
<td>bool</td>
<td>rw</td>
<td>synonym for halfduplex</td>
<td></td>
</tr>
<tr>
<td>lock</td>
<td>str</td>
<td>RW</td>
<td>lock file for ACU logging</td>
<td></td>
</tr>
<tr>
<td>log</td>
<td>str</td>
<td>RW</td>
<td>ACU log file</td>
<td></td>
</tr>
<tr>
<td>parity</td>
<td>str</td>
<td>rw</td>
<td>parity to be generated (pa)</td>
<td></td>
</tr>
<tr>
<td>phones</td>
<td>str</td>
<td>r</td>
<td>file for hidden phone numbers</td>
<td></td>
</tr>
<tr>
<td>prompt</td>
<td>char</td>
<td>rw</td>
<td>end of line indicator set by host</td>
<td></td>
</tr>
<tr>
<td>raise</td>
<td>bool</td>
<td>rw</td>
<td>upper case mapping switch</td>
<td></td>
</tr>
<tr>
<td>raisectr</td>
<td>char</td>
<td>rw</td>
<td>interactive toggle for raise</td>
<td></td>
</tr>
<tr>
<td>rawftp</td>
<td>bool</td>
<td>rw</td>
<td>do not filter non-printable characters</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>do not do translations like \n to \r.</td>
<td></td>
</tr>
<tr>
<td>record</td>
<td>str</td>
<td>rw</td>
<td>name of script output file</td>
<td></td>
</tr>
<tr>
<td>remote</td>
<td>str</td>
<td>r</td>
<td>system description file</td>
<td></td>
</tr>
<tr>
<td>script</td>
<td>bool</td>
<td>rw</td>
<td>session scripting switch</td>
<td></td>
</tr>
<tr>
<td>tabexpand</td>
<td>bool</td>
<td>rw</td>
<td>expand tabs during file transfers</td>
<td></td>
</tr>
<tr>
<td>taendem</td>
<td>bool</td>
<td>rw</td>
<td>use XON/XOFF flow control (ta and nt)</td>
<td></td>
</tr>
<tr>
<td>verbose</td>
<td>bool</td>
<td>rw</td>
<td>make noise during file transfers</td>
<td></td>
</tr>
<tr>
<td>SHELL</td>
<td>str</td>
<td>rw</td>
<td>name of shell for &quot;!&quot; escape</td>
<td></td>
</tr>
<tr>
<td>HOME</td>
<td>str</td>
<td>rw</td>
<td>home directory for &quot;c&quot; escape</td>
<td></td>
</tr>
</tbody>
</table>

**ENVIRONMENT VARIABLES**

The following variables are read from the environment:
- **REMOTE**  The location of the `remote` file.
- **PHONES**  The location of the file containing private phone numbers.
- **HOST**    A default host to connect to.
- **HOME**    One's log-in directory (for chdirs).
- **SHELL**   The shell to fork on a "!" escape.

**FILES**

- `~/.tiprc` initialization file.
- `/usr/spool/uucp/LCK.*` lock file to avoid conflicts with `uucp`

**DIAGNOSTICS**

Diagnostics are, hopefully, self explanatory.
SEE ALSO
remote(5), phones(5)

BUGS
Note that chardelay and linedelay are currently not implemented.
NAME
malloc, free, realloc, calloc, cfree, alloca — memory allocator

SYNOPSIS
char *malloc(size)
unsigned size;

free(ptr)
char *ptr;

char *realloc(ptr, size)
char *ptr;
unsigned size;

char *calloc(nelem, elsize)
unsigned nelem, elsize;
cfree(ptr)
char *ptr;
char *alloca(size)
int size;

DESCRIPTION
Malloc, free, realloc, calloc, and cfree provide a simple, general-purpose, heap memory allocation package. Alloca provides a way to allocate additional stack space for the current procedure.

Malloc returns a pointer to a block of at least size bytes beginning on a word boundary.

The argument to free is a pointer to a block previously allocated by malloc, realloc, or calloc; this space is made available for further allocation, but its contents are left undisturbed.

 Needless to say, grave disorder will result if the space assigned by malloc, realloc, or calloc is overrun, or if some random number is handed to free.

Malloc allocates the first big enough contiguous reach of free space found in a circular search from the last block allocated or freed, coalescing adjacent free blocks as it searches. It calls sbrk (see brk(2)) to get more memory from the system when there is no suitable space already free.

Realloc changes the size of the block pointed to by ptr to size bytes and returns a pointer to the (possibly moved) block. The contents will be unchanged up to the lesser of the new and old sizes.

Realloc also works if ptr points to a block freed since the last call of malloc, realloc or calloc; however, this practice is highly implementation-dependent and is not recommended.

Calloc allocates space for an array of nelem elements of size elsize. The space is initialized to zeros, and can be freed with free or cfree.

Alloca allocates size bytes of space in the stack frame of the caller. This temporary space is automatically freed when the caller returns.

Each of the allocation routines returns a pointer to space suitably aligned (possibly after pointer coercion) for storage of any type of object.

DIAGNOSTICS
Malloc, realloc and calloc return a null pointer (0) if there is no available memory or if the heap has been detectably corrupted by storing outside the bounds of a block. Malloc may be recompiled to check the heap stringently on every transaction; those sites with a source code license may check the source code to see how this can be done.

BUCKS
When realloc returns 0, the block pointed to by ptr may be destroyed.
The current incarnation of the allocator is unsuitable for direct use in a large virtual memory environment where many small blocks are to be kept, since it keeps all allocated and freed blocks on a single circular list. Just before more memory is allocated, all allocated and freed blocks are referenced; this can cause a huge number of page faults.

`Alloc` is machine-dependent; its use is discouraged.
NAME
mem, kmem, mbmem, mbio, vme16, vme24 — main memory and bus I/O space

SYNOPSIS
None; included with standard system.

DESCRIPTION
These devices are special files that map memory and bus I/O space. They may be read, written, seek’ed and (except for kmem) mmap(2)’ed.

Mem is a special file that is an image of the physical memory of the computer. It may be used, for example, to examine (and even to patch) the system.

Kmem is a special file that is an image of the kernel virtual memory of the system.

Mbmem is a special file that is an image of the Multibus memory of the system. Multibus memory is in the range from 0 to 1 Megabyte.

Mbio is a special file that is an image of the Multibus I/O space. Multibus I/O space extends from 0 to 64K.

Vme16 is a special file that is an image of the VME 16-bit address space, extending from 0 to 64K.

Vme24 is a special file that is an image of the VME 24-bit address space, extending from 0 to 16 Megabytes. The VME 16-bit address space overlaps the top 64K of the 24-bit address space.

Mbmem and mbio can only be accessed in Multibus based systems; vme16 and vme24 can only be accessed in VME based systems.

When reading and writing mbmem and mbio odd counts or offsets cause byte accesses and even counts and offsets cause word accesses.

FILES
/dev/mem
/dev/kmem
/dev/mbmem
/dev/mbio
/dev/vme16
/dev/vme24
NAME
zs — zilog 8530 SCC serial communications driver

SYNOPSIS
device zs0 at mb0 csr 0xf6a000 flags 3 priority 6

DESCRIPTION
The Zilog 8530 provides 2 serial communication lines with full 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; see tty(4) for the encoding.

Bit i of flags may be specified to say that a line is not properly connected, and that the line i should be treated as hard-wired with carrier always present. Thus specifying "flags 0x2" in the specification of zs0 would cause line ttyb to be treated in this way.

To allow a single tty line to be connected to a modem and used for both incoming and outgoing calls, a special feature, controlled by the minor device number, has been added. Minor device numbers in the range 0 — 127 correspond directly to the normal tty lines and are named tty*. Minor device numbers in the range 128 — 256 correspond to the same physical lines as those above (i.e. the same line as the minor device number minus 128) and are (conventionally) named cua*. The cua lines are special in that they can be opened even when there is no carrier on the line. Once a cua line is opened, the corresponding tty line can not be opened until the cua line is closed. Also, if the tty line has been opened successfully (usually only when carrier is recognized on the modem) the corresponding cua line can not be opened. This allows a modem to be attached to /dev/ttya (usually renamed to /dev/ttyd0) and used for dialin (by enabling the line for login in /etc/ttye) and also used for dialout (by tip(1C) or uucp(1C)) as /dev/cua0 when no one is logged in on the line. Note that the bit in the flags word in the config file (see above) must be zero for this line.

FILES
/dev/tty[a,b,s0-s3]
/dev/ttyd[0-9,a-f]
/dev/cua[0-9,a-f]

SEE ALSO
tty(4)
Zilog Z8030/Z8530 SCC Serial Communications Controller (Sun 800-1052-01)

DIAGNOSTICS
zs%d%c: silo overflow. The character input silo overflowed before it could be serviced.
NAME
diag — stand-alone disk initialization and diagnosis

SYNOPSIS
b stand/diag

DESCRIPTION
Diag is a general-purpose stand-alone utility package for disk initialization and diagnosis. Diag supports the various SMD and ST-506 disk controllers.

Note: that diag can only be called from the Sun Workstation PROM monitor — diag is not a system utility.

The most common use of diag is formatting and labelling a disk — see the format, label, and partition commands.

Diag is interactive — it prompts for options and arguments. There are two phases to using diag: in the first phase, diag prompts for information about the type of disk drive and controller that it is working with, and essentially `configures' itself to work with that disk and controller. At the end of this phase, diag tries to access the disk controller you have defined. If the attempt succeeds, diag gives you a status report on the disk and gives you a 'diag>' prompt; this signals the beginning of diag's second phase. If the attempt to access the controller fails (if the controller is mis-defined or non-existent, for example), you get a bus error message, and return to the PROM monitor.

During diag's second phase, you can use the commands listed below in response to the 'diag>' prompt. Only enough characters to uniquely identify the command need be typed. The commands that diag currently recognizes are:

abortdma Aborts/does not abort the dmatest command when a data miscompare is detected. That is, if the internal variable set by abortdma is 'on' (default), the dmatest command aborts as soon as it finds an error; otherwise, the dmatest continues.

clear Sends a restore command to a disk. This is needed to manually reset disk faults.

diag Re-initializes the diag program itself — goes back to phase one of the initialization process described above.

dmatest Begins a continuous DMA test. The test copies random data to and from the designated controller, comparing data. If a miscompare is found, an error message is displayed and the test aborts (unless the abortdma command is used — see abortdma, above). Note: this command is available only with the Xylogics 450 Controller.

errors Reports/does not report all errors as they occur (toggles; default is reporting off).

fix Formats and verifies a range of tracks; any defective tracks/sectors found are automatically corrected using mapping or slipping. Note: this command is available for SMD controllers only.

format For SMD disks, formats the entire disk; for SCSI disks, initiates the SCSI format program.

help or ? Displays a list of the available commands.

info Reports/suppresses report of all disk activity as it completes (toggles; default is reporting off).

label Labels the disk.

map Displays current mappings and allows you to explicitly map one track/sector to a different track/sector. Usually used for manual bad sector mapping. The format and fix commands usually do this automatically when a bad track/sector is found. Note: the map command is disk controller-dependent: you can map tracks with an
Interphase controller, and sectors with Xylogics controllers. This command is not available for use with Adaptec disk controllers (ST-506 interface).

**mapcheck** Enables/disables checking for overlapping mapped sectors/tracks during the position, read, test, or write commands. If the internal variable set by mapcheck is 'on' when an error occurs (default), then the current mappings (if any) are read from the disk and checked to see if there are overlapping mappings over the area of the disk where the transfer failed.

**partition** Creates, assigns, or modifies logical partition tables for a disk. The UNIX operating system requires logical partitions. The label command writes the partition map to the disk. There are standard partition tables for each type of disk that diag knows about.

**position** Continuously tests the disk by reading random sectors from the disk. To abort the test, type a "C (CONTROL-C).

**quit** Quits from diag and returns to the PROM monitor.

**rhdr** Reads and displays the track headers for a specified track. Note: this command is available for the Xylogics 450 controller only.

**read** Reads specified blocks from the disk. The read command prompts for the starting block number, number of blocks, and the block increment. The read command doesn't report the data it reads — it is intended for verifying that blocks are readable.

**scan** Continuously scans over a range of sectors looking for defective sectors by writing/reading/verifying various bit patterns to sequential sectors. Any data on the disk in the range to be scanned is destroyed. Sectors previously mapped are not scanned, so any errors reported will be newly found defective sectors. If used with a Xylogics controller, defective sectors can be automatically mapped/slipped when they are found. To abort the scan, type a "C (CONTROL-C).

**seek** Performs a seek test on the disk: a seek is made to every cylinder and to every possible cylinder distance.

**slip** Explicitly slips one sector on a track. Usually used for manual bad sector slipping. The format and fix commands usually do this automatically when bad sectors are found and the proper conditions exist. Note: slipping is available only with the Xylogics 450 controller, and only when the disk has a spare data sector per track.

**slipmsgs** Displays/suppresses display of track headers before and after each slip operation that occurs during the fix, format, and slip commands (toggles; default is display off).

**status** Reports the ready status of each drive on the current controller.

**test** Continuously tests the disk by writing random data to random sectors on the disk and then verifying that the correct data can be read back. The test command destroys data on the disk. To abort the test, type a "C (CONTROL-C).

**time** Turns timing on/off. When timing is on, diag reports on how long certain operations take — diag is less verbose in this state so it doesn't waste time displaying messages (toggles; default is timing off).

**translate** Translates a given block number into its decimal value, hexadecimal value, and logical disk address.

**verify** Reads and displays the label from the disk. Shows the logical partition assignments. This done automatically when the label command has labelled the disk.

**version** Displays the SCCS identification strings for this version of diag.
whdr  Modifies and writes the track headers for a specified track. **Note:** this command is available with the Xylogics 450 controller only. Also, it should be used only for specialized patching — misuse may destroy track data.

write  Verifies that blocks are writable by writing garbage data to specified blocks on the disk. The *write* command prompts for the starting block number, number of blocks, and the block increment.

+      Adds two block numbers and reports the result in decimal, hexadecimal, and as a logical disk address.

−      Subtracts two block numbers and reports the result in decimal, hexadecimal, and as a logical disk address.

Block numbers may be entered either as an absolute decimal block number, or as a disk address of the form cylinder/head/sector.

Any *diag* command may be aborted by typing **a** `CTRL-C` (CONTROL-C).
NAME
acct – execution accounting file

SYNOPSIS
#include <sys/acct.h>

DESCRIPTION
The acct(2) system call makes entries in an accounting file for each process that terminates. The
accounting file is a sequence of entries whose layout, as defined by the include file is:

/* @(#)acct.h 1.1 84/12/20 SMI; from UCB 6.1 83/07/29 */

/*
* Accounting structures;
* these use a comp_t type which is a 3 bits base 8
* exponent, 13 bit fraction “floating point” number.
*/
typedef u_short comp_t;

struct acct
{
    char ac_comm[10]; /* Accounting command name */
    comp_t ac_utime;   /* Accounting user time */
    comp_t ac_stime;   /* Accounting system time */
    comp_t ac_etime;   /* Accounting elapsed time */
    time_t ac_btime;   /* Beginning time */
    short ac_uid;      /* Accounting user ID */
    short ac_gid;      /* Accounting group ID */
    short ac_mem;      /* average memory usage */
    comp_t ac_io;      /* number of disk IO blocks */
    dev_t ac_tty;      /* control typewriter */
    char ac_flag;      /* Accounting flag */
};

#define AFORK 0001 /* has executed fork, but no exec */
#define ASU 0002 /* used super-user privileges */
#define ACOMPAT 0004 /* used compatibility mode */
#define ACORE 0010 /* dumped core */
#define AXSIG 0020 /* killed by a signal */

#ifndef KERNEL
#ifndef SYSACCT
struct acct acctbuf;
struct vnode *acctp;
#else
#define acct()
#endif
#endif

If the process does an execve(2), the first 10 characters of the filename appear in ac_comm. The
accounting flag contains bits indicating whether execve(2) was ever accomplished, and whether
the process ever had super-user privileges.

SEE ALSO
acct(2), execve(2), sa(8)
NAME
utm, wtmp — login records

SYNOPSIS
#include <utmp.h>

DESCRIPTION
The utmp file records information about who is currently using the system. The file is a sequence
of entries with the following structure declared in the include file:

/*
#include h 1.1 84/12/20 SMI; from UCB 4.2 83/05/22 */

/*
* Structure of utmp and wtmp files.
* * Assuming the number 8 is unwise.
*/
struct utmp {
    char ut_line[8];       /* tty name */
    char ut_name[8];       /* user id */
    char ut_host[16];      /* host name, if remote */
    long ut_time;          /* time on */
};

This structure gives the name of the special file associated with the user's terminal, the user's
login name, and the time of the login in the form of time(3C).

The wtmp file records all logins and logouts. A null user name indicates a logout on the associ-
ated terminal. Furthermore, the terminal name "" indicates that the system was rebooted at the
indicated time; the adjacent pair of entries with terminal names "'" and '"' indicate the system-
maintained time just before and just after a date command has changed the system's idea of the
time.

Wtmp is maintained by login(1) and init(8). Neither of these programs creates the file, so if it is
removed record-keeping is turned off. It is summarized by ac(8).

FILES
/etc/utmp
/usr/adm/wtmp

SEE ALSO
login(1), init(8), who(1), ac(8)
NAME

bessel functions — of two kinds for integer orders

SYNOPSIS

function besj0 (x)

function besj1 (x)

function besjn (n, x)
integer*4 n

function besy0 (x)

function besy1 (x)

function besyn (n, x)
integer*4 n

double precision function dbesj0 (x)
double precision x

double precision function dbesj1 (x)
double precision x

double precision function dbesjn (n, x)
integer*4 n
double precision x

double precision function dbesy0 (x)
double precision x

double precision function dbesy1 (x)
double precision x

double precision function dbesyn (n, x)
integer*4 n
double precision x

DESCRIPTION

These functions calculate Bessel functions of the first and second kinds for real arguments and integer orders.

DIAGNOSTICS

Negative arguments cause besy0, besy1, and besyn to return a huge negative value. The system error code will be set to EDOM (33).

FILES

/usr/lib/libF77.a

SEE ALSO

j0(3m), perror(3F)
NAME

bit — and, or, xor, not, rshift, lshift, bic, bis, bit, setbit functions

SYNOPSIS

(generic) function and (word1, word2)

(generic) function or (word1, word2)

(generic) function xor (word1, word2)

(generic) function not (word)

(generic) function rshift (word, nbits)

(generic) function lshift (word, nbit)

subroutine bic (bitnum, word)
Integer*4 bitnum, word

subroutine bis (bitnum, word)
Integer*4 bitnum, word

subroutine setbit (bitnum, word, state)
Integer*4 bitnum, word, state

logical function bit (bitnum, word)
Integer*4 bitnum, word

DESCRIPTION

The and, or, xor, not, rshift, and lshift functions are generic functions expanded inline by the
compiler. Their arguments must be integer or logical values (short or long). The returned
value has the data type of the first argument.

and computes the bitwise ‘and’ of its arguments.
or computes the bitwise ‘or’ of its arguments.
xor computes the bitwise ‘exclusive or’ of its arguments.
not returns the bitwise complement of its argument.
lshift is a logical left shift with no end around carry.
rshift is an arithmetic right shift with sign extension. No test is made for a reasonable value of
nbit.

Bic, bis, and setbit are external subroutines which operate on integer*4 arguments.
bls sets bitnum in word.
bic clears bitnum in word.
setbit sets bitnum in word to 1 if state is nonzero and clears it otherwise.
bit is an external function which tests bitnum in word and returns .true. if bitnum is a 1
(one), and returns .false. if bitnum is a 0 (zero).

FILES

/usr/lib/libF77.a
NAME
time, ctime, ltime, gmtime — return system time

SYNOPSIS

integer function time()

character*24 function ctime (stime)
integer*4 stime

subroutine ltime (stime, tarray)
integer*4 stime, tarray(9)

subroutine gmtime (stime, tarray)
integer*4 stime, tarray(9)

DESCRIPTION

Time returns the time since 00:00:00 GMT, Jan. 1, 1970, measured in seconds. This is the value of the UNIX system clock.

Ct ime converts a system time to a 24 character ASCII string. The format is described under ctime(3). No 'newline' or NULL will be included.

Ltime and gmtime dissect a UNIX time into month, day, etc., either for the local time zone or as GMT. The order and meaning of the 9 elements returned in tarray is described under ctime(3).

FILES
/usr/lib/libU77.a

SEE ALSO
c time(3), idate(3F), fdate(3F)
NAME
rand, drand, irand — return random values

SYNOPSIS

function irand (iflag)

function rand (iflag)

double precision function drand (iflag)

DESCRIPTION
These functions use random(3) to generate sequences of random numbers. If iflag is '1', the generator is restarted and the first random value is returned. If iflag is otherwise non-zero, it is used as a new seed for the random number generator, and the first new random value is returned. The three functions share the same 256 byte state array.

irand returns positive integers in the range 0 through 2147483647. Rand and drand return values in the range 0.0 through 1.0.

FILES
/usr/lib/libF77.a

SEE ALSO
random(3)
NAME
etime, dtime — return elapsed execution time

SYNOPSIS
real function etime (tarray)
real tarray(2)

real function dtime (tarray)
real tarray(2)

DESCRIPTION
These two routines return elapsed runtime in seconds for the calling process. Dtime returns the elapsed time since the last call to dtime, or the start of execution on the first call.

The argument array returns user time in the first element and system time in the second element. Elapsed time, the returned value, is the sum of user and system time.

The resolution is determined by the system clock frequency.

FILES
/usr/lib/libU77.a

SEE ALSO
getrusage(2)
NAME
fdate — return date and time in an ASCII string

SYNOPSIS

subroutine fdate (string)
character*24 string

character*24 function fdate()

DESCRIPTION
Fdate returns the current date and time as a 24 character string in the format described under ctime(3). Neither 'newline' nor NULL will be included.

Fdate can be called either as a function or as a subroutine. If called as a function, the calling routine must define its type and length. For example:

character*24 fdate
write(*,*) fdate()

FILES
/usr/lib/libu77.a

SEE ALSO
cftime(3), time(3F), idate(3F)
NAME

stat, lstat, fstat — get file status

SYNOPSIS

Integer function stat (name, statb)
character(*) name
Integer statb(12)

Integer function lstat (name, statb)
character(*) name
Integer statb(12)

Integer function fstat (lunit, statb)
Integer statb(12)

DESCRIPTION

These routines return detailed information about a file. Stat and lstat return information about file name; fstat returns information about the file associated with fortran logical unit lunit. The meaning of the information returned in array statb is as described for the structure stat under stat(2). 'Spare' values are not included, the order is shown below.

The value of either function will be zero if successful; an error code otherwise.

statb(1) device inode resides on
statb(2) this inode's number
statb(3) protection
statb(4) number of hard links to the file
statb(5) user-id of owner
statb(6) group-id of owner
statb(7) the device type, for inode that is device
statb(8) total size of file
statb(9) file last access time
statb(10) file last modify time
statb(11) file last status change time
statb(12) optimal blocksize for file system i/o ops
statb(13) actual number of blocks allocated

FILES

/usr/lib/libU77.a

SEE ALSO

stat(2), access(3F), perror(3F), time(3F)

BUGS

Pathnames can be no longer than MAXPATHLEN as defined in <sys/param.h>.
NAME
long, short — integer object conversion

SYNOPSIS
integer*4 function long (int2)
integer*2 int2

integer*2 function short (int4)
integer*4 int4

DESCRIPTION
These functions provide conversion between short and long integer objects. Long is useful when
constants are used in calls to library routines and the code is to be compiled with ‘-i2’. Short is
useful in similar context when an otherwise long object must be passed as a short integer.

FILES
/usr/lib/libF77.a
Appendix A

Distribution Tape Reorganization

If you are installing a complete system distribution from 1/4” tape, your 1.1 System Manager’s Manual is incorrect about the number of cartridges you should have: the distribution now takes three, rather than two, quarter-inch tape cartridges. The 1.1 manual is still correct for 1/2” tape distributions.

The contents of the tape cartridges is:

Table A-1: Contents of Quarter-inch Distribution Tape Cartridges

<table>
<thead>
<tr>
<th>Tape</th>
<th>File Number</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tape 1</td>
<td>1</td>
<td>A general purpose boot program which knows how to boot from the various devices that can be attached to the Sun Workstation. The PROM monitor boots this general purpose boot program.</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>A copy of the new version of the <em>diag</em> program. <em>diag</em> is used during installation to format and label disks. Remember to use the procedures in this 1.4 Manual if you use <em>diag</em> to initialize a new disk.</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>A stand alone <em>copy</em> program which can copy from specified sources to specified destinations.</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>An image of a miniature version of the root file system, which contains enough information to get something up and running.</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>Copyright file.</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>The complete root file system for the UNIX operating system. This is on the tape in <em>tar</em>(1) format and is loaded in by the <em>xtr</em> or <em>rztr</em> utility.</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>Dummy placeholder. On the 1.1 tapes, this file contained the online manual sources, demos, and games. These have been moved to the third tape cartridge (see below). We put a small file here to maintain the original order of the tape files.</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>Copyright file.</td>
</tr>
</tbody>
</table>

Release 1.4 of February 1985
<table>
<thead>
<tr>
<th>Tape</th>
<th>File Number</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tape 2</td>
<td>1</td>
<td>Copyright file.</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>A <code>tar</code> image of the <code>/usr</code> file system. The <code>setup</code> program transfers this file system to the disk.</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Copyright file.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tape</th>
<th>File Number</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tape 3</td>
<td>1</td>
<td>Copyright file.</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Optional software: online manual sources (<code>/usr/man</code>), games (<code>/usr/games</code>), demonstration executables and source (<code>/usr/demo</code>), and font files for the Versatec printer (<code>/usr/lib/ufont</code>). You may load any/all of these to your disk(s) if you wish (space permitting); procedures are given below.</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Copyright file.</td>
</tr>
</tbody>
</table>

The only part of installation this reorganization affects is the final, optional, step in the 1.1 System Manager’s Manual: *Loading the Manuals, Demos, and Games Directories* (Section 3.3.13 of the 120/170 Manual; Section 4.3.13 of the 100U/150U Manual). If you wish to load the optional software, use the directory sizes cited and procedures given below rather than those from the 1.1 Manual.

### A.1. Loading Optional Software

If you wish to load the manual pages, demos, games, or `ufont` files from quarter-inch tape, proceed as follows:

1. Check your available disk space with `df(1)` to make sure you have adequate storage space for the software you wish to extract. For example:

   ```
   anysys% df
   Filesystem     kbytes   used avail capacity Mounted on
   /dev/xy0a      7437     6254   439    93%    /
   /dev/xy0g     10027    2846   6178   32%    /pub
   /dev/xy0d     28351    22786  2729    89%    /usr
   /dev/xy0e     122119   47130  62777   43%    /usr2
   [and so on]
   ``)

   You can use the following table to estimate how much space is required to store the software on disk. Calculate ± 5%.
Table A-2: Optional Software Storage Requirements

<table>
<thead>
<tr>
<th>Software</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Games</td>
<td>1.80 MBytes</td>
</tr>
<tr>
<td>Manual pages</td>
<td>2.20 MBytes</td>
</tr>
<tr>
<td>lib/vfont files</td>
<td>5.60 MBytes</td>
</tr>
<tr>
<td>Demos</td>
<td>4.10 MBytes</td>
</tr>
</tbody>
</table>

2. Load the third cartridge of the distribution tape (you can use a local or remote tape drive), and use `tar` with the appropriate keywords (man, demo, games, lib/vfont) to extract the software you want. Used without keywords, `tar` loads everything in the tape file.

For a machine with a local tape drive, use the following command series. Remember to substitute the correct device abbreviation (st or ar) for `tape`:

```sh
gaia# cd /usr
gaia# mt -f /dev/nrtape0 rew
gaia# mt -f /dev/nrtape0 fsf 1
gaia# tar xvfb /dev/nrtape0 200 { keyword ... }
```

For a machine using a remote tape drive, type the following. Remember to replace `tape` with the appropriate device abbreviation for the remote tape drive you are using, and to replace `remote_host` with the hostname of the machine this tape drive is attached to:

```sh
gaia# cd /usr
gaia# rsh remote_host mt -f /dev/nrtape0 rew
gaia# rsh remote_host mt -f /dev/nrtape0 fsf 1
gaia# rsh remote_host dd if=/dev/nrtape0 bs=200b l tar xvBf - {keyword ... }
```

You can ignore the “Broken pipe” message delivered after the last command.