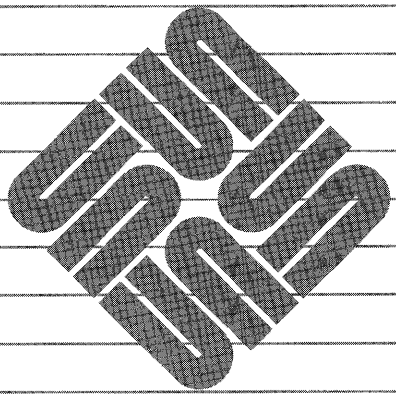
sun microsystems

Release 3.2 Manual *for the* Sun Workstation®



Sun Microsystems, Inc. • 2550 Garcia Avenue • Mountain View, CA 94043 • 415-960-1300

Part Number: 800-1364-10
Revision: A of 19 September 1986

Credits and trademarks

Sun Workstation, **Sun-1**, and **Sun-2**, **Sun-3** are trademarks of Sun Microsystems, Incorporated.

Multibus is a trademark of Intel Corporation.

UNIX is a trademark of AT&T Bell Laboratories.

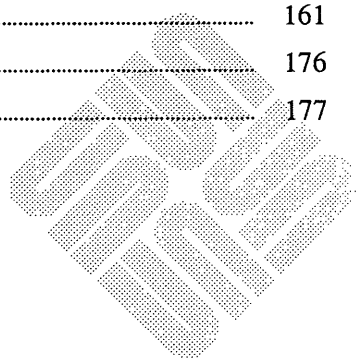
VMEbus is a trademark of VMEbus Manufacturers Group.

Copyright © 1982, 1983, 1984, 1985, 1986 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.

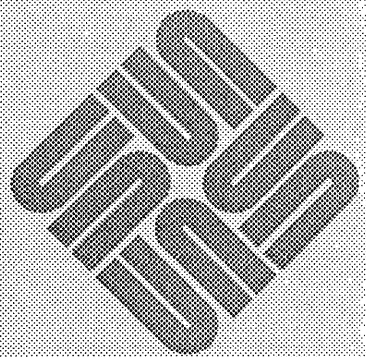
Contents

Tables



1

Introduction



1

Introduction

Sun's Release 3.2 has many new features:

- Software support for the new Sun-3/110, Sun-3/120 , Sun-3/260, and Sun-3/280 systems.
- Introduction of System V compatibility package
- Enhancements for 3.0 systems
- Bug fixes for Release 3.0 and 3.1
- This release is completely compatible with 3.0 and 3.1. Any program that has been developed to run under 3.0 and 3.1 runs on 3.2. However, you have to recompile to take advantage of the new features.

NOTE *Some of the software that was standard in previous releases is now optional.*

- More of the software in Release 3.2 can be optionally loaded through *Setup* This will allow a user to load only the software needed, thus freeing up more space on local disks. See *Appendix C* for a complete list of optional software.

1.1. Supporting Documentation

- *Installing Unix on the Sun Workstation* for Release 3.2 (800-1521)
- *System V Enhancements Overview* (800-1541)
- *Writing Device Driver Manual* (800-1304)
- *Commands Reference Manual for the Sun Workstation* (800-1295)
- *Unix Interface Reference Manual* (800-1303)
- *SunView Programmer's Guide* (800-1345)
- *SunView System Programmer's Guide* (800-1342)
- *Sun System Diagnostics Manual* (800-1529)
- *Floating Point Programmer's Guide for the Sun Workstation* (800-1552)

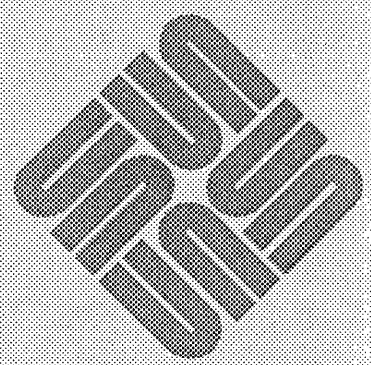
1.2. Documentation conventions

The following conventions are used in the procedures and examples throughout this document:

- 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.

2

3.2 Release Upgrade Installation



2

3.2 Release Upgrade Installation

2.1. Introduction

Purpose

In this chapter, we guide you through the 3.2 upgrade. Because there were some small shipments of earlier software released under 3.2*pilot*, 3.2*beta*, and 3.2+*extentions*, it is necessary to make the following distinctions for those users who might be running these earlier releases:

3.0*fcs*= 3.0
3.2*fcs*= 3.2
3.2*pilot*= 3.2 pilot version
3.2*beta*= 3.2 beta version
3.2+*extensions*= 3.2pilot + Sun-3/260 patch tape software

Any users running the software shipped with this release will be running the final version of the 3.2 release software (3.2*fcs*).

The upgrade procedures are designed to upgrade systems currently running 3.0*fcs*, 3.2*pilot* or 3.2+*extensions* releases only.

NOTE *If you are currently running release 3.1, you must install Release 3.2 using* Setup. *See* Installing Unix on the Sun Workstation *(800-1521) for complete details.*

This upgrade procedure supports the following configurations:

- Standalone machines with local tape drive and disk(s).
- Standalone machines with disk(s) but without local tape drive.
- Homogeneous servers with local tape drive and disk(s).
- Homogeneous servers with disk(s) but without local tape drive.
- Heterogeneous servers with local tape drive and disk(s).
- Heterogeneous servers with disk(s) but without local tape drive.

Please read through this chapter a few times and familiarize with the procedures before you start the upgrade.

Terminology

In this section, we define some of the specific terms that are used in this chapter. We assume that you have some experience in installing UNIX on a Sun Workstation. If you do not understand some of the instructions or terms in this chapter, refer to *Installing UNIX on the Sun Workstation* and the *System Administration Manual for the Sun Workstation* for details.

Tape host: The system with the tape drive is called the *tape host.*

Homogeneous Server: A server that supports clients of its own architecture.

Heterogeneous Server: A server that supports both MC68010 and MC68020 clients.

Requirements For Each Configuration

In this section, we list the requirements of each configuration that the upgrade procedures support. You MUST meet all the requirements of your configuration in order to upgrade your system. Upgrading with a remote tape drive is slower than upgrading with a local tape drive. Unless these is no other choice, upgrade with a local tape drive is highly recommended. Client partitions that are commented out in /etc/nd.local on a server will not be upgraded. Refer to section **2.4** of this chapter for instructions on how to upgrade these partitions. A client that boots a special kernel from its root partition will be upgraded with the 3.2*fcs* GENERIC kernel in its root partition after the upgrade. If you want that client to boot the kernel that it ran before the upgrade, there are three steps you need to follow:

- Make sure there are no compatibility problems.
- Copy the kernel to your home directory before the upgrade.
- Move the kernel back to the client root partition after the upgrade and before you bring up the client system.

If you wish the client to boot a special 3.2 kernel, configure a new kernel after the upgrade and move it to the client root partition before you bring up the client system.

The following upgrade walkthroughs are provided in this chapter:

1. Standalone System With Local Tape Drive

- Standalone system running 3.0, 3.2*pilot* or 3.2+*extensions* with local tape drive

2. Standalone System With Remote Tape Drive

- Standalone system running 3.0, 3.2*pilot*, or 3.+*extensions*.
- The tape host must be reachable from the standalone system through the ethernet. The tape host and the standalone system must be on the same network.
- `/etc/hosts` of a standalone system must contain the internet address of the tape host whether or not yp is running.
- `/etc/hosts` of the tape host must contain the internet address of the standalone system if yp is not running. If yp is running, the `/etc/hosts` file on the master yp server must contain the internet address of the standalone

system.

- `/.rhosts` of the tape host must contain the name of the standalone system
- The standalone system must be able to boot the FCS version of miniroot from a server on the same network with same architecture as the standalone system. This server can be the tape host or another server on the same network.

  The FCS version of miniroot requires 5 M of disk space from the server's */pub*. Put the miniroot under the server's /pub. Create a symbolic link under */tftpboot* if the standalone system is a Sun-3 machine. Boot the miniroot from server. See *Appendix B* of *Installing Unix on the Sun Workstation.*

3. Homogeneous Server With Local Tape Drive

- Homogeneous server running 3.0, 3.2*pilot*, or 3.2+*extensions* with local tape drive

4. Homogeneous Server With Remote Tape Drive

- Homogeneous server running 3.0, 3.2*pilot*, or 3.2+*extensions*
- The tape host must be reachable from the homogeneous server through the ethernet. The tape host and the homogeneous server must be on the same network.
- `/etc/hosts` of homogeneous server must contain the internet address of the tape host whether or not yp is running.
- `/etc/hosts` of the tape host must contain the internet address of the homogeneous server if yp is not running. If yp is running, the `/etc/hosts` file on the master yp server must contain the internet address of the standalone system.
- `/.rhosts` of the tape host must contain the name of the homogeneous server
- Homogeneous server must be able to boot the FCS version of the miniroot from a server on the same network with same architecture as the homogeneous server. This server can be the tape host or another server on the same network.

  The FCS version of miniroot requires 5 M of disk space from server's `/pub`. Put the miniroot under server's */pub*. Create a symbolic link under */tftpboot* if the standalone system is a Sun-3 machine. Boot the miniroot from the server. See *Appendix B* of *Installing Unix on the Sun Workstation* for more details.

5. Heterogeneous Server With Local Tape Drive

- Heterogeneous server running 3.0, 3.2*pilot*, or 3.2+*extensions* with local tape drive
- Make sure each client mounts a user file system of its own architecture from the server. A MC68010 client uses the server's `/usr.MC68010` file. A

MC68020 client uses the server's `/usr.MC68020` file. Make sure `/etc/fstab` file for each client partition contains the correct information.

6. Heterogeneous Server With Remote Tape Drive

- Heterogeneous server running 3.0, 3.2*pilot*, or 3.2+*extensions*
- The tape host must be reachable from the heterogeneous server through the ethernet. The tape host and the heterogeneous server must be on the same network.
- `/etc/hosts` of the heterogeneous server must contain the internet address of the tape host whether or not yp is running.
- `/etc/hosts` of tape host must contain the internet address of the heterogeneous server if yp is not running. If yp is running, the `/etc/hosts` file on the master yp server must contain the internet address of the standalone system.
- `/.rhosts` of tape host must contain the name of the heterogeneous server.
- Heterogeneous server must be able to boot the FCS version of the miniroot from a server on the same network with same architecture as the heterogeneous server. This server can be the tape host or another server on the same network.

  The FCS version of the miniroot requires 5 M of disk space from the server's /pub. Put the miniroot under the server's */pub*. Create a symbolic link under */tftpboot* if the standalone system is a Sun-3 machine. Boot the miniroot from server. See *Appendix B* of *Installing Unix on the Sun Workstation* for more details
- Make sure each client mounts a user file system of its own architecture from the server. A MC68010 client uses the server's `/usr.MC68010` file. A MC68020 client uses the server's `/usr.MC68020` file. Make sure `/etc/fstab` file for each client partition contains the correct information.

Table 2-1 *System Sizes without Optional Software*

System	*Size*
Standalone System	
(/)	5707K
(/usr)	13651K
Homogeneous Server	
(/)	1695K
(/pub)	4012K
(/usr)	13651K
Heterogeneous Server	
(/)	1695K
(/pub.MC68010)	4012K
(/pub.MC68020)	4012K
(/usr.MC68010)	13651K
(/usr.MC68020)	13651K

NOTE *Each client root partition requires at least 1661 Kbytes.*

2.2. Upgrade Overview

1. Login to the system and use the *df*(1) command to display information about the space available in each file system. All optional software currently existing on the disk(s) will automatically be loaded with the latest version. The upgrade procedure will ask if you wish to load any optional software which does not exist on the disk(s). If you wish to load additional software, you MUST make sure the file systems have enough space available BEFORE you start the upgrade. Use the table below to figure out the space requirements on your disks. If there is not enough space to load additional software, it is recommended that you run *Setup* to reinstall your system with adequate disk space. See *Installing Unix on the Sun Workstation*(800-1521)

Table 2-2 *Optional Software Space Requirements*

Optional Software	*Size*
Networking	2320K (/usr)
Debugging	1041K (/usr)
Suntools_users	1940K (/usr)
Suntools_programmers	2269K (/usr)
Suntools_source	401K (/usr)
Text_processing	801K (/usr)
Setup	957K (/usr)
Stand_diag	4K (/pub) 4K (/usr)
Fortran	837K (/usr)
Usr_diag	1538K (/usr)
Graphics	2860K (/usr)
Pascal	998K (/usr)
Profiled	879K (/usr)
Uucp	5K(/) 536K (/usr)
System V	3518K (/usr)
Man	4510K (/usr)
Demo	2270K (/usr)
Games	2494K (/usr)
Vtroff	6028K (/usr)

2. Full backups are strongly recommended before you start the upgrade. Remember to halt all clients before the backups if the system is a server. Users' home directories on the disk will not be touched by the upgrade procedure. Therefore, you can copy the files you want to save on the disk to your home directory and restore the files after the upgrade. `/usr/lib/sendmail.cf` will not be touched. The copy of `/usr/lib/sendmail.cf` that currently exists on the disk will be saved and be used after the upgrade. Earlier versions of the mail configuration files are compatible with 3.2*fcs*. If you wish to use the latest version of the mail configuration file, refer to the Communications Chapter of the *System Administration Manual* for detailed instructions.

3. Become superuser and halt your system. You need to make sure all clients are halted before you halt the system if the system is a server.

   ```
   host# /etc/halt
   ```

4. If you are upgrading the system with a remote tape drive, skip this step and go to the next step (step 5). If you are upgrading the system with a local tape drive, boot the general purpose bootstrap program from the tape by typing

'b' followed by two character device abbreviation for your tape drive type, and open and closed parentheses.

```
>b tape()
```

Table 2-3 *Tape Devices*

Devices	*Description*
ar	Archive quarter-inch tape cartridge
mt	Nine-track magnetic 1/2" tape-Tapemaster controller
st	SCSI tape controller cartridge
xt	Nine-track magnetic 1/2" tape-Xylogics 472 controller

5. Load the minimal subset of UNIX call "miniroot" onto your disk. All upgrade software resides on this miniroot. If you are upgrading with a local tape drive, do the following:

```
Boot: tape(0,0,4)
Standalone Copy
From: tape(0,0,5)
To: disk(0,0,1)
```

Table 2-4 *Disk Devices*

Devices	*Description*
xy	Xylogics 440/450/451 SMD disk controller
sd	SCSI disk controller
ip	Interphase disk controller (Sun-2 only)

If you are upgrading with a remote tape drive, do the following:

```
>b ethernet(0,serverhost)stand/copy
From: ethernet(0,serverhost,pub#)miniroot
To: disk(0,0,1)
```

Table 2-5 *Ethernet Types*

Type	*Description*
ec	3COM ethernet controller
ie	Sun-2 ethernet controller
le	Sun-3 ethernet controller

serverhost is the host number of the server, which has the bootable FCS version of miniroot, in hexadecimal representation. Refer to the section *Requirements for*

Each Configuration in the beginning of this chapter.

The `pub#` is 0 if the system you are upgrading is Sun-2 and 1 if the system you are upgrading is Sun-3.

6. Boot the miniroot from your disk.

```
Boot: disk(0,0,1)vmunix -as
      .
      .
      .
root device ? disk0*
      .
      .
      .
```

7. When the system is up and displays a # prompt, make sure the date is correct. Now change your working directory to `/usr/etc/upgrade`.

```
# cd /usr/etc/upgrade
```

Start the upgrade procedure by typing the following command:

```
# UPGRADE
```

8. Specify the type of the system.

```
Enter system type ? [standalone | server]:
```

If the system is a server, you need to specify whether it is a homogeneous server or a heterogeneous server.

```
Enter server type ? [homo | heter]:
```

9. Specify whether the upgrade will be done with a local tape drive or remote tape drive. If you are upgrading the system with a remote tape drive, make sure you meet all the requirements specified in the last section.

```
Enter tape drive type ? [local | remote]:
```

If you are upgrading the system with a remote tape drive, you also need to specify the name of the tape host and the ethernet type of your system.

```
Enter host of remote drive ?

Enter ethernet type of this system ? [ec | ie | le]
:
```

See *Table 2-5.*

10. Specify abbreviation of tape device.

```
Enter tape type ? [ar | st | mt | xt]:
```

See *Table 2-3.*

11. Specify whether the system is running yellow pages or not.

```
Enter yp type of machine? [master | slave | client
| none]:
```

12. Specify the disk partition where the root file system resides.

```
Enter root disk partition for the MC680x0 architec-
ture (e.g. xy0a)?
```

13. Specify the list of optional software to be loaded. If the system is a heterogeneous server, you need to specify the list of optional software to be loaded for both MC68010 and MC68020 architectures. Optional software currently existing on the disk will be automatically loaded. Optional software currently not existing on the disk will be prompted for your attention.

```
Select optional software for the MC680x0 architecture :

Do you want to install "Networking"? [y/n]:
Do you want to install "Debugging"? [y/n]:
Do you want to install "Suntools_users"? [y/n]:
Do you want to install "Suntools_programmers"? [y/n]:
Do you want to install "Suntools_source"? [y/n]:
Do you want to install "Text_processing"? [y/n]:
Do you want to install "Setup"? [y/n]:
Do you want to install "Stand_diag"? [y/n]:
Do you want to install "Fortran"? [y/n]:
Do you want to install "Usr_diag"? [y/n]:
Do you want to install "Graphics"? [y/n]:
Do you want to install "Pascal"? [y/n]:
Do you want to install "Profiled"? [y/n]:
Do you want to install "Uucp"? [y/n]:
Do you want to install "System_V"? [y/n]:
Do you want to install "Man"? [y/n]:
Do you want to install "Demo"? [y/n]:
Do you want to install "Games"? [y/n]:
Do you want to install "Vtroff"? [y/n]:
```

14. UPGRADE saves some of the administrative files from the release the system is currently running for upgrade purpose. If you answer 'y', UPGRADE will remove these old files after UPGRADE is completed. If you answer 'n', UPGRADE will leave these old files on the disk after UPGRADE is completed. Since you won't be needing these files after the upgrade and they take some disk space, it is recommended that you answer 'y' to remove these files after the upgrade.

    Do you want to remove files saved from 3.0FCS after the upgrade ? [y/n]:
    or
    Do you want to remove files saved from 3.2PILOT after the upgrade? [y/n]:

15. The upgrade procedure is about to begin. If you entered information incorrectly, you can answer 'n' and restart by going back to step 7. If you are ready to start the upgrade, answer 'y' and you will be prompted for attention only when tape needs to be changed.

    ```
    Are you ready to start the upgrade ? [y/n] :
    ```

NOTE *Below is the upgrade procedure for a Sun-3 standalone system called godzilla which was running 3.0fcs before the upgrade. All optional software is chosen to be loaded in this example.*

```
Beginning 3.0FCS to 3.2FCS upgrade for the MC68020 architecture.

Saving administrative files from 3.0FCS :
Start preserving godzilla's files.
All done preserving files.

Changing directory to "/".

Extracting "root" files from "/dev/nrmt0" release tape.

[ This takes approximately 2 1/2 minutes with mt/xt and
approximately 2 1/2 minutes with ar/st. ]

Extracting "pub" files from "/dev/nrmt0" release tape.

[ This takes approximately 3 minutes with mt/xt and
approximately 7 minutes with ar/st. ]

Changing directory to "/usr".

Extracting "sys" files from "/dev/nrmt0" release tape.

[ This takes approximately 8 minutes with mt/xt and
approximately 12 minutes with ar/st. ]

Extracting "user" files from "/dev/nrmt0" release tape.
```

```
[ This takes approximately 14 minutes with mt/xt and
approximately 15 minutes with ar/st. ]

Extracting "Networking" files from "/dev/nrmt0" release tape.

[ This takes approximately 5 minutes with mt/xt and
approximately 4 minutes with ar/st. ]

Extracting "Debugging" files from "/dev/nrmt0" release tape.

[ This takes approximately 3 minutes with mt/xt and
approximately 3 1/3 minutes with ar/st. ]

Extracting "Suntools_users" files from "/dev/nrmt0" release tape.

[ This takes approximately 4 1/2 minutes with mt/xt and
approximately 3 5/6 minutes with ar/st. ]

Extracting "Suntools_programmers" files from "/dev/nrmt0" release tape.

[ This takes approximately 2 1/2 minute with mt/xt and
approximately 4 1/4 minutes with ar/st. ]

Extracting "Suntools_source" files from "/dev/nrmt0" release tape.

[ This takes approximately 2 1/2 minutes with mt/xt and
approximately  3 minutes with ar/st. ]

Extracting "Text_processing" files from "/dev/nrmt0" release tape.

[ This takes approximately 2 minute with mt/xt and
approximately 3 minutes with ar/st. ]

Extracting "Setup" files from "/dev/nrmt0" release tape.

[ This takes approximately 1 1/2 minutes with mt/xt and
approximately 3 1/2 minutes with ar/st. ]

Extracting "Stand_diag" files from "/dev/nrmt0" release tape.

[ This takes approximately 1 1/2 minutes with mt/xt and
approximately 3 minutes with ar/st. ]

Extracting "Fortran" files from "/dev/nrmt0" release tape.

[ This takes approximately 2 minute with mt/xt and
approximately 3 minutes with ar/st. ]

Extracting "Usr_diag" files from "/dev/nrmt0" release tape.

[ This takes approximately 3 minute with mt/xt and
approximately 5 minutes with ar/st. ]
```

```
Extracting "Graphics" files from "/dev/nrmt0" release tape.

[ This takes approximately 4 minutes with mt/xt and
approximately 5 minutes with ar/st. ]

Extracting "Pascal" files from "/dev/nrmt0" release tape.

[ This takes approximately 4 minutes with mt/xt and
approximately 4 minutes with ar/st. ]

Extracting "Profiled" files from "/dev/nrmt0" release tape.

[ This takes approximately 2 minutes with mt/xt and
approximately 5 minutes with ar/st. ]

Extracting "Uucp" files from "/dev/nrmt0" release tape.

[ This takes approximately 2 minute with mt/xt and
approximately 5 minutes with ar/st. ]

Extracting "System_V" files from "/dev/nrmt0" release tape.

[ This takes approximately 5 minutes with mt/xt and
approximately 6 minutes with ar/st. ]

Extracting "Man" files from "/dev/nrmt0" release tape.

[ This takes approximately 15 minutes with mt/xt and
approximately 17 minutes with ar/st. ]

Extracting "Demo" files from "/dev/nrmt0" release tape.

[ This takes approximately 2 minute with mt/xt and
approximately 4 minutes with ar/st. ]

Extracting "Games" files from "/dev/nrmt0" release tape.

[ This takes approximately 2 minutes with mt/xt and
approximately 5 minutes with ar/st. ]

Extracting "Vtroff" files from "/dev/nrmt0" release tape.

[ This takes approximately 4 minutes with mt/xt and
approximately 8 minutes with ar/st. ]

Restoring administrative files from 3.0FCS release :
Start restoring godzilla's files.
All done restoring files.

Removing administrative files from 3.0FCS release :
Start cleaning godzilla's files.
All done cleaning files.
```

```
Checking filesystems :
/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0g: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)
3.0FCS to 3.2FCS upgrade completed.
Reboot your system and configure a kernel for your system.
#
```

15. Abort the system by typing 'L1-A' or hit <break>.

16. Boot system kernel from the disk by typing

```
>b
```

17. Reconfigure a kernel for your system. Refer to *Chapter 3* for details.

2.3. Example Upgrade Walkthroughs

This section contains example upgrade walkthroughs for the following system configurations:

- Standalone System with a Local Tape Drive
- Standalone System with a Remote Tape Drive
- Homogeneous Server with a Local Tape Drive
- Homogeneous Server with a Remote Tape Drive
- Heterogeneous Server with a Local Tape Drive
- Heterogeneous Server with a Remote Tape Drive

Upgrade A Standalone System with a Local Tape Drive

Assume the standalone system has a local 1/2" tape drive (`mt`) and the system is running 3.2*pilot*. Below is a list of the optional software that currently exists on the disk:

Suntools_users
Suntools_source
Fortran
Pascal
Profiled
Man
Suntools_programmers

```
>b mt()
Boot: mt(,,4)
From: mt(,,5)
To: xy(,,1)
Boot: xy(,,1)vmunix -as .
      .
      .
      .
root device ? xy0*
```

```
      .
      .
      .
#
# cd /usr/etc/upgrade
# UPGRADE

Enter system type ? [standalone | server]: standalone

Enter tape drive type ? [local | remote]: local

Enter tape type ? [ar | st | mt | xt]: mt

Enter yp type for machine? [master | slave | client | none]: client

Enter root disk partition for the MC68020 architecture (e.g. xy0a)?
/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0g: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)

Select optional software for the MC68020 architecture :
Do you want to install "Debugging"? [y/n]: y
Do you want to install "Text_processing"? [y/n]: y
Do you want to install "Setup"? [y/n]: y
Do you want to install "Stand_diag"? [y/n]: y
Do you want to install "Usr_diag"? [y/n]: y
Do you want to install "Graphics"? [y/n]: y
Do you want to install "Uucp"? [y/n]: y
Do you want to install "System_V"? [y/n]: y
Do you want to install "Demo"? [y/n]: y
Do you want to install "Games"? [y/n]: y
Do you want to install "Vtroff"? [y/n]: y

Do you want to remove files saved from 3.2PILOT after the upgrade ? [y/n]: y

Are you ready to start the upgrade ? [y/n] : y

Beginning 3.2PILOT to 3.2FCS upgrade for the MC68020 architecture.

Saving administrative files from 3.2PILOT :
Start preserving godzilla's files.
All done preserving files.

Changing directory to "/".

Extracting "root" files from "/dev/nrmt0" release tape.

Extracting "pub" files from "/dev/nrmt0" release tape.

Changing directory to "/usr".

Extracting "sys" files from "/dev/nrmt0" release tape.

Extracting "user" files from "/dev/nrmt0" release tape.
```

```
Extracting "Networking" files from "/dev/nrmt0" release tape.

     .
     .
     .

Extracting "Vtroff" files from "/dev/nrmt0" release tape.

Restoring administrative files from 3.2PILOT release :
Start restoring godzilla's files.
All done restoring files.

Removing administrative files from 3.2PILOT release :
Start cleaning godzilla's files.
All done cleaning files.

Checking filesystems :
/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0g: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)

3.2PILOT to 3.2FCS upgrade completed.
Reboot your system and configure a kernel for your system.
#
```

Upgrade A Standalone System With Remote Tape Drive

Assume the host number of the system we are booting from is 114. 114 in decimal is equal to 72 in hexidecimal. We have a Sun-3 machine and it is running 3.0*fcs*(3.0). Therefore, the pub number is 1. Below is a list of the optional software that currently exists on the disk:

Suntools_users
Suntools_source
Fortran
Pascal
Profiled
Man

```
>b ie(,72)boot -a
Boot: ie(,72)stand/copy
From: ie(,72,1)miniroot
To: xy(,,1)
Boot: xy(,,1)vmunix -as
     .
     .
     .

root device ? xy0*

     .
     .
     .
```

```
#
# cd /usr/etc/upgrade
# UPGRADE

Enter system type ? [standalone | server]: standalone

Enter tape drive type ? [local | remote]: remote

Enter host of remote drive ? pebbles

Enter ethernet type of this system ? [ec | ie | le] : ie

Enter tape type ? [ar | st | mt | xt]: st

Enter yp type for machine? [master | slave | client | none]: none

Enter root disk partition for MC68020 architecture (e.g xy0a)?

/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0g: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)

Select optional software for the MC68020 architecture :
Do you want to install "Stand_diag"? [y/n]: y
Do you want to


Do you want to install "System_V"? [y/n]: y
Do you want to install "Demo"? [y/n]: y
Do you want to install "Games"? [y/n]: y
Do you want to install "Vtroff"? [y/n]: y

Do you want to remove files saved from 3.0FCS after the upgrade ? [y/n]: y

Are you ready to start the upgrade ? [y/n] : y

Beginning 3.0FCS to 3.2FCS upgrade for the MC68020 architecture.
Saving administrative files from 3.0FCS :
Start preserving godzilla's files.
All done preserving files.

Changing directory to "/".

Extracting "root" files from "/dev/nrmt0" release tape.

Extracting "pub" files from "/dev/nrmt0" release tape.

Changing directory to "/usr".

Extracting "sys" files from "/dev/nrmt0" release tape.

Extracting "user" files from "/dev/nrmt0" release tape.

Extracting "Networking" files from "/dev/nrmt0" release tape.
```

```
Extracting "Debugging" files from "/dev/nrmt0" release tape.

     .
     .
     .

Extracting "Vtroff" files from "/dev/nrmt0" release tape.

Restoring administrative files from 3.0FCS release :
Start restoring godzilla's files.
All done restoring files.

Removing administrative files from 3.0FCS release :
Start cleaning godzilla's files.
All done cleaning files.

Checking filesystems :
/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0g: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)

3.0FCS to 3.2FCS upgrade completed.
Reboot your system and configure a kernel for your system.
#
```

Upgrade A Homogeneous Server With Local Tape Drive

Assume we have a server called godzilla with three Sun-3 clients : frodo, grendel and sofia. In this case, all 3.0 optional software exists on the disk.

```
>b mt()
Boot: mt(,,4)
From: mt(,,5)
To: xy(,,1)
Boot: xy(,,1)vmunix -as

     .
     .
     .

root device ? xy0*

     .
     .

#
# cd /usr/etc/upgrade
# UPGRADE

Enter system type ? [standalone | server]: server

Enter server type ? [homo | heter]: homo

Enter tape drive type ? [local | remote]: local
```

```
Enter tape type ? [ar | st | mt | xt]: mt

Enter yp type for machine? [master | slave | client | none]: slave

Enter root disk partition for MC68020 architecture (e.g. xy0a)?

/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0h: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)
/dev/rxy0f: 83 files, 3495 used, 840 free (18 frags, 104 blocks)

Select optional software for the MC68020 architecture:
Do you want to install "System V"? [y/n]: y

Do you want to remove files saved from 3.0FCS after the upgrade ? [y/n]: y

Are you ready to start the upgrade ? [y/n] : y

Beginning 3.0FCS to 3.2FCS upgrade for the MC68020 architecture.

Saving administrative files from 3.0FCS :
Start preserving frodo's files.
Start preserving grendel's files.
Start preserving sofia's files.
Start preserving godzilla's files.
All done preserving files.

Changing directory to "/".

Extracting "root" files from "/dev/nrmt0" release tape.

Changing directory to "/pub".

Extracting "pub" files from "/dev/nrmt0" release tape.

Beginning 3.0FCS to 3.2FCS upgrade on MC68020 diskless clients.

Beginning 3.0FCS to 3.2FCS upgrade on client frodo.

[ Ignore this message: tar: can't create ./lib/: No such file or directory

Restoring adm files for client frodo :
Start restoring frodo's files.
All done restoring files.

Completed 3.0FCS to 3.2FCS upgrade on client frodo.

Beginning 3.0FCS to 3.2FCS upgrade on client grendel.

[ Ignore this message: tar: can't create ./lib/: No such file or directory

Restoring adm files for client grendel :
Start restoring grendel's files.
All done restoring files.
```

```
Completed 3.0FCS to 3.2FCS upgrade on client grendel.

Beginning 3.0FCS to 3.2FCS upgrade on client sofia.

[ Ignore this message: tar: can't create ./lib/: No such file or directory ]

Restoring adm files for client sofia :
Start restoring sofia's files.
All done restoring files.

Completed 3.0FCS to 3.2FCS upgrade on client sofia.

Changing directory to "/usr".

Extracting "sys" files from "/dev/nrmt0" release tape.

Extracting "user" files from "/dev/nrmt0" release tape.

Extracting "Networking" files from "/dev/nrmt0" release tape.

Extracting "Debugging" files from "/dev/nrmt0" release tape.
     .
     .
     .
     .
Extracting "Vtroff" files from "/dev/nrmt0" release tape.

Restoring administrative files from 3.0FCS release :
Start restoring godzilla's files.
All done restoring files.

Removing administrative files from 3.0FCS release :
Start cleaning frodo's files.
Start cleaning grendel's files.
Start cleaning sofia's files.
Start cleaning godzilla's files.
All done cleaning files.

Checking filesystems :
/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0h: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)
/dev/rxy0f: 83 files, 3495 used, 840 free (18 frags, 104 blocks)

3.0FCS to 3.2FCS upgrade completed.
Reboot your system and configure a kernel for your system.
#
```

Upgrade A Homogeneous Server With Remote Tape Drive

Assume the host number of the system we are booting from is 114. 114 in decimal is equal to 72 in hexidecimal. We have a Sun-3 machine and the system is running 3.0+*extensions*. Therefore, the pub number is 1. In this case, none of the optional software exists on the disk.

```
>b ie(,72)boot -a
Boot: ie(,72)stand/copy
From: ie(,72,1)miniroot
To: xy(,,1)
Boot: xy(,,1)vmunix -as

     .
     .
     .

root device ? xy0*

     .
     .
     .

#
# cd /usr/etc/upgrade
# UPGRADE

Enter system type ? [standalone | server]: server

Enter server type ? [homo | heter]: homo

Enter tape drive type ? [local | remote]: remote

Enter host of remote drive ? pebbles

Enter ethernet type of this system ? [ec | ie | le] : ie

Enter tape type ? [ar | st | mt | xt]: mt

Enter yp type for machine? [master | slave | client | none]: client

Enter root disk partition for MC68020 architecture (e.g. xy0a)?

/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0h: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)
/dev/rxy0f: 83 files, 3495 used, 840 free (18 frags, 104 blocks)

Select optional software for the MC68020 architecture :
Do you want to install "Suntools_users"? [y/n]: y
Do you want to install "Suntools_programmers"? [y/n]: y
Do you want to install "Suntools_source"? [y/n]: y
Do you want to install "Stand_diag"? [y/n]: y
Do you want to install "Fortran"? [y/n]: y
Do you want to install "Usr_diag"? [y/n]: y
Do you want to install "Graphics"? [y/n]: y
```

```
Do you want to install "Pascal"? [y/n]: y
Do you want to install "Profiled"? [y/n]: y
Do you want to install "Uucp"? [y/n]: y
Do you want to install "System_V"? [y/n]: y
Do you want to install "Man"? [y/n]: y
Do you want to install "Demo"? [y/n]: y
Do you want to install "Games"? [y/n]: y
Do you want to install "Vtroff"? [y/n]: y

Do you want to remove files saved from 3.2+extensions after the upgrade ? [y/n

Are you ready to start the upgrade ? [y/n] : y

Beginning 3.2+extensions to 3.2FCS upgrade for the MC68020 architecture.

Saving administrative files from 3.0FCS :
Start preserving frodo's files.
Start preserving grendel's files.
Start preserving sofia's files.
Start preserving godzilla's files.
All done preserving files.

Changing directory to "/".

Extracting "root" files from "/dev/nrmt0" release tape.

Changing directory to "/pub".

Extracting "pub" files from "/dev/nrmt0" release tape.

Beginning 3.2+extensions to 3.2FCS upgrade on MC68020 diskless clients.

Beginning 3.2+extensions to 3.2FCS upgrade on client frodo.

[ Ignore this message: tar: can't create ./lib/: No such file or directory ]

Restoring adm files for client frodo :
Start restoring frodo's files.
All done restoring files.

Completed 3.2+extensions to 3.2FCS upgrade on client frodo.

Beginning 3.2+extensions to 3.2FCS upgrade on client grendel.

[ Ignor this message: tar: can't create ./lib/: No such file or directory ]

Restoring adm files for client grendel :
Start restoring grendel's files.
All done restoring files.

Completed 3.2+extensions to 3.2FCS upgrade on client grendel.

Beginning 3.2+extensions to 3.2FCS upgrade on client sofia.
```

```
[ Ignore this message: tar: can't create ./lib/: No such file or directory ]

Restoring adm files for client sofia :
Start restoring sofia's files.
All done restoring files.

Completed 3.2+extensions to 3.2FCS upgrade on client sofia.

Changing directory to "/usr".

Extracting "sys" files from "/dev/nrmt0" release tape.

Extracting "user" files from "/dev/nrmt0" release tape.

Extracting "Networking" files from "/dev/nrmt0" release tape.

Extracting "Debugging" files from "/dev/nrmt0" release tape.

     .
     .
     .

Extracting "Vtroff" files from "/dev/nrmt0" release tape.

Restoring administrative files from 3.2+extensions release :
Start restoring godzilla's files.
All done restoring files.

Removing administrative files from 3.2+extensions release :
Start cleaning frodo's files.
Start cleaning grendel's files.
Start cleaning sofia's files.
Start cleaning godzilla's files.
All done cleaning files.

Checking filesystems :
/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0h: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)
/dev/rxy0f: 83 files, 3495 used, 840 free (18 frags, 104 blocks)

3.2+extensions to 3.2FCS upgrade completed.
Reboot your system and configure a kernel for your system.
#
```

Upgrade A Heterogeneous Server With Local Tape Drive

Assume we have a server called godzilla with two Sun-3 clients (frodo and grendel) and one Sun-2 client (sofia). In this case we assume that none of the optional software from both architectures exists on the disk.

```
>b mt()
Boot: mt(,,4)
From: mt(,,5)
```

```
To: xy(,,1)
Boot: xy(,,1)vmunix -as

     .
     .
     .

root device ? xy0*

     .
     .

#
# cd /usr/etc/upgrade
# UPGRADE

Enter system type ? [standalone | server]: server

Enter server type ? [homo | heter]: heter

Enter tape drive type ? [local | remote]: local

Enter tape type ? [ar | st | mt | xt]: mt

Enter yp type for machine? [master | slave | client | none]: slave

Enter root disk partition for the MC68020 architecture (e.g. xy0a)?

/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0h: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)
/dev/rxy0f: 83 files, 3495 used, 840 free (18 frags, 104 blocks)
/dev/rxy0g: 1163 files, 18150 used, 40393 free (73 frags, 5040 blocks)
/dev/rxy0e: 83 files, 3574 used, 905 free (17 frags, 111 blocks)

Select optional software for the MC68020 architecture :
Do you want to install "Suntools_users"? [y/n]: y
Do you want to install "Suntools_programmers"? [y/n]: y
Do you want to install "Suntools_source"? [y/n]: y
Do you want to install "Stand_diag"? [y/n]: y
Do you want to install "Fortran"? [y/n]: y
Do you want to install "Usr_diag"? [y/n]: y
Do you want to install "Graphics"? [y/n]: y
Do you want to install "Pascal"? [y/n]: y
Do you want to install "Profiled"? [y/n]: y
Do you want to install "Uucp"? [y/n]: y
Do you want to install "System_V"? [y/n]: y
Do you want to install "Man"? [y/n]: y
Do you want to install "Demo"? [y/n]: y
Do you want to install "Games"? [y/n]: y
Do you want to install "Vtroff"? [y/n]: y

Select optional software for the MC68010 architecture :
Do you want to install "Suntools_users"? [y/n]: y
```

```
Do you want to install "Suntools_programmers"? [y/n]: y
Do you want to install "Suntools_source"? [y/n]: y
Do you want to install "Stand_diag"? [y/n]: y
Do you want to install "Fortran"? [y/n]: y
Do you want to install "Usr_diag"? [y/n]: y
Do you want to install "Graphics"? [y/n]: y
Do you want to install "Pascal"? [y/n]: y
Do you want to install "Profiled"? [y/n]: y
Do you want to install "Uucp"? [y/n]: y
Do you want to install "System_V"? [y/n]: y
Do you want to install "Man"? [y/n]: y
Do you want to install "Demo"? [y/n]: y
Do you want to install "Games"? [y/n]: y
Do you want to install "Vtroff"? [y/n]: y

Do you want to remove files saved from 3.0FCS after the upgrade ? [y/n]: y

Are you ready to start the upgrade ? [y/n] : y

Beginning 3.0FCS to 3.2FCS upgrade for the MC68020 architecture.

Saving administrative files from 3.0FCS :
Start preserving frodo's files.
Start preserving grendel's files.
Start preserving sofia's files.
Start preserving godzilla's files.
All done preserving files.

Changing directory to "/".

Extracting "root" files from "/dev/nrmt0" release tape.

Changing directory to "/pub".

Extracting "pub" files from "/dev/nrmt0" release tape.

Beginning 3.0FCS to 3.2FCS upgrade on MC68020 diskless clients.

Beginning 3.0FCS to 3.2FCS upgrade on client frodo.

[ Ignore this message: tar: can't create ./lib/: No such file or directory ]

Restoring adm files for client frodo :
Start restoring frodo's files.
All done restoring files.

Completed 3.0FCS to 3.2FCS upgrade on client frodo.

Beginning 3.0FCS to 3.2FCS upgrade on client grendel.

[ Ignore this message: tar: can't create ./lib/: No such file or directory ]

Restoring adm files for client grendel :
```

```
Start restoring grendel's files.
All done restoring files.

Completed 3.0FCS to 3.2FCS upgrade on client grendel.

Changing directory to "/usr".

Extracting "sys" files from "/dev/nrmt0" release tape.

Extracting "user" files from "/dev/nrmt0" release tape.

Extracting "Networking" files from "/dev/nrmt0" release tape.

Extracting "Debugging" files from "/dev/nrmt0" release tape.

     .
     .
     .

Extracting "Vtroff" files from "/dev/nrmt0" release tape.

Beginning 3.0FCS to 3.2FCS upgrade for the MC68010 architecture.

Changing directory to "/pub.MC68010".

Extracting "pub" files from "/dev/nrmt0" release tape.

Beginning 3.0FCS to 3.2FCS upgrade on MC68010 diskless clients.

Beginning 3.0FCS to 3.2FCS upgrade on client sofia.

[ Ignore this message: tar: can't create ./lib/: No such file or directory ]

Restoring adm files for client sofia :
Start restoring sofia's files.
All done restoring files.

Completed 3.0FCS to 3.2FCS upgrade on client sofia.

Changing directory to "/usr.MC68010".

Extracting "sys" files from "/dev/nrmt0" release tape.

Extracting "user" files from "/dev/nrmt0" release tape.

Extracting "Networking" files from "/dev/nrmt0" release tape.

Extracting "Debugging" files from "/dev/nrmt0" release tape.

     .
     .
     .
```

```
Extracting "Vtroff" files from "/dev/nrmt0" release tape.

Restoring administrative files from 3.0FCS release :
Start restoring godzilla's files.
All done restoring files.

Removing administrative files from 3.0FCS release :
Start cleaning frodo's files.
Start cleaning grendel's files.
Start cleaning sofia's files.
Start cleaning godzilla's files.
All done cleaning files.

Checking filesystems :
/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0h: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)
/dev/rxy0f: 83 files, 3495 used, 840 free (18 frags, 104 blocks)
/dev/rxy0g: 1163 files, 18150 used, 40393 free (73 frags, 5040 blocks)
/dev/rxy0e: 83 files, 3574 used, 905 free (17 frags, 111 blocks)

3.0FCS to 3.2FCS upgrade completed.
Reboot your system and configure a kernel for your system.
#
```

Upgrade A Heterogeneous Server With Remote Tape Drive

Assume the host number of the system we are booting from is 114. 114 in decimal is equal to 72 in hexidecimal. We have a Sun-3 machine. Therefore, the pub number is 1. In this case, we assume none of the optional software from both architectures exists on the disk.

```
>b ie(,72)boot -a
Boot: ie(,72)stand/copy
From: ie(,72,1)miniroot
To: xy(,,1)
Boot: xy(,,1)vmunix -as

     .

root device ? xy0*

     .
     .

#
# cd /usr/etc/upgrade
# UPGRADE

Enter system type ? [standalone | server]: server

Enter server type ? [homo | heter]: heter
```

```
Enter tape drive type ? [local | remote]: remote

Enter host of remote drive ? pebbles

Enter ethernet type of this system ? [ec | ie | le] : ie

Enter tape type ? [ar | st | mt | xt]: st

Enter yp type for machine? [master | slave | client | none]: slave

Enter root disk partition for the MC68020 architecture  (e.g. xy0a)?

/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0h: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)
/dev/rxy0f: 83 files, 3495 used, 840 free (18 frags, 104 blocks)
/dev/rxy0g: 1163 files, 18150 used, 40393 free (73 frags, 5040 blocks)
/dev/rxy0e: 83 files, 3574 used, 905 free (17 frags, 111 blocks)

Select optional software for the MC68020 architecture :
Do you want to install "Suntools_users"? [y/n]: y
Do you want to install "Suntools_programmers"? [y/n]: y
Do you want to install "Suntools_source"? [y/n]: y
Do you want to install "Stand_diag"? [y/n]: y
Do you want to install "Fortran"? [y/n]: y
Do you want to install "Usr_diag"? [y/n]: y
Do you want to install "Graphics"? [y/n]: y
Do you want to install "Pascal"? [y/n]: y
Do you want to install "Profiled"? [y/n]: y
Do you want to install "Uucp"? [y/n]: y
Do you want to install "System_V"? [y/n]: y
Do you want to install "Man"? [y/n]: y
Do you want to install "Demo"? [y/n]: y
Do you want to install "Games"? [y/n]: y
Do you want to install "Vtroff"? [y/n]: y

Select optional software for the MC68010 architecture :
Do you want to install "Suntools_users"? [y/n]: y
Do you want to install "Suntools_programmers"? [y/n]: y
Do you want to install "Suntools_source"? [y/n]: y
Do you want to install "Stand_diag"? [y/n]: y
Do you want to install "Fortran"? [y/n]: y
Do you want to install "Usr_diag"? [y/n]: y
Do you want to install "Graphics"? [y/n]: y
Do you want to install "Pascal"? [y/n]: y
Do you want to install "Profiled"? [y/n]: y
Do you want to install "Uucp"? [y/n]: y
Do you want to install "System_V"? [y/n]: y
Do you want to install "Man"? [y/n]: y
Do you want to install "Demo"? [y/n]: y
Do you want to install "Games"? [y/n]: y
Do you want to install "Vtroff"? [y/n]: y

Do you want to remove files saved from 3.0FCS after the upgrade ? [y/n]: y
```

```
Are you ready to start the upgrade ? [y/n] : y

Beginning 3.0FCS to 3.2FCS upgrade for the MC68020 architecture.

Saving administrative files from 3.0FCS :
Start preserving frodo's files.
Start preserving grendel's files.
Start preserving sofia's files.
Start preserving godzilla's files.
All done preserving files.

Changing directory to "/".

Extracting "root" files from "/dev/nrst0" release tape.

Changing directory to "/pub".

Extracting "pub" files from "/dev/nrst0" release tape.

Beginning 3.0FCS to 3.2FCS upgrade on MC68020 diskless clients.

Beginning 3.0FCS to 3.2FCS upgrade on client frodo.

[ Ignore this message: tar: can't create ./lib/: No such file or directory ]

Restoring adm files for client frodo :
Start restoring frodo's files.
All done restoring files.

Completed 3.0FCS to 3.2FCS upgrade on client frodo.

Beginning 3.0FCS to 3.2FCS upgrade on client grendel.

[ Ignore this message: tar: can't create ./lib/: No such file or directory ]

Restoring adm files for client grendel :
Start restoring grendel's files.
All done restoring files.

Completed 3.0FCS to 3.2FCS upgrade on client grendel.

Changing directory to "/usr".

Extracting "sys" files from "/dev/nrst0" release tape.

Extracting "user" files from "/dev/nrst0" release tape.

Extracting "Networking" files from "/dev/nrst0" release tape.

Extracting "Debugging" files from "/dev/nrst0" release tape.

      .
      .
```

```
Extracting "Vtroff" files from "/dev/nrst0" release tape.

Beginning 3.0FCS to 3.2FCS upgrade for the MC68010 architecture.

Changing directory to "/pub.MC68010".

Extracting "pub" files from "/dev/nrst0" release tape.

Beginning 3.0FCS to 3.2FCS upgrade on MC68010 diskless clients.

Beginning 3.0FCS to 3.2FCS upgrade on client sofia.

[ Ignore this message: tar: can't create ./lib/: No such file or directory ]

Restoring adm files for client sofia :
Start restoring sofia's files.
All done restoring files.

Completed 3.0FCS to 3.2FCS upgrade on client sofia.

Changing directory to "/usr.MC68010".

Extracting "sys" files from "/dev/nrst0" release tape.

Extracting "user" files from "/dev/nrst0" release tape.

Extracting "Networking" files from "/dev/nrst0" release tape.

Extracting "Debugging" files from "/dev/nrst0" release tape.

     .
     .
     .

Extracting "Vtroff" files from "/dev/nrst0" release tape.

Restoring administrative files from 3.0FCS release :
Start restoring godzilla's files.
All done restoring files.

Removing administrative files from 3.0FCS release :
Start cleaning frodo's files.
Start cleaning grendel's files.
Start cleaning sofia's files.
Start cleaning godzilla's files.
All done cleaning files.

Checking filesystems :
/dev/rxy0a: 516 files, 2575 used, 4896 free (16 frags, 610 blocks)
/dev/rxy0h: 1678 files, 18810 used, 39733 free (69 frags, 4958 blocks)
/dev/rxy0f: 83 files, 3495 used, 840 free (18 frags, 104 blocks)
```

```
/dev/rxy0g: 1163 files, 18150 used, 40393 free (73 frags, 5040 blocks)
/dev/rxy0e: 83 files, 3574 used, 905 free (17 frags, 111 blocks)

3.0FCS to 3.2FCS upgrade completed.
Reboot your system and configure a kernel for your system.
#
```

2.4. How To Upgrade A Client Partition To 3.2*fcs*

This section explains how you can upgrade a partition to 3.2*fcs* software that was not upgraded to 3.2*fcs* during the upgrade of the server. Here is a sample /etc/nd.local file :

```
#
# These lines added by the Sun Setup Program
#
clear
version 1
user 0 1 /dev/xy0f 0 11960 -1
user client1 0 /dev/xy0c 50600 10120 0
user client1 1 /dev/xy0c 60720 39560 -1
#user client2 0 /dev/xy0c 100280 10120 1
#user client2 1 /dev/xy0c 110400 39560 -1
user client3 0 /dev/xy0c 149960 10120 2
user client3 1 /dev/xy0c 160080 39560 -1
son
#
# End of lines added by the Sun Setup Program
#
```

The Steps to Upgrade Client2 to 3.2*fcs*.

1. Make sure `/etc/hosts` on the server contains the internet address of client2. You need to make sure `/etc/hosts` on the yp master contains the internet address of client2 if you are running yellow pages.
2. Make sure `/etc/ethers` on the server contains the ethernet address of client2. You need to make sure `/etc/ethers` on the yp master contains the ethernet address of client2 if you are running yellow pages.
3. Fix /etc/nd.local by removing '#' from these two lines.

   ```
   #user client2 0 /dev/xy0c 100280 10120 1
   #user client2 1 /dev/xy0c 110400 39560 -1
   ```

4. Run nd.

   ```
   server# /etc/nd < /etc/nd.local
   ```

5. Mount client partition.

   ```
   server# mount /dev/ndl1 /mnt
   ```

6. Go to the client partition.

```
server# cd /mnt
```

7. Save all administrative files from the current release.

```
server# cp .login .login.save
server# cp .cshrc .cshrc.save
server# cp .rhosts .rhosts.save
server# cp etc/passwd etc/passwd.save
server# cp etc/printcap etc/printcap.save
server# cp etc/fstab etc/fstab.save
server# cp etc/hosts.equiv etc/hosts.equiv.save
server# cp etc/hosts etc/hosts.save
server# cp etc/rc.local etc/rc.local.save
server# cp etc/rc.boot etc/rc.boot.save
server# cp private/usr/lib/crontab private/usr/lib/crontab.save
server# cp private/usr/lib/sendmail.cf private/usr/lib/sendmail.cf.save
```

8. Mount tape #1 of release tape to the tape drive.
9. Position tape to the right file on the release tape.

```
server# mt -f /dev/nrtape0 fsf 8
```

Refer to *Table* 2-3 for tape devices.

10. Extract files from the tape.

```
server# tar xvpf /dev/nrtape0
server# mt -f /dev/nrtape0 rew
```

Refer to *Table* 2-3 for tape devices.

11. Restore all administrative files by comparing the files saved and the new files and merge them by hand.
12. Get out of the client partition.

```
server# cd /
```

13. Unmount the client partition.

```
server# umount /dev/ndl1
```

2.5. How To Restart The Upgrade Procedures

If for any reason UPGRADE terminated before it was completed, you can restart the upgrade procedure depending on the state of the system. Here are some suggestions. Please do not try these suggestions unless you fully understand the instructions.

1. If UPGRADE terminated before you saw this message on the console:

```
Extracting "root" files from "/dev/nrtape" release tape.
```

On the console, do the following:

- make sure you are still in the miniroot
- make sure no file systems are mounted.
- **# cd /usr/etc/upgrade**
- **# UPGRADE**

2. If `UPGRADE` terminated after you saw this message

```
Extracting "root" files from "/dev/nrtape" release tape.
```

On the console, do the following:

- make sure you are still in the miniroot
- make sure root and user file systems are mounted to `/a` and `/a/usr`
- go through all files under `/a`, `/a/etc` and `/a/private/usr/lib` ending with "`.save`" and make sure the files contain correct information. If the files do not contain correct information and you do not know how to recover the files, you will not be able to continue with the upgrade procedure and you will have to use *Setup* to rebuild your system from scratch.
- Make sure `/a/usr/sys/conf/RELEASE` contains the release number of the system before you started the upgrade (3.0FCS or 3.2PILOT)
- If your system is a server, you need to go through each client partition to make sure all files contain correct information for the partition.
- If the files are correct, run these commands:

```
# cd usr/etc/upgrade
# adm_tool restore name machinetype yptype release
```

Usage: adm_tool op name machinetype yptype release
where:

op = save, restore or clean
name = name to be performed the operation
machinetype = standalone, server or diskless
yptype = master, slave, client or none
release = 3.0FCS or 3.2PILOT

- If the system is a server, go through each partition and do the following:

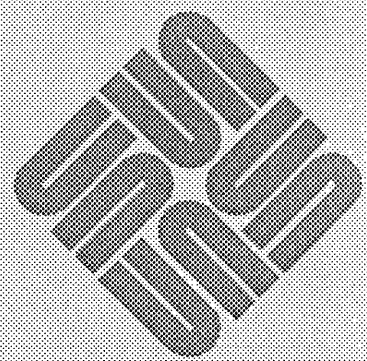
```
# mkdir /a/client
# mount /dev/ndl# /a/client
# cd /a/client
# /usr/etc/upgrade/adm_tool restore client_name diskless yptype release
# cd /
# umount /dev/ndl#
```

Replace '#' with a positive integer

- unmount root and user file systems
- make sure no file systems are mounted
- **# cd /usr/etc/upgrade**
- **# UPGRADE**

3

Reconfiguring Your Kernel



3

Reconfiguring Your Kernel

You do NOT have to reconfigure your kernel because a GENERIC kernel is now provided on the upgrade tape. However, we recommend that you reconfigure your kernel in order to save space or to customize your kernel to recognize your hardware. You can do so by using the following procedure. You must reboot your system whether you reconfigure or not.

If you are doing kernel configuration for the first time, you can use the procedures in

> *Installing UNIX on the Sun Workstation*
> (Part Number: 800-1521).

If you have previously configured a kernel, you can use the following sections to guide you through reconfiguration. The first subsection gives reconfiguration procedures for standalone machines, the second subsection addresses servers, and the third subsection is an annotated copy of the new *GENERIC* kernel configuration file; read it carefully to make sure you are including the correct device description lines for your system. The last two sections of this chapter contain specific *GENERIC* files for a Sun-2 or Sun-3 machine.

NOTE *See* General and Specific System Description Lines *sections of* Installing Unix on the Sun Workstation *for more detailed information concerning the Annotated GENERIC file.*

Kernel Reconfiguration for Standalone Systems

For standalone machines, proceed as follows.

1. Change the current directory to */sys/conf*:

   ```
   # cd /sys/conf
   ```

2. Create a kernel configuration file.

   Copy the file *GENERIC* and comment out the lines that don't apply to your system. We'll call the new file *SYS_NAME* (the name of the system). For example,

   ```
   # cp GENERIC SYS_NAME
   # chmod +w SYS_NAME
   ```

3. Edit `/sys/conf/`*SYS_NAME* to reflect your system configuration. Use the annotated copy of *GENERIC* provided in the following 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*):

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

5. Now you can save your old kernel and install your new one:

```
# mv /vmunix /vmunix.old
# cp vmunix /vmunix
# /etc/shutdown -h now
    The system goes through the halt sequence, then
    the monitor displays its prompt, at which point you
    can boot the system:
> b vmunix
```

6. If the system appears to work, this completes the upgrade procedure. 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:

```
# /etc/shutdown -h now
> b vmunix.old -s
# mv /vmunix /vmunix.oops
# mv /vmunix.old /vmunix
# ^D      [ Brings the system up multi-user ]
```

Kernel Reconfiguration for Servers

For server machines, proceed as follows.

1. Change the current directory to */sys/conf*:

```
# cd /sys/conf
```

2. Create a kernel configuration file.

   Copy the file *GENERIC* and comment out the lines that don't apply to your system. We'll call the new file *SYS_NAME* (the name of the system). For example,

```
# cp GENERIC SYS_NAME
# chmod +w SYS_NAME
```

3. Edit */sys/conf/SYS_NAME* to reflect your system configuration. Use the annotated copy of *GENERIC* provided in the next section for an explanation of these changes. Make sure you include the proper device description lines for your system.

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

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

5. Now prepare a kernel for your clients in the same way. When editing the configuration file (called *CLIENT_KERNEL_NAME* in the following), remember to include the entire set of devices used by all the machines:

```
# cd /usr.MCclient_arch/sys/conf
# cp TEMPLATE_NAME CLIENT_KERNEL_NAME
# chmod +w CLIENT_KERNEL_NAME
[ Edit CLIENT_KERNEL_NAME to reflect all clients' systems.
  Be especially careful with the device description lines. ]
# /etc/config CLIENT_KERNEL_NAME
# cd ../CLIENT_KERNEL_NAME
# 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:

```
# cd /usr/sys/SYS_NAME
# mv /vmunix /vmunix.old
# cp vmunix /vmunix
```

7. Next, install the appropriate client kernel in */pub* for the architecture.. To install the clients' kernel, **make sure all the clients are halted**, save the original kernel (if there is one), install the new kernel image in */pub*, and then test it out by booting up one of the clients:

```
# cd /usr.MCclient_arch/sys/CLIENT_KERNEL_NAME
[or wherever your client kernel is]
# mv vmunix /pub/vmunix

[ On the client machine: ]
>b vmunix
```

8. Since at this point normal system performance is a highly, but not absolutely, certain indicator of a trouble-free kernel, if your system(s) appears to work you may proceed with some confidence. You have successfully completed installation. Congratulations!

   If, on the other hand, either of the new kernels does not seem to be functioning properly, halt all systems and boot from the original kernel. Then move the faulty kernel away and re-install the original in its place. Once you are booted up on the original, you can go about trying to fix the faulty kernel.

For example, on the server:

```
# /etc/halt
> b vmunix.old -s
# cd /
# mv vmunix vmunix.bad
# mv vmunix.old vmunix
# ^D                [ Brings the system up multi-user ]
```

For clients, halt all the clients on the server. You will have to correct the problem from the server.

On the server:

```
# cd /pub.MCclient_arch
# mv vmunix vmunix.bad
# mv vmunix.old vmunix
```

You may now boot up the clients and allow them to run while or until a new client kernel is made and ready to install; or if the clients can remain down, build and install a new client kernel now.

Kernel Reconfiguration — an Annotated Copy of a *GENERIC* File

The following is an EXAMPLE of an annotated copy of a *GENERIC* file to help you identify the lines you need to include in your own system configuration file.

NOTE *For the specific GENERIC files for a Sun-2 or Sun-3 machine, see* Section 3.1 *(Sun-2) or* Section 3.2 *(Sun-3) of this chapter.*

The comments explain the device and pseudo-device lines, and may also refer you to the reference manual entry which covers the device in question. If the comments say the line is **mandatory**, the line *must* be included in every system configuration file, either exactly as it stands, or, if commentary indicates variables, with the variables adjusted to fit your system.

NOTE *A number of parameters relating to the System V Inter-Process Communication (IPC) extensions may also be tuned in the configuration file. These parameters do not appear in the GENERIC file but are documented in the* System V Overview.

NOTE ** You need not include all machine types, only the machine type(s) that you may be running.*

Configuration Line	*Comments*	*Description*
#		
# GENERIC SUN-3		
#		
machine "sun3"	**mandatory**	Identifies the specific machine
cpu "SUN3_160"	**mandatory***	Identifies the specific CPU type (Sun-3/160, Sun-3/180, or Sun-3/75)
cpu "SUN3_50"	**mandatory***	Identifies the specific CPU type
cpu "SUN3_260"	**mandatory***	Identifies the specific CPU type (Sun-3/260 or Sun-3/280)
cpu "SUN3_110"	**mandatory***	Identifies the specific CPU type (Sun-3/110
ident GENERIC	**mandatory**	See *General* and *Specific System Description Lines* for information. Finally, if *SYS_NAME* contains both alpha and numeric characters 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. Can also use half hour designations.
maxusers 4	**mandatory**	Number may vary. For most systems, "4" is the proper value for maxusers. See the section *General System Description Lines* for information.
options INET	**mandatory**	Controls inclusion of Internet code — see *inet* (4). You must also include the "pseudo-device loop" lines below.
options SYSACCT	*optional*	Controls inclusion of code to do process accounting — see *acct* (2) and *acct* (5).
options QUOTA	*optional*	Controls the disk quota checking system.
options NFS	*optional*	Inclusion of NFS code.
options NIT	*optional*	Inclusion of network interface tap code
options IPCMESSAGE	*optional*	Controls inclusion of code for SystemV IPC Message Facility.
options IPCSEMAPHORE	*optional*	Controls inclusion of code for SystemV IPC Semaphore Facility.
options IPCSHMEM	*optional*	Controls inclusion of code for SystemV IPC Shared-Memory Facility.
config vmunix swap generic	**mandatory**	Specify kernel name and configuration clauses. Please see *Specific System Description Lines* for information.
pseudo-device pty	*optional*	Pseudo-tty's. Needed for network or window system.
pseudo-device bk	*optional*	Berknet line discipline for high speed tty input — see *bk* (4).
pseudo-device ether	*optional*	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	*optional*	Network disk. Necessary for servers and diskless clients, and for machines serving as remote hosts for remote installation — see *nd* (4).
pseudo-device win128	*optional*	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 dtop4	*optional*	Maximum number of screens ('desktops'). Required for window system.
pseudo-device ms3	*optional*	Maximum number of mice. Required for window system — see *ms* (4).

`pseudo-device kb3`	*optional*	Maximum number of Sun keyboards. Required if using any Sun keyboard, and for the window system.

The following are connections for machine types. These connections, in conjunction with controllers, devices, and disks for a structure that enable your system to recognize various hardware and software attached to it. For each device or controller on a bus, you need to have the bus type it is connected to listed under connections for machine type. It easiest to leave all lines for machine types that way as you add controllers and devices the connections are already in place and will be recognized by your system.

connections for machine type type 1 (SUN3_160)

```
controller      virtual 1 at nexus ?
controller      obmem 1 at nexus ?
controller      obio 1 at nexus ?
controller      vme16d16 1 at nexus ?
controller      vme24d16 1 at nexus ?
controller      vme32d16 1 at nexus ?
controller      vme16d32 1 at nexus ?
controller      vme24d32 1 at nexus ?
controller      vme32d32 1 at nexus ?
```

connections for machine type type 2 (SUN3_50)

```
controller      virtual 2 at nexus ?
controller      obmem 2 at nexus ?
controller      obio 2 at nexus ?
```

connections for machine type 3 (SUN3_260)

```
controller      virtual 3 at nexus ?
controller      obmem 3 at nexus ?
controller      obio 3 at nexus ?
controller      vme16d16 3 at nexus ?
controller      vme24d16 3 at nexus ?
controller      vme32d16 3 at nexus ?
controller      vme16d32 3 at nexus ?
controller      vme24d32 3 at nexus ?
controller      vme32d32 3 at nexus ?
```

connections for machine type type 4 (SUN3_110)

```
controller      virtual 4 at nexus ?
controller      obmem 4 at nexus ?
controller      obio 4 at nexus ?
controller      vme16d16 4 at nexus ?
controller      vme24d16 4 at nexus ?
controller      vme32d16 4 at nexus ?
controller      vme16d32 4 at nexus ?
controller      vme24d32 4 at nexus ?
controller      vme32d32 4 at nexus ?
```

The following are controllers and devices (devices, disks, and tapes) that connect to bus types. Bus types and devices

must hang off the appropriate controller, which in turn hangs off another controller until a configuration is formed that gets you to a bus type that hangs off a "nexus". On Sun system, all bus types are just considered to hang of a "nexus". For example:

disk:

```
xy0 at xyc0 drive 0
```

hangs off of controller:

```
xyc0 at vme16d16 ? csr 0xee40 priority 2 vector xyintr 0x48
```

which hangs off bus type:

```
controller      vme16d16 1 at nexus ?
```

In order to determine and note what devices are present on your machine, boot the GENERIC kernel after you have executed *Setup*. If you want, you can delete those lines that pertain to devices not on your machine. Or you can configure your file with the devices that are on other machines that will possibly want to boot from the same kernel.

The following is an example of controllers and devices you will find in a Sun-3 configuration file.

NOTE *It is not recommended that you remove the* `zs` *lines from the configuration file. These represent UARTS. If removed the system will not recognize the presence of the keyboard or serial ports.*

```
controller      xyc0 at vme16d16 ? csr 0xee40 priority 2 vector xyintr 0x48
controller      xyc1 at vme16d16 ? csr 0xee48 priority 2 vector xyintr 0x49
disk            xy0 at xyc0 drive 0
disk            xy1 at xyc0 drive 1
disk            xy2 at xyc1 drive 0
disk            xy3 at xyc1 drive 1
controller      sc0 at vme24d16 ? csr 0x200000 priority 2 vector scintr 0x40
disk            sd0 at sc0 drive 0 flags 0
disk            sd1 at sc0 drive 1 flags 0
tape            st0 at sc0 drive 32 flags 1
disk            sd2 at sc0 drive 8 flags 0
tape            st1 at sc0 drive 40 flags 1
#disk           sf0 at sc0 drive 8 flags 2
controller      si0 at vme24d16 ? csr 0x200000 priority 2 vector siintr 0x40
controller      si0 at obio ? csr 0x140000 priority 2
disk            sd0 at si0 drive 0 flags 0
disk            sd1 at si0 drive 1 flags 0
tape            st0 at si0 drive 32 flags 1
disk            sd2 at si0 drive 8 flags 0
tape            st1 at si0 drive 40 flags 1
#disk           sf0 at si0 drive 8 flags 2
device          zs0 at obio ? csr 0x20000 flags 3 priority 3
device          zs1 at obio ? csr 0x00000 flags 0x103 priority 3
device          mti0 at vme16d16 ? csr 0x620 flags 0xffff priority 4
```

```
                vector mtiintr 0x88
device          mti1 at vme16d16 ? csr 0x640 flags 0xffff priority 4
                vector mtiintr 0x89
device          mti2 at vme16d16 ? csr 0x660 flags 0xffff priority 4
                vector mtiintr 0x8a
device          mti3 at vme16d16 ? csr 0x680 flags 0xffff priority 4
                vector mtiintr 0x8b
device          ie0 at obio ? csr 0xc0000 priority 3
device          ie1 at vme24d16 ? csr 0xe88000 priority 3 vector ieintr 0x75
device          le0 at obio ? csr 0x120000 priority 3
controller      tm0 at vme16d16 ? csr 0xa0 priority 3 vector tmintr 0x60
controller      tm1 at vme16d16 ? csr 0xa2 priority 3 vector tmintr 0x61
tape            mt0 at tm0 drive 0 flags 1
tape            mt1 at tm1 drive 0 flags 1
controller      xtc0 at vme16d16 ? csr 0xee60 priority 3 vector xtintr 0x64
controller      xtc1 at vme16d16 ? csr 0xee68 priority 3 vector xtintr 0x65
tape            xt0 at xtc0 drive 0 flags 1
tape            xt1 at xtc1 drive 0 flags 1
device          gpone0 at vme24d16 ? csr 0x210000
device          cgtwo0 at vme24d16 ? csr 0x400000
device          cgfour0 at obmem 4 csr 0xff000000 priority 4
device          bwtwo0 at obmem 1 csr 0xff000000 priority 4
device          bwtwo0 at obmem 2 csr 0x100000 priority 4
device          bwtwo0 at obmem 3 csr 0xff000000 priority 4
device          bwtwo0 at obmem 4 csr 0xff000000
device          vpc0 at vme16d16 ? csr 0x480 priority 2 vector vpcintr 0x80
device          vpc1 at vme16d16 ? csr 0x500 priority 2 vector vpcintr 0x81
device          des0 at obio ? csr 0x1c0000
device          fpa0 at virtual ? csr 0xe0000000
```

3.1. Sun-2 GENERIC Configuration File

The following is the GENERIC configuration file for a Sun-2 system.

```
#
# GENERIC SUN2
#
machine     "sun2"
cpu         "SUN2_120"# generic for machine type 1 (Multibus)
cpu         "SUN2_50"# generic for machine type 2 (VMEbus)
ident       GENERIC
timezone    8 dst
maxusers    4
options     INET
options     SYSACCT
options     QUOTA
options     NFS
options     NIT
options     IPCMESSAGE# SystemV IPC Message Facility
options     IPCSEMAPHORE# SystemV IPC Semaphore Facility
options     IPCSHMEM# SystemV IPC Shared-Memory Facility

config      vmunixswap generic
```

```
pseudo-devicepty
pseudo-devicebk
pseudo-deviceether
pseudo-deviceloop
pseudo-devicend
pseudo-devicewin128
pseudo-devicedtop4
pseudo-devicems3
pseudo-devicekb3

# connections for machine type 1 (SUN2_120)
controller  virtual 1 at nexus ?# virtual preset
controller  obmem 1 at nexus ?# on board memory
controller  obio 1 at nexus ?# on board io
controller  mbmem 1 at nexus ?# Multibus memory
controller  mbio 1 at nexus ?# Multibus io

# connections for machine type 2 (SUN2_50)
controller  virtual 2 at nexus ?# virtual preset
controller  obmem 2 at nexus ?# on board memory
controller  obio 2 at nexus ?# on board io
controller  vme16 2 at nexus ?# 16 bit address VMEbus (16 bit data)
controller  vme24 2 at nexus ?# 24 bit address VMEbus (16 bit data)

controller  ipc0 at mbio ? csr 0x40 priority 2
controller  ipc1 at mbio ? csr 0x44 priority 2
disk        ip0 at ipc0 drive 0
disk        ip1 at ipc0 drive 1
disk        ip2 at ipc1 drive 0
disk        ip3 at ipc1 drive 1
controller  xyc0 at mbio ? csr 0xee40 priority 2
controller  xyc0 at vme16 ? csr 0xee40 priority 2 vector xyintr 0x48
controller  xyc1 at mbio ? csr 0xee48 priority 2
controller  xyc1 at vme16 ? csr 0xee48 priority 2 vector xyintr 0x49
disk        xy0 at xyc0 drive 0
disk        xy1 at xyc0 drive 1
disk        xy2 at xyc1 drive 0
disk        xy3 at xyc1 drive 1
controller  sc0 at mbmem ? csr 0x80000 priority 2
controller  sc0 at vme24 ? csr 0x200000 priority 2 vector scintr 0x40
disk        sd0 at sc0 drive 0 flags 0
disk        sd1 at sc0 drive 1 flags 0
tape        st0 at sc0 drive 32 flags 1
disk        sd2 at sc0 drive 8 flags 0
tape        st1 at sc0 drive 40 flags 1
#disk       sf0 at sc0 drive 8 flags 2
controller  sc1 at mbmem ? csr 0x84000 priority 2
disk        sd2 at sc1 drive 0 flags 0
disk        sd3 at sc1 drive 1 flags 0
tape        st1 at sc1 drive 32 flags 1
#disk       sf1 at sc1 drive 8 flags 2
device      sky0 at mbio ? csr 0x2000 priority 2
device      sky0 at vme16 ? csr 0x8000 priority 2 vector skyintr 0xb0
```

```
device      zs0 at obio 1 csr 0x2000 flags 3 priority 3
device      zs0 at obio 2 csr 0x7f2000 flags 3 priority 3
device      zs1 at obmem 1 csr 0x780000 flags 0x103 priority 3
device      zs1 at obio 2 csr 0x7f1800 flags 0x103 priority 3
device      zs2 at mbmem ? csr 0x80800 flags 3 priority 3
device      zs3 at mbmem ? csr 0x81000 flags 3 priority 3
device      zs4 at mbmem ? csr 0x84800 flags 3 priority 3
device      zs5 at mbmem ? csr 0x85000 flags 3 priority 3
device      mti0 at mbio ? csr 0x620 flags 0xffff priority 4
device      mti1 at mbio ? csr 0x640 flags 0xffff priority 4
device      mti2 at mbio ? csr 0x660 flags 0xffff priority 4
device      mti3 at mbio ? csr 0x680 flags 0xffff priority 4
device      mti0 at vme16 ? csr 0x620 flags 0xffff priority 4
            vector mtiintr 0x88
device      mti1 at vme16 ? csr 0x640 flags 0xffff priority 4
            vector mtiintr 0x89
device      mti2 at vme16 ? csr 0x660 flags 0xffff priority 4
            vector mtiintr 0x8a
device      mti3 at vme16 ? csr 0x680 flags 0xffff priority 4
            vector mtiintr 0x8b
device      ie0 at obio 2 csr 0x7f0800 priority 3
device      ie0 at mbmem ? csr 0x88000 priority 3
device      ie1 at mbmem ? csr 0x8c000 flags 2 priority 3
device      ie1 at vme24 ? csr 0xe88000 priority 3 vector ieintr 0x75
device      ec0 at mbmem ? csr 0xe0000 priority 3
device      ec1 at mbmem ? csr 0xe2000 priority 3
controller  tm0 at mbio ? csr 0xa0 priority 3
controller  tm0 at vme16 ? csr 0xa0 priority 3 vector tmintr 0x60
controller  tm1 at mbio ? csr 0xa2 priority 3
controller  tm1 at vme16 ? csr 0xa2 priority 3 vector tmintr 0x61
tape        mt0 at tm0 drive 0 flags 1
tape        mt1 at tm1 drive 0 flags 1
controller  xtc0 at mbio ? csr 0xee60 priority 3
controller  xtc0 at vme16 ? csr 0xee60 priority 3 vector xtintr 0x64
controller  xtc1 at mbio ? csr 0xee68 priority 3
controller  xtc1 at vme16 ? csr 0xee68 priority 3 vector xtintr 0x65
tape        xt0 at xtc0 drive 0 flags 1
tape        xt1 at xtc1 drive 0 flags 1
device      ar0 at mbio ? csr 0x200 priority 3
device      ar1 at mbio ? csr 0x208 priority 3
device      gpone0 at vme24 ? csr 0x210000
device      cgtwo0 at vme24 ? csr 0x400000
device      cgone0 at mbmem ? csr 0xec000 priority 3
device      bwtwo0 at obmem 1 csr 0x700000 priority 4
device      bwtwo0 at obio 2 csr 0x0 priority 4
device      bwone0 at mbmem ? csr 0xc0000 priority 3
device      vp0 at mbio ? csr 0x400 priority 2
device      vpc0 at mbio ? csr 0x480 priority 2
device      vpc0 at vme16 ? csr 0x480 priority 2 vector vpcintr 0x80
device      vpc1 at mbio ? csr 0x500 priority 2
device      vpc1 at vme16 ? csr 0x500 priority 2 vector vpcintr 0x81
device      pi0 at obio 1 csr 0x1800
device      des0 at obio 1 csr 0x1000
```

```
device      des0 at obio 2 csr 0x7f1000
device      tod0 at obio 1 csr 0x3800
device      tod0 at vme24 ? csr 0x200800
```

3.2. Sun-3 GENERIC Configuration File

The following is the GENERIC configuration file for a Sun-3 system.

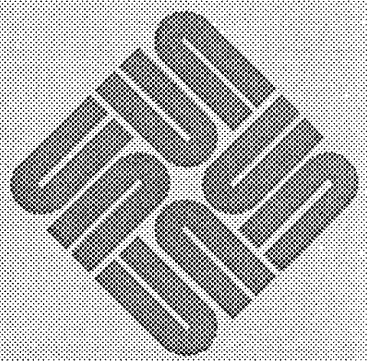
```
#
# GENERIC SUN3
#
machine     "sun3"
cpu         "SUN3_160"# (Sun-3/160 or Sun-3/75 cpu)
cpu         "SUN3_50"
cpu         "SUN3_260"# (Sun-3/280)
cpu         "SUN3_110"#
ident       GENERIC
timezone    8 dst
maxusers    4
options     INET
options     SYSACCT
options     QUOTA
options     NFS
options     NIT
options     IPCMESSAGE# SystemV IPC Message Facility
options     IPCSEMAPHORE# SystemV IPC Semaphore Facility
options     IPCSHMEM# SystemV IPC Shared-Memory Facility

config      vmunixswap generic

pseudo-devicepty
pseudo-devicebk
pseudo-deviceether
pseudo-deviceloop
pseudo-devicend
pseudo-devicewin128
pseudo-devicedtop4
pseudo-devicems3
pseudo-devicekb3

# connections for machine type 1 (SUN3_160)
controller  virtual 1 at nexus ?
controller  obmem 1 at nexus ?
controller  obio 1 at nexus ?
controller  vme16d16 1 at nexus ?
controller  vme24d16 1 at nexus ?
controller  vme32d16 1 at nexus ?
controller  vme16d32 1 at nexus ?
controller  vme24d32 1 at nexus ?
controller  vme32d32 1 at nexus ?

# connections for machine type 2 (SUN3_50)
```

```
controller  virtual 2 at nexus ?
controller  obmem 2 at nexus ?
controller  obio 2 at nexus ?

# connections for machine type 3 (SUN3_260)
controller  virtual 3 at nexus ?
controller  obmem 3 at nexus ?
controller  obio 3 at nexus ?
controller  vme16d16 3 at nexus ?
controller  vme24d16 3 at nexus ?
controller  vme32d16 3 at nexus ?
controller  vme16d32 3 at nexus ?
controller  vme24d32 3 at nexus ?
controller  vme32d32 3 at nexus ?

# connections for machine type 4 (SUN3_110)
controller  virtual 4 at nexus ?
controller  obmem 4 at nexus ?
controller  obio 4 at nexus ?
controller  vme16d16 4 at nexus ?
controller  vme24d16 4 at nexus ?
controller  vme32d16 4 at nexus ?
controller  vme16d32 4 at nexus ?
controller  vme24d32 4 at nexus ?
controller  vme32d32 4 at nexus ?

controller  xyc0 at vme16d16 ? csr 0xee40 priority 2 vector xyintr 0x48
controller  xyc1 at vme16d16 ? csr 0xee48 priority 2 vector xyintr 0x49
disk        xy0 at xyc0 drive 0
disk        xy1 at xyc0 drive 1
disk        xy2 at xyc1 drive 0
disk        xy3 at xyc1 drive 1
controller  sc0 at vme24d16 ? csr 0x200000 priority 2 vector scintr 0x40
disk        sd0 at sc0 drive 0 flags 0
disk        sd1 at sc0 drive 1 flags 0
tape        st0 at sc0 drive 32 flags 1
disk        sd2 at sc0 drive 8 flags 0
tape        st1 at sc0 drive 40 flags 1
#disk       sf0 at sc0 drive 8 flags 2
controller  si0 at vme24d16 ? csr 0x200000 priority 2 vector siintr 0x40
controller  si0 at obio ? csr 0x140000 priority 2
disk        sd0 at si0 drive 0 flags 0
disk        sd1 at si0 drive 1 flags 0
tape        st0 at si0 drive 32 flags 1
disk        sd2 at si0 drive 8 flags 0
tape        st1 at si0 drive 40 flags 1
#disk       sf0 at si0 drive 8 flags 2
device      zs0 at obio ? csr 0x20000 flags 3 priority 3
device      zs1 at obio ? csr 0x00000 flags 0x103 priority 3
device      mti0 at vme16d16 ? csr 0x620 flags 0xffff priority 4
            vector mtiintr 0x88
device      mti1 at vme16d16 ? csr 0x640 flags 0xffff priority 4
            vector mtiintr 0x89
```

```
device      mti2 at vme16d16 ? csr 0x660 flags 0xffff priority 4
            vector mtiintr 0x8a
device      mti3 at vme16d16 ? csr 0x680 flags 0xffff priority 4
            vector mtiintr 0x8b
device      ie0 at obio ? csr 0xc0000 priority 3
device      ie1 at vme24d16 ? csr 0xe88000 priority 3 vector ieintr 0x75
device      le0 at obio ? csr 0x120000 priority 3
controller  tm0 at vme16d16 ? csr 0xa0 priority 3 vector tmintr 0x60
controller  tm1 at vme16d16 ? csr 0xa2 priority 3 vector tmintr 0x61
tape        mt0 at tm0 drive 0 flags 1
tape        mt1 at tm1 drive 0 flags 1
controller  xtc0 at vme16d16 ? csr 0xee60 priority 3 vector xtintr 0x64
controller  xtc1 at vme16d16 ? csr 0xee68 priority 3 vector xtintr 0x65
tape        xt0 at xtc0 drive 0 flags 1
tape        xt1 at xtc1 drive 0 flags 1
device      gpone0 at vme24d16 ? csr 0x210000
device      cgtwo0 at vme24d16 ? csr 0x400000
device      cgfour0 at obmem 4 csr 0xff000000 priority 4
device      bwtwo0 at obmem 1 csr 0xff000000 priority 4
device      bwtwo0 at obmem 2 csr 0x100000 priority 4
device      bwtwo0 at obmem 3 csr 0xff000000 priority 4
device          bwtwo0 at obmem 4 csr 0xff000000
device      vpc0 at vme16d16 ? csr 0x480 priority 2 vector vpcintr 0x80
device      vpc1 at vme16d16 ? csr 0x500 priority 2 vector vpcintr 0x81
device      des0 at obio ? csr 0x1c0000
device      fpa0 at virtual ? csr 0xe0000000
```

4

Changes to 3.0 in Release 3.2



4

Changes to 3.0 in Release 3.2

Release 3.2 incorporates many new enhancements that update the 3.0 release.

4.1. Sun-3/110 Features

Release 3.2 contains support for the Sun-3/110. The sections that follow describe the system software that supports the Sun-3/110.

Overview

The Sun-3/110 requires 3.2 system software. Its frame buffer is divided into three *plane groups*: a 1152x900x8 color buffer, an 1152x900x1 *overlay* (monochrome) plane, and an 1152x900x1 *enable* plane. The Sun-3/110 frame buffer is also known as the *cgfour* frame buffer. The color of a pixel is determined by either the color buffer or the overlay monochrome plane in the following manner.

The color frame buffer feeds out 8-bit deep data. If the corresponding bit of the enable plane *is* set, the overlay monochrome plane determines the pixel's color (black or white). If the corresponding bit of the enable plane *is not* set, a color lookup table passes 8-bit deep intensity information (24 bits total) for the red, green, and blue components of that pixel, determining its color. You may notice that the intensity of white in the overlay monochrome plane is much stronger than that in the color buffer. This disparity may disappear in the future.

The Sun-3/110's functionality raises some questions regarding compatibility across Sun 3.*x* UNIX releases, including pixrects, pixwins, and SunView support. A general discussion is presented here. For programming specifics, you should refer to (1) the pixrect plane group interface described in the errata/addenda chapter of this document, and (2) the *SunView Enhancements* section of this document.

Networking Considerations

If you are adding a diskless Sun-3/110 to a network, remember that a server and all its diskless clients (those depending on the server for binaries) must run the same revision level of software. However, this has nothing to do with NFS server/client relationships; files can be shared between systems not running the same release of software.

Sun-3/110 Compatibility Issues

The Sun-3/110 contains a frame buffer, described in `cgfour`(4), that is supported by the pixrect, pixwin and SunView layers.

Pixrects

The cgfour frame buffer driver automatically initializes the enable plane for a pixrect application. If a pixrect application is linked with a pre-3.2 pixrect library, the enable plane is automatically set to all 1's, and the overlay monochrome plane is displayed. If a pixrect application is linked with a 3.2 pixrect library, the enable plane is automatically set to all 0's, and the color planes are displayed.

Therefore, color applications (including grayscale) built with pre-3.2 libraries and header files will not run in color on a Sun-3/110; you must recompile using the 3.2 libraries and header files. Your applications will then access the pixrect support for the 8-bit deep color buffer.

Black-and-white applications that are built with pre-3.2 libraries and header files will run on a Sun-3/110; you should be able to move these pre-3.2 binaries onto a 3.2 kernel and run them directly. If you do recompile, your application may run somewhat slower, since the auto-initialization feature will have set the application to run in the color planes. You can improve performance by making a source code change instructing your applications to use the overlay monochrome plane; this is discussed in the following. (For pixwins and SunView applications, you should make a different source code change, which is discussed under *Pixwins*.)

As mentioned earlier, for all applications, after recompiling with 3.2, the pixrect returned from `pr_open("/dev/fb")` or `pr_open("/dev/cgfour0")` accesses the color buffer. To speed up monochrome applications you may set the pixrect to run in the overlay monochrome plane instead of the color buffer with the following code:

```
#include <pixrect/pr_planegroups.h>

char groups[PIXPG_OVERLAY+1];

pr_available_plane_groups(pr, sizeof(groups), groups)

if (groups[PIXPG_OVERLAY] && groups[PIXPG_OVERLAY_ENABLE]) {
    pr_set_plane_group(pr, PIXPG_OVERLAY_ENABLE);
    pr_rop(pr, 0, 0, pr->pr_width, pr->pr_height, PIX_SET, 0, 0, 0);
    pr_set_plane_group(pr, PIXPG_OVERLAY);
}
```

If the pixrect `pr` does not have an overlay monochrome plane, this code will do nothing. In a future release, Sun may provide a function to set the pixrect to run in the overlay monochrome plane.

When you have made this source code change and then run a pixrect application, your workstation screen will flash, briefly displaying the color planes before switching to the overlay monochrome plane. This is due to the auto-initialization of the enable plane.

To summarize, the following table outlines how a pixrect application, without the previous pixrect source code change, will operate when moved onto the cgfour:

Table 4-1 *Pixrect Applications on a cgfour*

	Applications Linked with Pre-3.2 Library	Applications Linked with 3.2 Library
Black and White	Will use overlay plane	Will use color planes; make source code change to use overlay plane
Color or Grayscale	Will use overlay plane; recompile to use color planes	Will use color planes

Note to pixrect programmers: pixrects now has a new plane group interface that will support cgfour's multiple plane architecture. This interface is described in the errata/addenda chapter of this document. Specifically for the cgfour frame buffer, `pr_available_plane_groups` returns

```
5 (PIXPG_OVERLAY + 1)
```

and fills in `groups` with

```
{ 0, 0, 1, 1, 1 }
```

if it is at least five bytes long.

Pixwins

Anything you build with SunView or SunWindows uses pixwins. Pixwins differ from pixrects in that pixwins do not allow applications to have direct control of plane groups. Instead, the pixwin library utilizes the different plane groups for its own purposes. In particular, the pixwin library attempts to place monochrome windows in the overlay monochrome plane and color windows in the color buffer.

Pixwin operation on a Sun-3/110 parallels that of pixrects. Color applications (including grayscale) built with pre-3.2 libraries and header files and moved onto a Sun-3/110 will behave as if they were running on a monochrome monitor. To access the pixrect support for the 8-bit deep color buffer, you must recompile using the 3.2 libraries and header files.

Black-and-white applications that are built with pre-3.2 libraries and header files will run on a Sun-3/110; you should be able to move pre-3.2 binaries that run in black and white onto a 3.2 kernel and run them directly, but you will suffer from minor glitches that are artifacts of the multiple plane group architecture of the Sun-3/110 — for example, menus and prompts that intersect color windows will be occluded by the color windows. If you do recompile, you will not suffer from glitches, but your applications may run somewhat slower due to using the color buffer rather than the overlay monochrome plane. You can improve performance by making the source code change discussed in the following.

For all applications, when recompiling with 3.2, the pixwins associated with `canvas_pixwin(canvas)`, `gfx->gfx_pixwin`, and `pw_open(windowfd)` — canvases, graphics subwindows, or pixwins (and pixrects) you create — will access the color buffer. The pixwins associated with other SunView elements, such as frames, text subwindows, scrollbars, tty

subwindows, etc., will attempt to use the overlay monochrome plane. With no source code changes, your pixwin and SunView applications will run functionally, be visually correct, and will have no more glitches. If an application uses canvases, graphics subwindows, or pixwins/pixrects directly but does not use color, the application may run somewhat slowly due to using the color buffer. To cause the pixwin to run in the overlay monochrome plane instead of the color buffer, make the following source code changes:

- For canvases, use the CANVAS_FAST_MONO attribute to either

  ```
  window_create(frame, CANVAS, CANVAS_FAST_MONO, TRUE, ..., 0)
  ```

  or

  ```
  window_set(canvas, CANVAS_FAST_MONO, TRUE, 0)
  ```

- For graphics subwindow applications, call

  ```
  pw_use_fast_monochrome(gfx->gfx_pixwin)
  ```

- For raw pixwin applications, either call

  ```
  pw_use_fast_monochrome(pw)
  ```

  or get the original pixwin with the call

  ```
  pw = pw_open_monochrome(windowfd)
  ```

  These methods are ways of hinting to the pixwin library to put the window associated with the pixwin in the fast overlay monochrome plane, if one is available. If no such plane is available, or color is explicitly specified elsewhere, these actions do nothing.

To summarize, the following table outlines how pixwin and SunView applications, without any source code changes, will operate when moved onto the cgfour:

Table 4-2 *Pixwin and SunView Applications on a cgfour*

	Applications Linked with Pre-3.2 Library	Applications Linked with 3.2 Library
Black and White	Will use overlay plane with some glitches; recompile to get rid of glitches and make source code changes(s) to use overlay plane	Everything except canvases, graphics subwindows, and raw pixwins will use overlay plane; make source code change(s) to run everything in overlay plane
Color or Grayscale	Will behave as if on a monochrome workstation; recompile to use color planes	Will use color planes

Changes Visible to SunView Users

If you have applications that run on a monochrome frame buffer and are built with pre-3.2 libraries and header files, and you do not recompile, your applications will run on a 3.2 kernel with a few visible glitches that are artifacts of the multiple plane group architecture of the Sun-3/110. In particular, menus and prompts that intersect color windows will not be visible through the color windows. You can fix this problem by recompiling.

The previous table describing pixwin and SunView applications on a cgfour summarizes how SunView applications will behave when moved onto a cgfour. As the table notes, one part of your application can use the color planes, while another can use the overlay monochrome plane — for example, without recompiling a black-and-white application, your frame will run in the overlay monochrome plane, while a graphics subwindow inside it will run in the color planes. Of course, you can set your frame to be color and the subwindow inside it to be black and white by using the Suntools command line arguments.

If a cursor or crosshair originating in one plane group overlaps a region of the screen that belongs to another plane group, the cursor or crosshair will appear truncated.

Sun provides a new facility for creating two full desktops on the same monitor (each desktop in a separate plane group). Sun also provides an additional facility, called `switcher`, for moving between the different desktops, and for setting the enable plane to 0's or 1's if it appears to be having problems. For more information, refer to the *SunView Enhancements* section of this document.

`Overview` users are advised that, because of the existence of plane groups, pre-3.2 applications cannot successfully run under `overview` if `overview` itself is running in the color buffer. For more information, refer to the *SunView Enhancements* section of this document.

The `-b` or `-f` options for `suntools` are ignored, unless the applications are running in an `8bit_color_only` desktop. For more information, refer to the *SunView Enhancements* section of this document.

4.2. Graphics

Release 3.2 contains enhancements for the pixrect, GP microcode, *SunCGI* and *SunCore* software.

Pixrects

The following enhancements are provided by pixrects:

1) a plane group interface, described in the errata/addenda chapter of this document. (The plane group interface is important to Sun-3/110 pixrect users since it supports the cgfour multiple plane architecture of the Sun-3/110. Sun-3/110 users should refer to the previous section entitled *Sun-3/110 Features* in this chapter of the release notes.)

2) two new pixrect/pixwin routines for drawing textured or solid lines and polylines with width. These routines are described in the errata/addenda chapter of this document; the pixwin routine is also explained in the new *SunView Programmer's Guide.*

3) a new pixrect/pixwin routine for drawing multiple points. This routine is described in the errata/addendum chapter of this document; the pixwin

routine is also explained in the new *SunView Programmer's Guide.*

GP1 Microcode Extensions

1) Graphics programs which are linked with 3.2 libraries and run on a workstation with a *Graphics Processor* (*GP*) will now check whether up-to-date microcode is installed on the GP. If up-to-date microcode is *not* installed, programs which use new 3.2 microcode functionality will run, but at a slower rate because the new 3.2 microcode features will be emulated in software.

2) Software routines for handling textured vectors, polylines, fat lines, replrop, and textured polygons have been replaced by microcode routines, resulting in significant performance improvements in these areas. Both Pixrects and SunCGI have been changed to take advantage of this. Linking your application to the new SunCGI and/or Pixrect libraries is required to take advantage of these improvements.

SunCGI

1) The way *Open View Surface* function `open_vws` uses the `Cvwsurf` data structure has changed.

The `Cvwsurf` view surface structure allows an application to request a great variety of different "options" or "configurations" for view surface use. `open_vws` uses `Cvwsurf` to either overlay an existing window, or to create a new process (`view_surface` tool) with a graphics subwindow in it.

Table 2-2 on page 15 of the *SunCGI Reference Manual* should be modified to include the following:

Table 4-3 *Available* **SunCGI** *View Surfaces*

Name	*Description*
PIXWINDD	**SunView** on either a color or monochrome display.
BWPIXWINDD	**SunView** on a monochrome display.
CG4DD	Full screen on a Sun-3/110 color display.

Only some of these configurations are supported. That is, depending on certain fields, others will be ignored. The most important rule is that the view surface structure should not contain uninitialized variables; fields must contain user data or be zeroed out, as by NORMAL_VWSURF. If the `dd` field of the structure is the device driver code for a "raw" device (e.g. a Sun-3/110 CG4DD device), **SunCGI** attempts to open a device of that variety as follows:

a) If `screenname` is non-NULL, exists *and* matches the `dd` field, then open and use that device.

b) If `/dev/fb` exists and matches the `dd` field, then open and use `/dev/fb`.

c) Check "obvious" device names. (e.g., for GP1DD, try `/dev/gpone[0-9][a-d]`) The first to match the `dd` field is opened

and used.

This allows a **SunCGI** application to create a device-specific program.

The `dd` field of the `Cvwsurf` structure may be assigned (e.g., by NORMAL_VWSURF) the constant CG4DD, indicating the device driver for the "raw" Sun-3/110 device. (Behavior is as if it were a CG2).

The `dd` field can also contain PIXWINDD, BWPIXWINDD or CGPIXWINDD. The meaning of PIXWINDD is changed and BWPIXWINDD is added. BWPIXWINDD now specifies that **SunCGI** should find a window on a monochrome screen; CGPIXWINDD, on a color screen. PIXWINDD now means "find any window". (PIXWINDD formerly had the meaning BWPIXWINDD has now, but will generally work where it used to.) **SunCGI** uses the following algorithm in `open_vws` when the `dd` field is PIXWINDD, BWPIXWINDD or CGPIXWINDD:

a) if NEW_VWSURF bit of `flags` is 0, `windowname` is non-NULL, then overlay the window with that name;

b) If NEW_VWSURF bit of `flags` is 0, WINDOW_GFX environment variable exists *and* that window is not yet used (by this **SunCGI** process), then overlay WINDOW_GFX;

c) If `screenname` is non-NULL, create `view_surface_tool` on the screen with that name;

d) If `windowname` is non-NULL, create `view_surface_tool` on the same screen as `windowname;`

e) If WINDOW_GFX is non-NULL, create `view_surface_tool` on the same screen as WINDOW_GFX.

In all cases, the screen found by the above algorithm must also match the `dd` field color specification. This restriction is always met if the `dd` field is PIXWINDD, but by specifying BWPIXWINDD or CGPIXWINDD, a **SunCGI** application can refuse to run in certain environments.

2) **SunCGI** endstyle meanings have changed.

`line_endstyle` determines how the texture is imposed on a textured (non-`SOLID`) line. The enumerated type `Cendstyle` contains values that correspond to valid line endstyles.

```
typedef enum {
    NATURAL,
    POINT,
    BEST_FIT
} Cendstyle;
```

The behavior of **SunCGI** for each of these types has been modified to make the results more aesthetically pleasing, and to make the texture alignment for different line segment lengths more predictable. The following paragraphs contrast the old and new behavior of **SunCGI** endstyles. Polylines of greater than 2 points behave as if they were "bent" lines: the line texture restarts where it left off around the intermediate points. The line texture alternates between ON

sections (dashes or dots) and OFF sections (holes).

NATURAL

NATURAL texture fitting starts the texture at the beginning of the line, and continues drawing or not drawing pixels according to the texture, until it reaches the next point. If the end point falls within an ON texture section, it will be drawn; otherwise it will not be drawn. This can cause line segments to appear shorter than they really are. Prior to Release 3.2, **SunCGI** started the texture with the first ON element shortened to half its length, and drew only those ON elements which would fit in their entirety within the line length. This left a large gap at the end of a line when the line end point fell within an ON texture section.

POINT

POINT texture fitting draws the same pixels drawn by NATURAL endstyle, and in addition will always draw the pixel at the far end of the line. When the line ends in an ON texture section, the results are identical with NATURAL. POINT produces a line that does not appear shorter than it really is. Previously, POINT also drew a NATURAL texture plus the last pixel, but since NATURAL endstyle was different, this produced different results than 3.2 **SunCGI**'s POINTendstyle.

BEST_FIT

BEST_FIT texture fitting will attempt to center the texture between the two end points of a polyline with two points, thus making the texture sections at each end point the same length within one pixel. Due to the centering algorithm, this end point might fall in either an ON or OFF texture section. Therefore, the first and last pixels are always drawn, so the line reaches its end points and shows its true length. For polylines of more than two points (more than one line segment), the texture is not presently balanced over the entire polyline. Instead, the behavior is as with POINT endstyle. Previously, BEST_FIT attempted to place an ON texture section on the end point, but did not look consistent for all segment lengths.

SunCore

1) The **SunCore** library now uses the Notifier. Any programs using the **SunCore** library should use the Notifier instead of explicit `signal` calls. These changes were made for integration with **SunView**.

2) **SunCore** will work on the Sun-3/110 view surface. The name of the view surface driver is `cg4dd`.

3) An improved method for opening a view surface on an existing window has been added to **SunCore**.

4.3. Kernel Changes

cgfour frame buffer

The 3.2 kernel includes support for the "cgfour" frame buffer on the Sun-3/110.

XY450/451 disk driver rewrite

Support for overlapped seeks has been added, improving performance for systems with more than one disk per controller. Better support was also added for disk drives going offline and coming online while the system is running. The maximum supported configurations have been increased. Previously, there was a maximum of 2 xy controllers per system with a maximum of 2 disks per controller. In the 3.2 release, the maximum number of xy controllers and the maximum number of disks per controller are both user-configurable. The error messages for the xy450 driver have also been improved. They now report the partition-relative and absolute block numbers of the sector in error. Previously, the partition-relative block number of the first sector in an erroneous transfer was reported, rather than the actual sector in error.

NOTE: There is a problem with overlapped seeks not working properly on older versions of the 450. Config can be used to disable overlapped seeks in this case (See the xy *man page for details).*

SCSI disk driver

The maximum supported configurations have been increased. Previously, there was a maximum of one SCSI Host Adapter and 2 SCSI disks. In 3.2, both of these numbers are configurable. The driver now supports the Emulex MD21 ESDI controller and the associated 141Mb disks. Use of the new controller is transparent to the user.

Support for Hayes 2400 baud modem

Changes to uucp and tip now support the Hayes 2400 baud modem.

Diag revisions and enhancements

Support for the Emulex MD21 SCSI Disk Controller and two 141Mb drives (Micropolis 1355 and Toshiba MK156F) has been added. Support for the CDC 9720 SMD disk drive has also been added. The format utility has been improved to automatically read the manufacturer's defect list on all incoming SMD and ESDI drives. Support was also added to manipulate this list manually.

New kernel debugger

A standalone adb-style debugger now supports debugging the kernel, or debugging user-written device drivers. Basic capabilities include setting breakpoints, monitoring flow and data structures during execution.

Tektool enhancements

The Tektronix 4014 terminal emulator tool has been enhanced to include support of some of the new SunView capabilities (e.g., scrollbars) within the tektool window.

4.4. C Shell

Filename Completion Added to the C-Shell

A filename completion option has been added to the C-shell. If the variable *filec* is set, then typing CTRL-D at any point on a command line causes the shell to print a list of all filenames that start with the partial input word immediately preceding the CTRL-D. Typing ESC causes the shell to complete as much of the preceding word as is unambiguous.

This results in reduced typing, and makes it unnecessary to type **cd** ... and **ls** repeatedly when locating and using programs and files. See the `csh`(1) man page for more information on *filec*.

4.5. System V Compatibility Package

Users may select the System V Compatibility environment by including `/usr/5bin` in the PATH environment variable (before `/bin` and `/usr/bin`). The default Sun environment only includes those System V features that are a compatible superset of 4.xBSD (System V IPC, for example). See *System V Enhancements Overview* (800-1541) for a complete discription of the System V Compatibility environment.

- Some 4.2BSD functionality has been replaced with the System V implementation of the same functionality in cases where both were 100% compatible but the System V version is faster.

4.6. Networking

File and Record Locking

System V compatible file and record locking (`fcntl` calls) supports both local and remote files. A new signal has been added to allow recovery in the event that a remote server crashes and locks are lost. Ordinarily, locks will be reclaimed when the remote server recovers and SIGLOST will not be issued. The signal (SIGLOST) notifies the process in the event of a lost lock. The default action is to kill the process, and therefore existing System V programs need not be changed to run on the SunOS, unless they choose to recognize this failure notification.

See *Unix Interface Reference Manual* (800-1303-03) for the following man pages: *fcntl*(2), and *lockf*(3). See the *Commands Reference Manual for the Sun Workstation* (800-1295-03) for *lockd*(8c) and *statd(8c)*.

RPC protocol compiler

The new compiler, `rpcgen`, introduces a language for creating RPC servers and XDR routines without having to write any C routines. The compiler translates a user-built description of the desired network and services into the necessary C routines.

Remote Execution Service (REX)

REX is a RPC-based remote execution service that preserves the working directory and environment variables of the invoking process. Network administration must configure the `/etc/servers` file to determine if a machine will allow this service. A user on a particular workstation does not need to worry about a remote system "stealing" CPU time on that workstation without permission. Performance for remote execution for REX is usually better than using **rsh**, since REX can eliminate the shell invocation that **rsh** requires. Transparency is also better with REX than **rsh**, since **rsh** always executes in the home directory, while REX may use NFS to execute in the directory of invocation.

See the *Commands Reference Manual* for the following new REX man pages: *on*(1), *rexd*(8C), and *rex*(3R).

Name server

The Yellow Pages server uses an inter-domain name resolution protocol for names that are not in the default domain. To enable this feature, edit the `/etc/rc.local` file on the Yellow Pages server to add the **-i** option to the ypserv program. The inter-domain name service is provided by the `/usr/etc/in.named` program, as described in the *named*(8) man page, and the resolver is described in the *resolver*(3) and *resolver*(5) man pages. There is also a "trivial" name server available. It is described in the *tnamed*(8C) man page.

UNIFY

Users upgrading to 3.2 who have installed the UNIFY product should make the following changes to their UNIFY installation *after* 3.2 has been installed.

In the file `/etc/servers`, change the entry for `/usr/db/bin/dbrexd` to be `/usr/etc/rpc.rexd`, as it is in the `/etc/servers` file shipped with 3.2.

Replace the files `/usr/db/bin/dbrexd` and `/usr/db/bin/dbon` with symbolic links with the commands:

```
ln -s /usr/bin/on  /usr/db/bin/dbon
ln -s /usr/etc/rpc.rexd  /usr/db/bin/dbrexd
```

These changes will cause UNIFY to use the standard REX facilities instead of the earlier versions which were embedded in the UNIFY product.

4.7. Languages

Floating Point Accelerator Support

Programs compiled with the `-fswitch` option on Release 3.0 require relinking if an FPA is installed. To maximize FPA performance, recompile with the `-ffpa` option.

Inline Expansion Files

In-line code templates have been added to `/usr/lib`. This allows in-line replacement of major routines that previously required calls to library routines.

4.8. SunView Enhancements

Substantial performance improvements and some enhancements have been made to SunView. It is not feasible to list all changes, so only those with the greatest likely impact on customers have been included.

Backwards compatibility

The improved display locking and internal window tree enumerator described below are supported by changes to window support code in the kernel in 3.2. Hence although tools compiled under 3.2 will run on a system running 3.0, they will not have the resulting performance improvement. Tools compiled under 3.0 and 2.x will run under 3.2.

SunView Performance Improvements

Graphics performance improvements

The display locking overhead of the pixwin library (`pw_lock()`/`pw_unlock()`) has been reduced. This results in a general graphics speed up.

NOTE *This is implemented using a shared memory technique (specific to the locking code, not a general mechanism) that was not available in release 3.0.*

Raw text display speed has been improved on memory mapped devices (most displays).

Text subwindow performance improvements

Text subwindows (found in `textedit`(1), `cmdtool`(1), `mailtool`(1), etc.) can be made *retained* as an option in `defaultsedit`(1). This results in fast repaint of the window when opened or exposed, at the expense of a slight increase in memory usage. The same option for tty subwindows (`shelltool`(1) has a tty subwindow) has existed since 3.0. You should try making these window types retained and see if you like the change in interactive response on your machine.

Text subwindow editing speed and display speed have been substantially improved, by 40 – 60 % depending on the density of text.

Caret blinking takes fewer CPU cycles.

Mouse input performance improvements

It is now possible to control the "gain" of your mouse so that you can cover greater distances on the screen with smaller hand motions. This feature is settable in the *Input* section of `defaultsedit`.

Some settings put in the kernel to support early hardware (the parallel mouse on the Sun 100U and 150U) are not optimal for later machines. They can now be changed in `defaultsedit`. The default is for the performance restrictions to be in place, so if you have later hardware you should set */Input/Jitter_Filter* and */Input/Speed_Enforced* to off.

You use a new utility, `input_from_defaults`(1), to set these mouse motion controls from your personal defaults. This also sets your preferred SunView function key placement which used to be set by `setkeys`(1). One way to run this automatically is to add the following lines to your `.login` file:

```
if ('tty'==/dev/console) then
    echo 'Setting input defaults'
    input_from_defaults
endif
```

NOTE *Inadvertently running* `input_from_defaults`, *for example when you* `rlogin` *to someone else's machine, will change mouse and keyboard behavior even if someone else is using* `suntools`; *this is why the example above checks to see if you are running on the console.*

General performance improvements

By default, tools do not read the system defaults database in `/usr/lib/defaults` when starting up; they only read the user's private preferences in `~/.defaults`. This reduces tool startup time substantially. You should only need to change this if you modify the system defaults database; if so you should set *Private_only* in the *Defaults* category of `defaultsedit` to *False.*

Panel initialization and scrolling in panels are faster.

A quick window enumerator (it returns the position of all windows on the screen) has reduced the overhead of tool startup and icon placement upon closing a window.

NOTE *The quick window enumerator requires a kernel change, so the performance improvement is not available to 3.2 tools running under release 3.0.*

The performance of the folder menu in `mailtool` has been improved. It comes up rapidly if there have been no changes to the folders directory.

Merged programs

Each executable SunView program includes as much as 400 – 500 KBytes of library code from the *suntool, sunwindow* and *pixrect* libraries, so individual programs take up a lot of disk space. In 3.0 several groups of programs were made symbolic links to *`merge` programs. Each determines from the name it is called with what "program" it should run, and calls that program, which is really a subroutine. This allows several commands to share a single copy of the SunView libraries on disk and in memory.

In 3.2, this concept has been extended so that all SunView applications belong to one of two sets of tools:

in `/usr/bin/suntools`:

```
selection_svc
switcher
cmdtool
shelltool
ttytool
textedit
gfxtool
view_surface
coretool
clock
clocktool
suntools
mailtool
perfmeter
overview
perfmon
align_equals
capitalize
clear_functions
get_selection
insert_brackets
shift_lines
```

in /usr/bin/othertools:

```
fontedit
iconedit
tektool
traffic
defaultsedit
mailrc_to_defaults
defaults_to_mailrc
stty_from_defaults
defaults_from_input
input_from_defaults
canvas_demo
cursor_demo
bouncedemo
framedemo
jumpdemo
spheresdemo
lockscreen
lockscreen_default
adjacentscreens
swin
setkeys
toolplaces
scrolldefaults
defaults_to_indentpro
indentpro_to_defaults
```

The benefit of merging all the programs together is twofold. On disk, the two merged programs require greatly reduced storage compared with 46 separate programs. We have saved about 4 MBytes over 3.0 using this approach. When running, multiple ''programs'' from the same merge share resources of real memory and swap space by sharing the same text (code) segment.

The cost is that each process has a larger data segment that includes many variables used by other programs and is not shared. This takes up more swap space and can lead to greater paging due to virtual memory fragmentation. Also, if you run applications from different merges at once then two large programs contend for physical memory, leading to poor performance.

Sun has found the division of SunView applications into these two sets to be a good compromise. The `suntools` set includes what we think are the programs that are commonly used at the same time, erring toward including more programs than less. All the less commonly used programs and programs that are not used simultaneously with other programs, such as `lockscreen`, are in `othertools`.

We encourage users with better knowledge of their usage patterns to create their own merged set of programs that includes the programs they run interactively and simultaneously (Sun's and their own) and excludes the rest. The key is to only run programs from one set Specifying your own merge set is like configuring the kernel to remove unused device drivers.

How to make your own `toolmerge`

`toolmerge` is a new feature for 3.2 and subsequent releases. You need both SunView optional software programs in order to use it. The following is a brief description on how to use it:

Do the following:

- load the SunView Programmers' optional software.
- cd to `/usr/src/sun/suntool` and look at the following 3 files:

  toolmerge.c is a program which invokes one of a collection of applications; this is done by looking up the name by which it is invoked in the array cmd_routine_map, and then calling the corresponding program_main procedure.

  `basetools.h and othertools.h` are include-files which initialize cmd_routine_map in toolmerge.c, and also are used to generate symbolic links to the merge for each application.

  Because of this second use, the format of the files is critical: each line should contain the name by which the program will be invoked (enclosed in double quotes), followed by comma, space, the name of the program_main procedure to invoke (no quotes), another comma, and newline. Example: *<start-of-line>*`footool, foo_main,`*<new-line>*

  3) edit the two .h files until they describe the distribution you want: move lines from one file to the other to change the category of a particular application; (Be careful to get *all* references to a single main procedure!) remove all references to applications you want to eliminate; add lines to the appropriate include file to recognize and invoke applications you want to add to the toolmerge.

  4) make `basetools` and `othertools`

  When `toolmerge.c` is compiled, the `cmd_routine_map` is initialized from a file named by a constant passed in from the makefile. The Make variables used for this are `BASE_FILES` and `OTHER_FILES`; and their default values are `basetools.h` and `othertools.h` respectively.

  These definition may be overridden explicitly on the make line, thus: make **`basetools BASE_FILES=`***mybase.h*

  If new programs are to be added to either merge, their .o files should be named in the Make variable MOREOBJS, thus:

  `make basetools MOREOBJS="a.o b.o c.o"`
  `BASE_FILES=`*mybase.h*

5) make `install_bins`

This will generate links to the appropriate merge file for each program name in the two .h files. Example: if `basetools.h` includes the line

footool, **foo_main,**

then the following symbolic link will be constructed:

```
ln -s /usr/bin/suntools DIR/footool
```

where DIR is /usr/bin unless the substring "demo" occurs in the name of the program (e.g. "foodemo"); in that case, the link is placed in /usr/demo.

Sun-3/110 Support in SunView

The changes made to the pixrect and pixwin layers to provide Sun-3/110 support are documented in the *Sun-3/110* section above. The following subsections covers enhancements to SunView, and some suggested code changes to allow applications to take advantage of Sun-3/110 features. It is only relevant to Sun-3/110 users and programmers wishing to write Sun-3/110-compatible applications.

Smaller pixels ⇒ larger font

If you are using a Sun-3/110 with the 15 inch color monitor, you may want to increase the default font size that tools use to compensate for the smaller pixels. You can set */SunView/Font* in `defaultsedit` to a larger font to make a global change, or you can specify the `-Wt` *fontname* command line option to set the font in individual tools.

Running two desktops at once

The overlay plane and the color buffer are two separate plane groups on Sun-3/110. Changes to `suntools` in 3.2 allow you to run `suntools` separately on each plane group, so that two desktops can be maintained at once. The enable plane determine which desktop is visible. Here's how you do it:

a) Create a desktop that resides in the color buffer only. >From the console type:

```
suntools -8bit_color_only -toggle_enable
```

This creates a desktop on the color buffer that will not use the overlay plane.

b) Create a desktop that resides in the overlay plane only. >From a shelltool, cmdtool or the root menu execute:

```
suntools -d /dev/bwtwo0 -toggle_enable -n
```

This creates a second desktop running in the overlay plane; to the kernel the overlay plane looks identical to the Sun-2 B&W framebuffer. The `-n` flag bypasses reading in the `~/.suntools` or `/usr/lib/.suntools` startup files, thereby avoiding duplicate tools and consoles in the two desktops; you can use the `-s` *other_startup_file* flag instead to start up a different set of tools on the second desktop.

c) Run `adjacentscreens` (1) to set things up so that you can slide the cursor from one desktop to another. >From a `shelltool, cmdtool` or the root menu execute something like:

```
adjacentscreens -c /dev/fb -l /dev/bwtwo0
```

This tells the kernel that the overlay plane is a second "screen" to the left of the color buffer. When you move the cursor off the edge of one screen, it reappears on the other: the screen flips from one plane group to the other and the other desktop appears.

The effect is to give you rapid access to two different desktops, and thus more screen real estate, without the expense of hiding, exposing, closing and opening tools. Typically, you would run your color applications in the color buffer and ordinary SunView tools in the black and white overlay plane.

NOTE *If you are going to use this facility:*

- Do not run old pre-3.2 applications run on the `8bit_color_only` desktop; they will not appear because they will be writing to the overlay plane.
- Be careful to not have more than one shelltool or cmdtool acting as the console at once. Kill one console before starting another, or you will see weird effects on the screen.

`switcher`

There is a new application in `/usr/bin` called `switcher`. If you start it with the appropriate arguments then it can be used as an alternative to `adjacentscreens` for switching between the two desktops created on your Sun-3/110 by the above method. Clicking the `switcher` icon gets you to the other desktop using some amusing video wipe animation.

You can also use `switcher` just to set the enable plane to 0 or 1 if the enable plane gets out of sync so that you are not seeing the correct plane group. See the man page `switcher` (1) for details.

Plane groups and `overview`

You can't successfully run pre-3.2 applications under `overview` if `overview` itself is running in the color buffer. If `overview` is not in the overlay plane then the enable plane isn't be properly set up to see the application running under `overview`. This means that you can't run `overview` with the `-Wf` or `-Wb` generic tool command line arguments to set its colors. Also, you can't run `overview` in an `8bit_color_only` desktop.

Foreground and background in `suntools`

The `-b` or `-f` options to `suntools` are for the most part ignored unless you run it in an `8bit_color_only` desktop. This is because the overlay plane can *only* display black and white, and `suntools` attempts to run in the overlay plane to speed its operation and the tools run under it.

Using the overlay plane programmatically

Although the default plane group when you open a pixrect or pixwin is the color buffer, frames, panels, tty subwindows and text subwindows will all use the overlay plane if possible because the code for them hints to use the overlay plane.

Canvases and gfx subwindows in applications compiled under 3.2 will access the color buffer, on the assumption that you are likely to want to draw in them in color. This is fine, but will slow down your application if it is only drawing in black and white. When you recompile you can make minor code changes that cause the subwindow to display in the overlay plane:

- For canvases, use the `CANVAS_FAST_MONO` attribute in either

```
window_create(frame, CANVAS, CANVAS_FAST_MONO, TRUE, ..., 0)
```

or

```
window_set(canvas, CANVAS_FAST_MONO,TRUE, 0)
```

- For applications that use the old SunWindows gfx subwindow type, use the call `pw_use_fast_monochrome(gfx->gfx_pixwin)`.

These are just hints to the window system to use the overlay plane; the user can overrule them and use color by specifying the generic window command line arguments `-Wf`, `-Wb` and `-Wg`.

New routines for caching screen pixels

If you have written your own menu or prompt package that does its own caching of screen bits, you should instead use the new pixel caching utilities available in 3.2, `pw_save_pixels()` and `pw_restore_pixels`. Doing your own pixel caching can result in screen image damage on a Sun-3/110 in the plane group that you didn't cache. Programs that use the SunView menu and prompt facilities are not affected by this problem.

Extensions Visible to the User

Scrollbar extensions

A continuous repeat facility has been added to the scrollbar user interface; after a delay determined by */Scrollbar/Repeat_time* in `defaultsedit`, the scrolling action is repeated as long as you hold down the mouse button.

Thumbing (middle button) in the top or bottom *N* pixels of the scrollbar moves the scroll to the very beginning or end of the object. The value of *N* is */Scrollbar/End_point_area* in `defaultsedit`.

The thumbing bar and scroll boxes parts of a scrollbar have different cursors to indicate that the mouse buttons function differently in them.

Text extensions

A checkpoint facility has been added to `textedit` and `cmdtool`; it serves as an optional crash recovery mechanism. If enabled it periodically writes out a backup copy of the file you are editing to *filename*`%%`. The creation of this backup file is a separate process from the *filename*`%` file that `textedit` creates when you save a file.

The entire file is written to *filename*`%%` after every *N* edits to the file, where *N* is determined by the user; each character typed, each Get and each Delete operation counts as an edit. Thus to recover from a crash you need only copy *filename*`%%`, losing at most *N* edits.

Checkpointing is enabled by setting *Checkpoint_frequency* (*N*) to a non-zero value in the *Text* and *Tty* categories of `defaultsedit`. It can also be enabled by the command line option `-Ec` *N* of `textedit` and `-P` *N* of `cmdtool`. By default *N* is 0 so checkpointing is disabled.

The tool is "dead" while it checkpoints the file, so values of *N* below 200 or so are probably a bad idea.

There is another new defaults option for all text subwindows, */Text/Store_changes_file*. By default it is *True*, which means that 'Store' in the text subwindow works as it always has: after storing to the filename selected, it loads that file. If set to *False*, then 'Store' does not load the file that was stored to, keeping loaded the file you started editing.

New defaults

`defaultsedit` can be used to set additional default parameters; many have been mentioned already.

There is a new category, *Input*, which lets you control the responsiveness of your mouse (see *General Performance Improvements* above) and keyboard function key assignments (previously set by the `setkeys` command). The *Left_Handed* default both moves the SunView function keys to the right hand pad and swaps the left and right mouse buttons.

The *Input* category, together with `input_from_defaults` (described above), which sets preferences in the kernel, make `setkeys` somewhat obsolete unless you want to swap the keyboard around without affecting the mouse.

Other extensions

`iconedit` (1) has a browsing facility. A pop-up frame displays icons, cursors and glyphs in the current directory, which you can set.

You can specify whether frame labels are to be emboldened in `defaultsedit`.

You can specify either a pattern or an image to serve as the background image in `suntools` (1) instead of the "root gray" pattern in `defaultsedit` or by the `-pattern` or `-background` options to `suntools`.

Extensions Visible to the Programmer

Revised manuals

The *SunView Programmer's Guide* and *SunView System Programmer's Guide* have been revised and reprinted for release 3.2. The former in particular has been significantly improved:

- The explanation of the SunView model and programmatic interface is better.
- All functions and attributes are listed in a new summary chapter.
- There are many new examples throughout, and several long ones in a new appendix of examples.
- There is a new appendix on converting SunWindows-based code to SunView.
- There is a new appendix on SunView user interface guidelines.
- There are many more screendumps throughout the text.

Extensions to attributes

There is a new frame attribute, `FRAME_CMDLINE_HELP_PROC` to aid in writing custom "Usage: ..." messages in programs; this in turn can call the procedure `frame_cmdline_help()` which prints out the generic switches. This provides the same functionality as the old `tool_usage()` procedure.

You can specify that the frame label is to be emboldened with the attribute `FRAME_EMBOLDEN_LABEL` (or via `defaultsedit`; see above).

There is a new tty attribute, `TTY_BOLDSTYLE_NAME`, corresponding to the */Tty/Bold_style* preference in `defaultsedit`. The possible styles for bold characters are listed in `<suntool/ttysw.h>`.

It is possible to create reusable attribute lists with the new `attr_create_list()` routine. This generalizes the `panel_make_list()` call from 3.0.

Three new event descriptors, `WIN_LEFT_KEYS`, `WIN_TOP_KEYS` and `WIN_RIGHT_KEYS`, allow you to enable these groups of function keys as a whole.

There are several new text subwindow attributes:

```
TEXTSW_AGAIN_RECORDING
TEXTSW_CHECKPOINT_FREQUENCY
TEXTSW_CONTROL_CHARS_USE_FONT
TEXTSW_EDIT_COUNT
TEXTSW_IGNORE_LIMIT
TEXTSW_INSERT_MAKES_VISIBLE
TEXTSW_MEMORY_MAXIMUM
```

New pixwin calls:

Line drawing

There are new pixwin calls for drawing plain or textured lines and polylines with a "brush" of a specified width:

```
pw_line(pw, x0, y0, x1, y1, brush, tex, op)
pw_polyline(dpw, dx, dy, npts, ptlist, mvlist, brush, tex, o
```

These correspond to the pixrect level calls `pr_line()` and `pr_polyline()`, documented in the *errata to the Pixrect Reference Manual.*

Multiple points

There is a new routine to draw many pixels in a single call:

```
pw_polypoint(pw, dx, dy, npts, ptlist, op)
```

This is similar to the `pw_put()` routine, except that it fills in `npts` pixels in a single call.

Fullscreen access pixel caching and drawing routines

There are new routines to save and restore pixels that are compatible with the Sun-3/110 — see *New routines for caching screen pixels* in the *Sun-3/110 Support in SunView* section above. Typically, you would only need to use these routines if you are writing applications that implement their own menus or prompts instead of using the Sun supplied library packages. These applications need to save the image underneath the menu or prompt before displaying it, and the new

routines ensure that the pixels in the right plane group are cached. They are:

```
pw_save_pixels(pw, r);
pw_restore_pixels(pw, pc);
```

There are routines for drawing during fullscreen access that accommodate the multiple plane group frame buffer of the Sun-3/110:

```
fullscreen_pw_vector(pw, x0, y0, x1, y1, \
                     op, cms_index);
fullscreen_pw_write(pw, xw, yw, width, height, \
                    op, pr, xr, yr);
fullscreen_pw_copy(pw, xw, yw, width, height, \
                   op, pw_src, xr, yr);
```

These routines are documented in the *Menus & Prompts* chapter of the *SunView System Programmer's Guide*, (other programming topics in these change notes are mostly in the new *SunView Programmer's Guide*).

Unencoded input

A new option for the keyboard ioctl KIOCTRANS, TR_UNTRANS_EVENT, has been added to support unencoded input in the window system for the few applications that demand it. See the *Workstations* chapter in the *SunView System Programmer's Guide.*

Other Extensions

Two new `typedefs` were added (the header file `<pixrect/pixrect_hs.h>` includes them):

```
typedef struct pixrect Pixrect  in <pixrect/pixrect.h>
typedef struct pixfont Pixfont  in <pixrect/pixfont.h>
```

These are consistent with other SunView `typedefs` — `Pixwin`, `Rect`, `Event`, etc. Note that if you have defined these `typedefs` yourself in your code you will have to remove them.

4.9. New Fonts

There are now several new fonts available in 3.2. For a complete listing see `/usr/lib/fonts/fixedwidthfonts/README`.

4.10. Miscellaneous Changes

touch

The `touch` command formerly exited with an exit status of 0. In 3.2, it exits with an exit status giving the number of files it was not able to "touch". This may break some Makefiles because they may do a "touch" on a file that everybody who uses the Makefile can touch (e.g. they do not have "write" permission on the file. Change the Makefile so the exit status of the `touch` command is ignored. The - character can be used for this.

reading directories

In all releases before 3.0, programs could open a directory and read the contents with *read*(2) system call. In 3.0 and beyond, directories must be read with either *getdirentries*(2) or *directory*(3) library.

mount(8)

mount(8) now has additional options for mounting remote file systems. See the man page for more details.

timezone changes

Some new Daylight Savings time types have been added. They are specified as numbers. The "timezone" line of the config file is:

```
timezone <offset> [ dst <type> ]
```

<offset> is the offset of the current time zone, in hours. It can be positive or negative; if negative, it represents a time east of the Greenwich meridian (i.e., a negative offset). It can be specified as a floating point number, for fractional offsets; e.g., a time zone offset of minus 9 and a half hours from GMT would be specified as

```
timezone -9.5
```

with the appropriate "dst" value. If "dst" is not specified, Daylight Savings Time is not in effect. If it is specified, and no <type> is specified, it is assumed to obey the standard.

This stuff is described, although in slightly less detail, in *Installing UNIX on the Sun Workstation* in chapter 7, "Configuring the System Kernel", under "General System Description Lines". It lists a set of values for the time zone correction algorithm. The ones we added are:

6 (Canadian - same as the US, but without the changes in 1974 and 1975 and without the changes starting in 1987), 7 (Great Britain and Eire - from the last Sunday in March to the last Sunday in October), 9 (Turkish - same as Continental Europe, except that it turns on at 1:00 Standard time and off at 1:00 Daylight Savings time), and 10 (Australian "alternate" - see below). The US rules have been modified; the corrections for 1974 and 1975 have been corrected (1974 - runs from January 6 to the last Sunday in October, 1985 - Runs from the last Sunday in February to the last Sunday in October), and starting in 1987, it runs from the first Sunday in April to the last Sunday in October.

The Australian rules have been corrected also. They currently have it running from the last Sunday in October to the first Sunday in March (really - they're in the Southern Hemisphere), except in 1970 when it was not in effect at all, 1971 when it did not run in the beginning of the year, and 1972 when it ran from January 1 to February 27 in the beginning of the year.

The "Australian alternate" rules shift so that it runs until the third Sunday in March in the beginning of the year, starting in 1986, and still ends on the last Sunday in October. It is not known which states have this shift.

The rules for Continental Europe have been corrected; it always runs from the last Sunday in March to the last Sunday in September.

The starting and ending times have also been corrected. In Great Britain, Eire, and Western Europe (WET), it starts at 1:00 Standard time and turns off at 1:00 Standard time (i.e., 2:00 DST). In Central Europe, it starts at 2:00 Standard time and turns off at 2:00 Standard time (i.e., 3:00 DST). In Eastern Europe, it starts at 3:00 Standard time and turns off at 3:00 Standard time (i.e., 4:00 DST).

If the time zone offset is specified 0, then the *names* of the time zones will be GMT and BST (British Summer Time), rather than WET and WET DST (Western European Time). Portugal is the only other country with this offset.

In Australia, it starts at 2:00 Standard time and turns off at 2:00 Standard time (i.e., 3:00 DST).

It's still the same in the US and in Canada.

One warning: if any of the new times are selected, software built on a 3.1 or earlier system will not understand it; it will not have the new rules built in, and will think that Daylight Savings Time is not in effect. This will probably break few programs, but people should be aware that it may happen.

4.11. Sun-3/260 & Sun-3/280 Enhancements

This release also supports the Sun-3/260 and the Sun-3/280 systems. These are MC68020 based machines that are available with one of two monitors: a new high-resolution monochrome monitor and a standard 19" color monitor.

Applications

Black and white applications running on the new high-resolution monitor MUST be recomplied to scale correctly on the screen.

Color applications NEED NOT be recompiled, since they run unchanged on the standard 19" color monitor.

General Changes

If you are using a Sun-3/200 with the high resolution monochrome monitor, you may want to increase the default font size that tools use to compensate for its smaller pixels. You can set */SunView/Font* in `defaultsedit` to a larger font to make a global change, or you can specify the `-Wt` *fontname* command line option to set the font in individual tools.

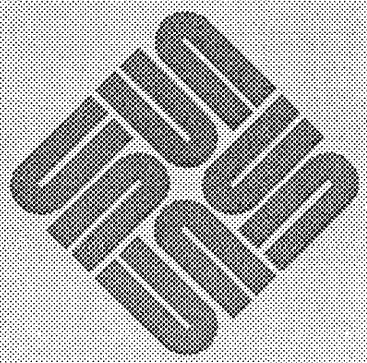
Similarly, you may want to make changes in your `~/.suntools` file to make use of the greater amount of screen real estate available.

A server and its diskless clients must run the same release software. All server/client clusters must be updated to this release.

Additional Adobe fonts have been added to the Font library and are standard with this release when installed through *Setup*. Because of the new high resolution monitor, there is the possibility that the default font will be smaller when using this release. The user may want to rescale the suntools file or pick a larger font.

5

Bug Fixes since Release 3.0



5

Bug Fixes since Release 3.0

5.1. Compiler

For all the compilers, the software command line options have been changed. In all previous versions you could concatanate separate options (e.g. -a, -b and -c: -abc) In 3.2, you must specify each option separately (-a, -b, or -c). Now we have multiple character options which make it impossible to parse the command line. This affects old compiler makefiles and shell scripts.

FORTRAN

An incorrectly initialized pointer in the static data area caused program bus errors on the RETURN statement in a module with multiple entry points. This has been fixed.

f77 now returns an error status when there is a loader error.

A data statement after the first executable statement that initialized a variable which was never referenced caused the compiler to abort with the message: `do_bss:initialized non-static`. This is fixed.

A DO loop with a floating point index variable and integer constant loop limits caused the compiler to abort with the message:`Compiler error: conssgn(nonconstant)`. This is fixed.

The compiler no longer computes the wrong constant offset for a reference of the form `STRING(1) (1:)` when `STRING` is a subroutine argument which is declared as `CHARACTER*(*)`.

The compiler no longer produces the error message "wrong number of subscripts" for statement function references if a dummy argument of a statement function has the same name as an array.

Character string constants or concatenation of character constants longer than 242 characters no longer causes the compiler to abort with a core dump.

The code generated for subscript checking when the -C option is specified previously used a different temporary for each subscript check. This caused a program to use an excessive amount of stack space or in some cases exceed the limit of 64K bytes of stack space per module. The compiler has been changed to use the same temporary variable for all subscript checks.

C compiler

printf used to round real numbers between 0.5 and 1.0 to 0. This has now been fixed.

The library routine `execat` used an incorrect constant for the maximum length of a pathname. The code has now been fixed.

A bug in the optimizer `/lib/c2` occasionally caused register load instructions to be deleted from program loops. This has been fixed. Only programs compiled with -O are affected.

Comparisons involving variables of small integral type and constants outside the legal range for that type now draw a warning. The code generated for an out-of-range comparison will be changed in release 4.0.

5.2. Graphics

Release 3.2 contains bug fixes for the *SunCGI* software.

SunCGI

1) `disjoint_polyline` used to draw 1 line too few.

2) When exiting **SunCGI** with multiple workstations, in particular when using *View Surface Tools*, **SunCGI** occasionally would kill its process group. The message `Killed` is no longer printed in the controlling window (`/dev/tty`) when a **SunCGI** program exits.

3) **SunCGI** always uses the "global drawing mode" (Raster Operation), so any primitive may be XOR-ed onto the screen.

4) Width=1 rectangle perimeters match a polyline drawn between the corners of the rectangle.

5) `open_vws` uses the "windowname" field of the `Cvwsurf` structure, overriding the value of the WINDOW_GFX environment variable.

6) Textured (non-SOLID) lines are optimized. The results are slightly more pleasing visually, and considerably faster, especially if on a screen using a *Graphics Processor*. This change affects `polyline`, `disjoint_polyline`, and the perimeters (if perimeter visibility is ON) of rectangles and polygons. Section 4.2.3 contrasts the old and new behavior of **SunCGI** endstyles.

7) Textured polygons (interior style PATTERN or HATCH) are optimized. The results are identical visually, and considerably faster for some polygons, especially on a screen using a *Graphics Processor*.

8) Coordinate handling in cgipw mode has been optimized, so certain `cgipw` output primitives should be faster, at least for non-*Graphics Processor* devices. These are `cgipw_polyline`, `cgipw_disjoint_polyline` and `cgipw_polygon`.

9) Previous releases of `libcgi77.a` would fail to link several internal routines.

10) As documented for *Clip Indicator*, the default clipping state is now CLIP.

Numerous bugs that affect the output of text characters have been fixed. Most combinations of text path and alignment did not work correctly in previous

versions, producing text that was misaligned, and returning text extent boxes from `inquire_text_extent` that did not match the text. Specific bugs that have been fixed are:

11) The text extent and concatenation point for fixed-width fonts was incorrect.

12) Text extent box did not include space for descenders unless characters with descenders were included in the string.

13) Text extent box never included parts of characters that rise above the cap line.

14) Vertical alignment did not distinguish between cap line and top line, or between base line and bottom line.

15) When the character base vector and character up vector were not perpendicular (skewed text) the text extent box did not enclose the text.

16) The extent parallelogram for text strings always included the space after the final character.

17) The character height was formerly the distance between the base line and the top line, instead of that between the base line and cap line.

18) Clipping in CHARACTER precision used only one corner of the character box to determine if the character was within the clipping window. CHARACTER and STROKE precision both clip correctly. In this release, both clip as in STROKE precision. Other optimizations should make both clipping styles faster.

19) The text append point for strings that extended past an edge of the window could wrap around if their length was as small as two window widths (or heights). This should no longer happen in normal cases.

20) The character extents for different fonts were treated as if they were identical, even thought the character definitions had different top lines and bottom lines. This caused some fonts to exceed the text extent parallelogram even for RIGHT path and (NRMAL, NORMAL) alignment (the default case).

21) STRING precision text was always aligned (LEFT, BASE) regardless of attributes the user set. STRING precision should now work correctly for all alignments, with the following caveats:

    a) Only RIGHT path is supported, as the standard permits.

    b) Cap line and top line are the same, since STRING precision uses the raster fonts, where no character is higher than a cap.

    c) Character height, orientation, expansion factor, and spacing are ignored, as the standard permits.

5.3. Kernel

driver

The following tty delays were too long: cr1, cr2, and n12. According to the *tty*(4) man page, cr1 should be about .08 seconds; cr2 about .16 seconds; and n12 about .10 seconds. In actuality, they were about 2 to 3 seconds. The algorithms in tty.c have been fixed.

Other

An mbuf chain released by `sbdrop` in the source file `sys/uipc_socket2.c` can no longer cause a `panic:sbflush2`.

The `/dev/MAKEDEV` script has been modified so you can position the tape using the QIC-24 device.

5.4. Utilities

SCCS

SCCS no longer trashes files when a file system runs out of space during an `admin` or a `delta` command.

`dbx` and `dbxtool`

Here is a list of new features in release 3.2 related to `dbx` and `dbxtool`:

1. There is a new command called `make`, which invokes the `make` program with the name of the program being debugged as an argument. It also gives the value of the `dbxenv makeargs` variable as an argument to the `make` program.
2. There is a new `dbxenv` variable called `makeargs`. The value of this variable is passed to the `make` program when the `make` command is executed. The default value is `CC=cc -g`.
3. The default setting of `file` and `func` has changed. Now, whenever the program being debugged stops, `dbx` sets the value of `func` by walking the active call stack and finding the lowest-level routine that was compiled with the `-g` option. The value of `file` is then set to the file that contains `func`.
4. In `dbxtool` there is now a menu of `dbx` commands available in the buttons subwindow. Pressing the menu button (right button) anywhere in this subwindow brings up the menu. The commands in the menu have the same behavior as the buttons; that is, they have a selection interpretation and use the command name, the interpretation, and the current selection to construct a command. The default set of menu items is:

```
menu expand display
menu expand undisplay
menu expand file
menu expand func
menu ignore status
menu lineno "cont at"
menu ignore make
menu ignore kill
menu expand list
menu ignore help
```

5. There is a new command called `menu`, which is analogous to the `button` command. Its syntax is as follows:

   `menu` *selection command-name*

   It allows a user to add items to the menu.

6. There is a new command called `unmenu`. This command is analogous to the `unbutton` command. Its syntax is as follows:

   `unmenu` *command-name*

   It allows a user to remove items from the menu.

7. The default set of buttons has changed for `dbxtool`; it is now:

```
button expand print
button expand "print *"
button ignore next
button ignore step
button lineno "stop at"
button ignore cont
button expand "stop in"
button lineno clear
button ignore where
button ignore up
button ignore down
button ignore run
```

8. In `dbxtool`, the meaning of the hollow arrow has changed, and a solid arrow has been introduced. The solid arrow is the "here I am" arrow. It points to the next statement to be executed. If the current stopping point was not compiled with `-g` and `dbx` finds a calling routine that was, a hollow arrow will be shown next to the call. In a similar manner, the hollow arrow is used to show the call sites for the `up` and `down` commands.

9. The debugging manual for release 3.0 mentions an `attach` command for `dbx`. No such command exists. The way to attach a process for debugging is with the `debug` command.

10. When referring to a non-unique name that is neither global nor local to the current procedure, you must use an unambiguous qualifier. The old way to constitute the qualifier was:

[*filename* .] *procedure-name* . *variable*

The new way is:

[`*filename*`] *procedure-name* `*variable*

A backquote is used instead of a period, and a backquote is also required in front of the filename. The period is now used only for referencing a field of a structure (or record).†

Programs

The curses input functions were going into RAW mode rather than CBREAK mode. This has been fixed; they now go into CBREAK mode as documented.

NOTE *Programs using "curses" which were compiled under pre-3.2 releases will run under 3.2 without change, but cannot be relinked under 3.2 without recompiling.*

Executing `touch -c` on a file that does not exist now returns a non-zero exit status.

The status code returned by `fsck -n` now indicates an error if there is one.

The *leave* command used to accept input with incorrect syntax, and then set the alarm to the wrong time. This has been fixed.

Using *man* and *more* on a Wyse terminal, italics get turned into underlines. If an underlined word were at the end of a line, the underline would continue to the edge of the screen. This has been fixed.

When **rm** was used on an existing file in a directory which was not readable or writable, a "`nonexistent`" file message was returned rather than `access denied`. This has been fixed.

The `spell` command was not finding all the mispelled words when it was run on a file. The problem was in `/usr/lib/spell`. The dictionary has been improved so it finds more mispelled words now.

Mail

Secret mail now properly decodes messages. There was a bug in the **mp** library which improperly handled an intersection in the domain and range of many functions.

sendmail

The file `sendmail.main.cf` did not work in 3.0. It now correctly uses the host name map (or the file `/etc/hosts` if not running the Yellow Pages) instead of the file `/usr/lib/mailhosts`.

`sendmail.cf` can now use the same domain name as set by the kernel. This should cut down the customization needed for `sendmail.cf` on each machine. The top-level domain default is now ".com" instead of ".uucp".

† When debugging Pascal programs, list *procedure-names* separated by backquotes to indicate nesting.

In some situations `sendmail` would silently discard messages if aliases formed a loop without any host names appearing. `sendmail` now issues an error message in these circumstances.

Load limits now default to zero which prevents `sendmail` from hanging when a kernel other than `/vmunix` is booted.

The `-ee` mode which `/bin/rmail` uses for incoming `uucp` mail now no longer sends extra copies of returned messages.

Underscore characters are now allowed in user names.

When a mailer dies from a signal, the error message now gives the symbolic name of the signal instead of the numeric value. For example, the "Kill" signal, number 9, usually means that swap space has run out.

`sendmail` with either the –bv `option or the SMTP VRFY command no longer causes address verification to send out a null message if it is interrupted before the verify operation completes.`

routed

The routing daemon `/etc/in.routed` previously read the interface information from the `/vmunix` and `/dev/kmem` files, which meant that routing would not work if a kernel other than `/vmunix` was booted. Routed was changed in 3.2 to use socket `ioctl` calls instead of looking at the kernel. Routing now works no matter what kernel is booted.

rlogin

The new rlogin program correctly propagates the terminal or window size to the remote system if the local and remote systems are both running 3.2 or newer. There was a problem with hanging connections (when exiting certain programs such as emacs) that has been fixed in this release.

5.5. Shell

Bourne Shell

The Bourne shell will ignore evironment variables with "funny" names, such as "A-B".

`csh` now prints messages like "Stopped", "Quit", "Segmentation Fault", and others as it should.

`csh` **kill** command now accepts all of the ASCII signal names (as per `/bin/kill`).

5.6. General Bug Fixes

Laserwriter

There is a new filter, **rasfilter8to1**, which converts an 8 bit rasterfile into a 1-bit rasterfile. See the man page *rasfilter8to1*(1) for complete details.

`tftpboot`

Diskless Sun-3 workstations use the `tftp` protocol to bootstrap. For some users, this may be a problem because the default `tftp` server allows access to any fil that is publicly readable. To increase security, remove the entry for `tftp` in the `/etc/servers` file on diskless clients, and change `/etc/servers` on the file server to be:

```
#tftp udp /usr/etc/in.tftpd
#tftp udp /usr/etc/in.tftpbootd
```

By default, the second line is commented out, which allows access to all publicly-readable files. To make the change to limit `tftp` to booting only, just move the comment character to the first line as indicated above, and kill and restart `/etc/inetd` (or reboot) for this to take effect.

5.7. SunView Bug Fixes

Release 3.0 was the first release of the new SunView window interface and toolkit. Many bugs in it have been fixed for 3.2. It is not feasible to list all the minor or internal bug fixes here, so those with the greatest likely impact on customers have been detailed.

Bug Fixes Visible to the User

Bug fixes in the text subwindow package and `textedit`

Scratch text files created in `/tmp` have read-write permissions for their owner only. This plugs a potential security leak.

You can strike a key to cancel confirmation prompts from `textedit`.

Pressing [Again] to repeat the invocation of a filter no longer dumps core.

When you insert a large file into a text window, it no longer scrolls all the way through line by line.

Numerous other bugs in `textedit` have been fixed.

Panel subwindow fixes

Panels with text items replace the caret with a grey diamond when they don't have the keyboard focus; this is consistent with the text subwindow.

Text now always goes to the item with the caret when you use [Get] in a panel.

Tool command line argument changes

Using the `-Wt` *fontname* command line option to specify the font for a tool now works for all tools.

In 3.0 the window width for `cmdtool` specified by the `-Ww` *columns* command line option included the scrollbar in the width calculation. Now the actual text display area matches the requested width.

Note: You may wish to change your `.suntools` file to adjust for these changes.

The default width of `cmdtool`, `textedit`, and `mailtool` is now 80 characters.

You can specify `-1` as the value for position and size attributes on the command line and an appropriate value will be used; this is for compatibility with old format `.suntools` files.

Subwindow resizing

In 2.0 SunWindows, when you resized tools the subwindows would be re-laid out within the tool border. In 3.0 SunView, subwindows were not re-laid out when windows were shrunk. This is fixed.

Other bug fixes

Hidden areas of windows are redrawn in English reading order when the tool is exposed.

In addition to internal bug fixes, `mailtool`'s folder menu is more square and in better order, its menu accelerators are clearer and its icon and name stripe more accurately reflect when there is *New Mail.*

Bug Fixes Visible to the Programmer

Fixes to attributes

`FRAME_LABEL` operates independently of `FRAME_SHOW_LABEL`; also the subwindow changes position accordingly upon setting and unsetting `FRAME_SHOW_LABEL`.

The `WIN_ROW_GAP` and `WIN_COL_GAP` for frames now default to 0, which makes it possible, e.g. to get the frame the same size as a 34x80 `shelltool`. They defaulted to `5` before, so this may affect the size of your tools.

You can now set the `WIN_EVENT_PROC` of ttysw's and textsw's. Before the attribute was only used by canvases and panels.

The default top margin of panels in tools written using the 2.0 SunWindows routines is set to 4 pixels in 3.2 when they are recompiled; this is as it was in 2.x. In 3.0 the top margin changed to 0 pixels when such old-style tools were recompiled. You may have to alter panel layout code to cope with the change.

`TTY_SAVE_PARAMETERS` has been de-implemented in 3.2 as the tty subwindow package no longer does anything in response to it. In 3.0 it was incorrectly documented as taking a boolean value. See the *TTY Subwindows* chapter in the *SunView Programmer's Guide* for more information.

Limitations in `window_loop()` fixed

In 3.0 there were limitations on what you could do when using `window_loop()` to put up a "confirmer" subframe. If the subwindow in the subframe put up a menu, the application was liable to get

```
Window data lock broken because pid nnn blocked
```

messages. If the subwindow did extensive computation, say in a panel item's notify proc, it might get

```
Window data lock broken after time limit exceeded by pid nnn
The offending process was sent SIGXCPU
```

messages, and be killed. Both problems have been fixed by using `win_grabio()` for the duration of `window_loop()` instead of holding the data lock.

Other bug fixes

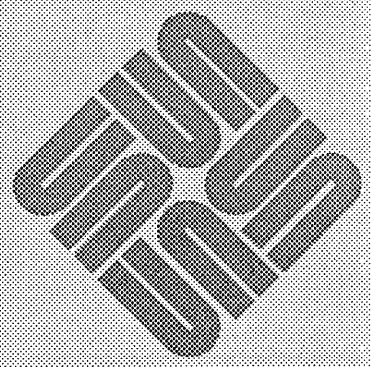
`window_destroy()` now closes the file descriptor of the window, plugging a file descriptor leak. (A similar bug in the old-style `tool_destroy()` call has also been fixed.)

The selection library has been fixed so it will not steal the socket of another RPC server in the same process; in 3.0 this would occasionally happen leading to random hung processes.

Input to a tty subwindow using `ttysw_input()` shows up immediately; in 3.0 it didn't show up until you moved the mouse into the subwindow.

6

Errata and Addenda for 3.0 Manuals



6

Errata and Addenda for 3.0 Manuals

- *Pixrects Reference Manual* (Part Number: 800-1254)
- *Windows & Window Based Tools: Beginner's Guide* (Part Number: 800-1287)
- *Fortran Programmer's Guide* (Part Number: 800-1371)
- *Pascal Programmer's Guide* (Part Number: 800-1376)
- *System Administration Manual for the Sun Workstation* (Part Number: 800-1323)
- *Assembly Language Reference Manual* (Part Number: 800-1372)
- *SunCore Reference Manual* (Part Number: 800-1257)
- *SunCGI Reference Manual* (Part Number: 800-1256)

Pixrect Reference Manual

The following pages are Errata and Addenda for Sun Release 3.0 version of the *Pixrect Reference Manual* — Sun Part Number: 800-1254.

6.1. Graphics: Pixrect Reference Manual

Raster Operations and Color Pixrects — Addenda

Page 14

Section 2.6, *Ops with a Constant Source Value*, should include the following information:

1) As of this release, Sun fully supports 1-bit and 8-bit pixrects; Sun partially supports pixrects of 16 and 32 bits. Currently, operations can only be performed between pixrects of the same depth, unless the source pixrect depth is 1 bit and the destination pixrect depth is 8 bits. This exception is discussed below.

2) The color argument is encoded in the raster operation code with the `PIX_COLOR` macro. `PIX_COLOR` is used in two cases: (1) if the source pixrect is NULL, in which case the destination pixrect is operated on using the source value specified with `PIX_COLOR`; and (2) if the source pixrect and destination pixrect have different depths (a source pixrect depth of 1 and a destination pixrect depth of 8), in which case the destination pixrect is operated upon per (4) below.

3) For source and destination pixrects of equal depths, all raster operations are valid and will act on color indices. For example, when you OR two colors, the operation is performed on the two color indices, not the colors themselves.

4) If a source pixrect has a depth of 1 and a destination pixrect has a depth of 8, the source pixrect is treated as though each pixel set to 0 was depth-8 (0 0 0 0 0 0 0 0), and each pixel set to 1 was depth-8 and of the color indicated by the `PIX_COLOR` index. The two indices are operated upon as in (3). When `PIX_COLOR` is omitted or set to 0 in the `op` argument but is needed for a raster operation, then a `PIX_COLOR` index of −1 is used (foreground).

5) On monochrome machines, a non-zero `PIX_COLOR` index is treated as 1, and a zero `PIX_COLOR` index is treated as 0.

Grayscale Workstations — Addenda

Page 22

Section 2.8, *Colormap Access*, should include the following information on colormaps and grayscale workstations:

1) For clarification: Sun colormaps are set up as follows:

- The colormap size for monochrome monitors is 2.
- The colormap size for grayscale monitors is 256.
- The colormap size for color monitors is 256.

2) If you plan to use a colormap with your grayscale workstation, be sure to load all three (RGB) colormaps with the same values when programming the colormap. It will ensure that your application will behave as you want if you use a color workstation.

3) All current Sun grayscale workstations use the Red gun to drive its grayscale monitor.

Multi-Pixel Operations — Addenda

Section 2.7 should include the following: (1) two new routines that draw textured or solid lines and polylines with width, and (2) a new routine that draws multiple points.

Draw Textured or Solid Lines with Width

```
#define pr_line(pr, x0, y0, x1, y1, brush, tex, op)
struct pixrect *pr;
int x0, y0, x1, y1;
struct pr_brush *brush;
struct pr_texture *tex;
int op;
```

The `pr_line` macro draws a textured line based on the Bresenham line drawing algorithm, using a pen-up, pen-down approach. The user may specify a pattern of an arbitrary length or use a predefined default pattern (dash-dot, dotted, etc.). All pattern segments (and their corresponding offsets) can be adjusted according to the angle at which the line is drawn.

Note: If the brush pointer is null, or if the width is 0 or 1, a single width vector is drawn.

The line is drawn in the pixrect indicated by `pr`, with endpoints at (`x0, y0`) and (`x1, y1`). The `brush` field is a pointer to a structure of type `pr_brush` which holds the width of the line segments to be rendered. If the `brush` pointer is null, or if `width` is `0` or `1`, a single width vector is drawn. The `pr_brush` structure is defined in the include file <`pixrect/pr_line.h`> as follows:

```
typedef struct pr_brush {
    int width;
} Pr_brush;
```

Note: If the tex pointer is null, a solid vector is drawn.

The `tex` field is a pointer to a structure of type `pr_texture`. If the `tex` pointer is null, a solid vector is drawn.

The pr_texture structure is defined in the include file <pixrect/pr_line.h> as follows (fields that begin with the prefix res_ are reserved for program internals, and are not user-definable):

```
typedef struct pr_texture {
    short *pattern;
    short offset;
    struct pr_texture_options {
        unsigned startpoint : 1,
        endpoint : 1,
        balanced : 1,
        givenpattern : 1,
        res_fat : 1,
        res_poly: 1,
        res_mvlist : 1,
        res_right : 1,
        res_close : 1;
    } options;
    short res_polyoff;
    short res_oldpatln;
    short res_fatoff;
} Pr_texture;
```

pattern is a pointer to an array of shorts which contains the length of each segment in the pattern. The lengths are in units of pixels. If the line is drawn at an angle, the lengths drawn can be adjusted (the givenpattern field set to 0) to correspond to the length of the pattern if a horizontal or vertical line was drawn. This array must be null-terminated. The first segment of the pattern array is assumed to be pen-down, and following segments alternate.

The addresses of the following predefined pattern arrays may be stored in the pattern field of the texture structure as well:

```
extern short pr_tex_dotted[];
extern short pr_tex_dashed[];
extern short pr_tex_dashdot[];
extern short pr_tex_dashdotted[];
extern short pr_tex_longdashed[];
```

The user-defined elements of the pattern array are not altered within the routine, supporting multiple calls using the same pattern. offset is an integer offset into the pattern, specified in pixels. Since the first segment of the pattern array is assumed to be pen-down, you must specify an offset to start on a pen-up segment. offset is adjusted according to the angle at which the line is drawn if the original pattern was adjusted (dependent upon the givenpattern bit, described later). Because of integer approximation, the adjusted offset could vary plus or minus one pixel from the exact adjusted offset.

In the options bit fields, if startpoint is set, the first point is always drawn, and if endpoint is set, the last point is drawn; if these are not specified, the line will be drawn with no extra pixels set. The balanced bit field effectively centers the pattern within the line by computing an offset into the pattern. If the givenpattern bit is set, the pattern is drawn without true length correction, at any angle; this increases performance. However, the pattern of radiating lines

from a common center will form concentric squares instead of circles. If the `givenpattern` bit is not set, the segment length of each element of the pattern is adjusted according to the angle at which the line is drawn. The true (angle-dependent) segment lengths are computed for one period of the pattern, using an incremental algorithm which approximates the formula:

angle_pattern_length = *given_pattern_length* * cos (*angle*)

where all units are in pixels, and *angle* is measured from the positive *x*-axis. Since the algorithm angle-corrects for one period of the pattern, the longer the period, the more exact the results will be.

The `op` argument specifies the raster operations used to produce destination pixel values and color.

Draw Textured or Solid Polylines with Width

```
pr_polyline(dpr, dx, dy, npts, ptlist, mvlist, brush, tex, op)
struct pixrect *dpr;
int dx, dy, npts;
struct pr_pos *ptlist;
u_char *mvlist;
struct pr_brush *brush;
struct pr_texture *tex;
int op;
```

`pr_polyline` draws a polyline, or a series of disjoint polylines, using the features available in `pr_line`. The polyline is drawn in the destination pixrect indicated by `dpr`, with `dx` and `dy` being the offset into the destination pixrect for vertices to be translated in x and y, respectively. `npts` is the number of vertices in the polyline (which is always the number of lines plus 1). The `ptlist` field is an array of `npts` structures of type `pr_pos` (which hold vertices). The `mvlist` field is a pointer to an array of `npts` elements in which if any element after the first is non-zero, a segment is not drawn to that vertex. The first element of the `mvlist` array controls whether the polyline(s) are automatically closed; if set, each continuous polyline is closed. If disjoint polylines are not desired (no `mvlist` is specified), the constants `POLY_CLOSE` and `POLY_DONTCLOSE` determine this behavior. `POLY_CLOSE` and `POLY_DONTCLOSE` are defined as follows:

```
#define POLY_CLOSE   ((u_char *) 1)
#define POLY_DONTCLOSE   ((u_char *) 0)
```

The `brush` field is a pointer to a structure of type `pr_brush`, and the `tex` field is a pointer to a structure of type `pr_texture`. If the `tex` pointer is null, a solid vector is drawn. If the `brush` structure is null, single-width vectors are drawn. `op` specifies the raster operations used to produce destination pixel values and color. `brush` and `tex` are described in detail under `pr_line`.

Draw Multiple Points

```
pr_polypoint(dpr, dx, dy, npts, ptlist, op)
struct pixrect *dpr;
int dx, dy, npts;
struct pr_pos *ptlist;
int op;
```

The `pr_polypoint` routine draws an array of points on the screen under the

control of the op argument. The array of points is drawn in the destination pixrect dpr, with an offset specified by the arguments dx and dy. Npts is the number of points to be rendered, and ptlist is a pointer to an array of structures of type pr_pos, which hold the vertices for each point. Color is encoded in the op argument. Portions of the array outside the pixrect are clipped unless the PIX_DONTCLIP flag is set in the op argument.

If npts is negative, pr_polypoint fails, and PIX_ERR is returned.

Plane Groups — Addenda

Chapter 2, *Pixrect Operations*, should include a new section on Plane Groups that describes the following routines.

A *plane group* is a subset of a frame buffer pixrect. Each plane group is a collection of one or more related bit planes with stored state (plane mask, color map, etc.). Each pixrect has a current plane group which is the target of attribute, color map, and rendering operations.

A plane group is described by a small constant in the include file <pixrect/pr_planegroups.h>:

```
#define PIXPG_CURRENT           0
#define PIXPG_MONO              1
#define PIXPG_8BIT_COLOR        2
#define PIXPG_OVERLAY_ENABLE    3
#define PIXPG_OVERLAY           4
```

Plane group 0 is the currently active plane group for the pixrect. In the initial implementation, the plane group is encoded as a 7-bit field in the pixrect attribute word.

You should assume that all implemented plane groups with non-zero plane masks are active. Be sure to explicitly disable plane groups which are not in use by setting their plane masks to zero with pr_set_planes (described later).

Determine Supported Plane Groups

```
ngroups = pr_available_plane_groups(pr, maxgroups, groups);
Pixrect *pr;
int maxgroups;
char groups[maxgroups]
```

pr_available_plane_groups provides a means by which you determine which plane groups are supported by the machine you are working on. pr_available_plane_groups fills the character array groups with true (1) values for the plane groups implemented by the pixrect pr. The entry for the current plane group (groups[0]) array is always set to false (0). The size of groups is passed to the function as maxgroups to avoid overwriting the end of the array.

pr_available_plane_groups returns the index of the highest-numbered implemented plane group plus one.

Get Implemented Plane Group

```
group = pr_get_plane_group(pr);
Pixrect *pr;
```

`pr_get_plane_group` returns the current plane group number for the pixrect `pr`. If the current plane group is unknown, the function returns `PIXPG_CURRENT`.

Set Plane Group and Mask

```
void pr_set_plane_group(pr, group);
Pixrect *pr;
int group;

void pr_set_planes(pr, group, planes)
Pixrect *pr;
int group;
int planes;
```

`pr_set_plane_group` sets the current plane group for the pixrect `pr` to the value given by `group`. If this plane group is `PIXPG_CURRENT` or unimplemented, `pr_set_plane_group` does nothing.

The `pr_set_planes` function is equal to a `pr_set_plane_group(pr, group)` followed by `pr_putattributes(pr, &planes)`. `planes` contains a bitplane write-enable mask. Only those planes corresponding to mask bits having a value of `1` will be affected by subsequent pixrect operations. However, these planes can still be read.

Memory Pixrects — Errata

Page 36

Section 4.2, *Create Memory Pixrect* should be corrected to include the following proviso:

> In a future release, memory pixrects created with `mem_create` on a 32-bit system will have each line padded to a 32-bit boundary, unless it is only 16 bits wide; that is, the `md_linebytes` structure member will contain either 2 or a multiple of 4. Non-pixrect code which operates on memory pixrect data should examine `md_linebytes` instead of relying on the `mpr_linebytes` macro.
>
> If it is necessary to create a memory pixrect with rows padded to 16-bit boundaries, the `mem_point` function should be used.

Pages 37-38

Section 4.4, *Pixel Layout in Memory Pixrects* should be corrected to read as follows:

> Currently, memory pixrects are only supported for pixels of 1, 8, 16, or 32 bits (not 24 bits). Some operations may not work for 16-bit and 32-bit pixels.

SunCGI Reference Manual

The following pages are Errata and Addenda for Sun Release 3.0 version of the *SunCGI Reference Manual* — Part Number: 800-1256

6.2. Graphics: SunCGI Reference Manual — Errata

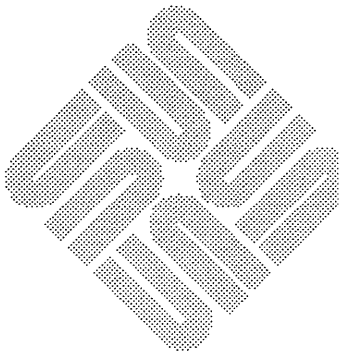
1) In Chapter 2 of the *SunCGI Reference Manual*, many functions are listed as possibly generating error ENOTCCPW[112]. This is correct, but the message is incorrect. The message should be:

   ENOTCCPW [112] Function or argument not compatible with CGIPW mode.

2) EVALOVWS[69] is no longer detected by `pixel_array`, `cgipw_pixel_array` or `inquire_pixel_array`. `pixel_array` and `cgipw_pixel_array` merely clip their output. `inquire_pixel_array` returns a 0 for array entries outside the view surface.

3) The example program in Figure G-1 of the *SunCGI Reference Manual* has an incorrect call to `cfopenvws`. Figure 6-1 below contains a new example program. In addition Table G-4 had an erroneous description of the `cfopenvws` function. Table 6-1 contains a new description of this function.

Figure 6-1 *Example CGI FORTRAN Program*



```
      program test

      parameter (ibignum=256)

      integer name
      character screenname* (ibignum)
      character windowname* (ibignum)
      integer windowfd
      integer retained
      integer dd
      integer cmapsize
      character cmapname* (ibignum)
      integer flags
      character ptr* (ibignum)
      integer noargs
c         coordinates of glass
      integer xc(10),yc(10),n
c         coordinates of waterline.
      integer xc2(2),yc2(2)
      data xc /0,-10,-1,-1,-15,15,1,1,10,0 /
      data yc /0,0,1,20,35,35,20,1,0,0 /
      data xc2 /-12,12/
      data yc2 /33,33/

c         open cgi
      call cfopencgi()
c         open a color pixwin
      dd = 5
      call cfopenvws(name,screenname,windowname,
     +  windowfd,retained,dd,cmapsize,cmapname,
     +  flags,ptr,noargs)
c         reset VDC space
      call cfvdcext(-50,-10,50,80)
c         draw martini glass and waterline
      n = 10
      call cfpolyline(xc,yc,n)
      n = 2
      call cfpolyline(xc2,yc2,n)
c         sleep for 10 seconds
      call sleep(10)
c         close and exit
      call cfclosecgi()
      call exit()
      end
```

Table 6-1 *SunCGI Fortran Binding – Part IV*

CGI Specification Name	*Fortran Binding*

Open View Surface (SunCGI Extension)	`call cfopenvws(name,screenname,windowname,`
	`1 windowfd,retained,dd,cmapsize,`
	`2 cmapname,flags,ptr,noargs)`
	
	`integer function cfopenvws(name, screenname, windowname,`
	`1 windowfd, retained, dd, cmapsize, cmapname, flags,`
	`2 ptr, noargs)`
	`integer name`
	`character*(*) screenname`
	`character*(*) windowname`
	`integer windowfd`
	`integer retained`
	`integer dd`
	`integer cmapsize`
	`character*(*) cmapname`
	`integer flags`
	`character*(*) ptr`
	`integer noargs`

SunCore Reference Manual

nr cP 1 The following are Errata and Addenda for Sun Release 3.0 version of the *SunCore Reference Manual* — Sun Part Number: 800-1257

6.3. Graphics: SunCore Reference Manual — Errata

Section 2.5, page 21.

Handling Signals with SunCore (SunCore Extension)

SunCore uses the **SunView** notifier to deal with signals. A program that uses the **SunCore** library should not use the `signal()` call. For more information see the *SunView Programmer's Guide.*

Note: The SIGWINCH (SIGnal WINdow Change) signal is received via the **SunView** notifier to indicate to **SunCore** that graphical images may need to be redrawn. The *notify function* used by **SunCore** is `_core_winsig`.

Application programs that wish to do their own signal handling should ensure that `_core_winsig` is invoked as well.

SunCore View Surfaces

View Surface Types

Appendix B, page 119.

`cg1dd`
The Sun-1 color graphics display used as a raw device.

`cg2dd`
The Sun-2 or Sun-3 color graphics display used as a raw device.

`cg4dd`
The on-board color frame buffer used as a raw device.

`pixwindd`
A monochrome (one bit deep) graphics window within the **SunView** window environment. This window may appear on either a color or monochrome display.

`cgpixwindd`
A color graphics window within the Suntools window environment. This window must appear on a color display.

`gp1dd`
A Sun-2/160 or Sun-3/160 graphics display with a *Graphics Processor* option.

`gp1pixwindd`
A color graphics window within the Suntools window environment running on a Sun-2/160 or Sun-3/160 color graphics display with a *Graphics Processor* option.

Only view surface types `cg1dd`, `cg2dd`, `cg4dd`, `cgpixwindd` `gp1dd`, and `gp1pixwindd` support hidden surface removal. In the discussion above, gray scale devices are considered to be color devices.

The term 'raw device' above implies that the physical device specified by `vwsurf.screenname` is used completely and only for display of graphics output directed to one view surface. This allows somewhat more efficient display of output primitives. It also implies that the user has not started up a Suntools window environment using the device as a desktop.

Low-level device-dependent routines are not part of **SunCore**. For efficiency, such routines are necessary for some applications. The *Pixrect Reference Manual* contains information on low-level routines corresponding to `bw1dd`, `bw2dd`, `cg1dd`, `cg2dd`, `cg4dd` and `gp1dd`, (the 'pixrect' level) and `pixwindd`, `cgpixwindd` and `gp1pixwindd` (the `pixwin` level).

View Surface Specification for Window Devices

Section B.4.2, page 127.

An example of a program that opens multiple view surface follows. This section describes the mechanism that **SunCore** uses to decide which window to use for a new view surface of types `pixwindd`, `cgpixwindd` and `gp1pixwindd`.

SunCore first checks to see if there is a window name (e.g `/dev/win5`) specified in `vwsurf.windowname`. If so **SunCore** will try to open that window directly.

Note: **SunCore** allows the user to open multiple view surfaces on the same window in this manner. While this is not flagged as an error, it is of questionable value. Each view surface would obscure the graphical contents of previous ones.

Otherwise, the WINDOW_GFX environment variable is used to determine which window to open. **SunCore** tests whether any surface has already been opened by this method. If WINDOW_GFX has already been used in this manner, the program `/usr/lib/view_surface` is invoked to create a new window on the same physical display device as WINDOW_GFX. This new window becomes the view surface. Thus, if a **SunCore** program is run from the tty subwindow of a *Graphics Tool*, the first default view surface will occupy the display space covered by the graphics subwindow of the tool. Subsequent default view surfaces will appear as graphics windows, each within a separate *View Surface Tool* on the same screen as the *Graphics Tool.*

This **SunCore** program takes two arguments that contain window names of the form `/dev/win???`. Two corresponding view surfaces are opened. Vertical vectors are drawn in the first view surface, horizontals in the second. It was

written to run on monochrome windows – change references of `pixwindd` to `cgpixwindd` or `gp1pixwindd` if appropriate. (Window names can be obtained by entering `echo $WINDOW_ME` in a window).

```
#include <usercore.h>
#include <stdio.h>

int pixwindd();
struct vwsurf vwsurf1 = DEFAULT_VWSURF(pixwindd);
struct vwsurf vwsurf2 = DEFAULT_VWSURF(pixwindd);

main(argc, argv)
int argc;
char *argv[];
{
    register int i;

    if (argc<3) {
        fprintf(stderr, "Usage: %s /dev/win? /dev/win?0, argv[0]);
        exit(1);
    }

    initialize_core(BASIC, NOINPUT, TWOD);
    strncpy(vwsurf1.windowname, argv[1], DEVNAMESIZE);
    initialize_view_surface(&vwsurf1, FALSE);
    strncpy(vwsurf2.windowname, argv[2], DEVNAMESIZE);
    initialize_view_surface(&vwsurf2, FALSE);
    set_window(-100.0, 100.0, -100.0, 100.0);
    srandom(getpid());
    if (select_view_surface(&vwsurf2))
        fprintf(stderr, "Cannot select second view surface");
    set_viewport_2(0.0, 1.0, 0.0,.75);
    create_retained_segment(101);
        for (i=0; i<41; i++) {
            move_abs_2(-100.0, -100.0 + 5 * i);
            line_abs_2(100.0, -100.0 + 5 * i);
        }
    close_retained_segment(101);
    deselect_view_surface(&vwsurf2);
    if (select_view_surface(&vwsurf1))
        fprintf(stderr, "Cannot select first view surface");
    set_viewport_2(0.0, 1.0, 0.0,.75);
    create_retained_segment(100);
        for (i=0; i<41; i++) {
            move_abs_2(-100.0 + 5 * i, -100.0);
            line_abs_2(-100.0 + 5 * i, 100.0);
        }
    close_retained_segment(100);
    deselect_view_surface(&vwsurf1);
    sleep(10);
    terminate_core();
}
```

The following example program replaces the program in Figure B-3 in the *SunCore Reference Manual.*

```
#include <sunwindow/window_hs.h>
#include <sys/file.h>
#include <sys/ioctl.h>
#include <sun/fbio.h>
#include <stdio.h>
#include <usercore.h>

int bw1dd(); /* All device-independent/device-dependent */
int bw2dd(); /* routines are referenced in this function. */
int cg1dd(); /* This means the linker will pull in all of them. */
int cg2dd();
int cg4dd();
int gp1dd();
int pixwindd();
int cgpixwindd();
int gp1pixwindd();

static char *devchk;
static float glassdx[ ] = { -10.0,9.0,0.0,-14.0,30.0,-14.0,0.0,9.0,-10.0 };
static float glassdy[ ] = { 0.0,1.0,19.0,15.0,0.0,-15.0,-19.0,-1.0,0.0 };
static int devhaswindows;
static struct vwsurf nullvs = NULL_VWSURF;

main(argc, argv)
int argc;
char **argv;
{
    struct vwsurf vwsurf;

    initialize_core(BASIC, NOINPUT, TWOD);
    if(get_view_surface(&vwsurf, argv))
        exit(1);
    initialize_view_surface(&vwsurf, FALSE);
    select_view_surface(&vwsurf);
    set_viewport_2(0.125, 0.875, 0.125, 0.75);
    set_window(-50.0, 50.0, -10.0, 80.0);
    create_temporary_segment();
    move_abs_2(0.0, 0.0);
    polyline_rel_2(glassdx, glassdy, 9);
    move_rel_2(-12.0, 33.0);
    line_rel_2(24.0, 0.0);
    close_temporary_segment();
    sleep(10);
    deselect_view_surface(&vwsurf);
    terminate_core();
}
/*
 * get_view_surface -- Determines from command-line arguments and the
 * environment a reasonable view surface for a SunCore program to run on.
 */

int get_view_surface(vsptr, argv)
struct vwsurf *vsptr;
```

```
char **argv;
{
    int devfnd, fd, fbtype, chkdevhaswindows();
    char *wptr, dev[DEVNAMESIZE], *getenv();
    struct screen screen;

    *vsptr = nullvs;
    devfnd = FALSE;
        /*
         * If command line arguments are passed, process them using
         * win_initscreenfromargv() (see the "SunView System Programmer's
         * Guide").  The only option used by get_view_surface is the -d
         * option, allowing the user to specify the display device on
         * which to run.
         */
    if (argv) {
        win_initscreenfromargv(&screen, argv);
        if (screen.scr_fbname[0] != ' ') {
            /* -d option was found */
            devfnd = TRUE;
            strncpy(dev, screen.scr_fbname, DEVNAMESIZE);
            /*
             * Check to see if this device has a window system
             * running on it.  If so devhaswindows will be TRUE
             * following the call to win_enumall.  win_enumall is
             * a function in libsunwindow.a.  It takes a function
             * as its argument, and applies this function to
             * every window being displayed on any screen by the
             * window system.  To do this it opens each window
             * and passes the windowfd to the function.  The
             * enumeration continues until all windows have been
             * tried or the function returns TRUE.
             */
            devchk = dev;
            devhaswindows = FALSE;
            win_enumall(chkdevhaswindows);
        }
    }
    if (!devfnd)
        /* No -d option was specified */
        if (wptr = getenv("WINDOW_ME")) {
            /*
             * Running in the window system.  Find the device
             * from which this program was started.
             */
            devhaswindows = TRUE;
            if ((fd = open(wptr, O_RDWR, 0)) < 0) {
                fprintf(stderr, "get_view_surface: Can't open %s0, wptr);
                return(1);
            }
            win_screenget(fd, &screen);
            close(fd);
            strncpy(dev, screen.scr_fbname, DEVNAMESIZE);
```

```
    } else {
        /*
         * Not running in the window system.  Assume device
         * is /dev/fb.
         */
        devhaswindows = FALSE;
        strncpy(dev, "/dev/fb", DEVNAMESIZE);
    }
/* Now have device name.  Find device type. */
if ((fd = open(dev, O_RDWR, 0)) < 0) {
    fprintf(stderr, "get_view_surface: Can't open %s0, dev);
    return(1);
}
if ((fbtype = pr_getfbtype_from_fd(fd)) == -1) {
    fprintf(stderr, "get_view_surface: pr_getfbtype_from_fd() failed for %s0, dev);
    close(fd);
    return(1);
}
close(fd);
/* Now have device type and know if window system is running on it. */
if (devhaswindows)
    switch (fbtype) {
    case FBTYPE_SUN1BW:
    case FBTYPE_SUN2BW:
        vsptr->dd = pixwindd;
        break;
    case FBTYPE_SUN1COLOR:
    case FBTYPE_SUN2COLOR:
    case FBTYPE_SUN4COLOR:
        vsptr->dd = cgpixwindd;
        break;
    case FBTYPE_SUN2GP:
        vsptr->dd = gp1pixwindd;
        break;
    default:
        fprintf(stderr, "get_view_surface: %s is unknown fbtype0, dev);
        return(1);
    }
else
    switch (fbtype) {
    case FBTYPE_SUN1BW:
        vsptr->dd = bw1dd;
        break;
    case FBTYPE_SUN2BW:
        vsptr->dd = bw2dd;
        break;
    case FBTYPE_SUN1COLOR:
        vsptr->dd = cg1dd;
        break;
    case FBTYPE_SUN2COLOR:
        vsptr->dd = cg2dd;
        break;
    case FBTYPE_SUN4COLOR:
```

```
            vsptr->dd = cg4dd;
            break;
        case FBTYPE_SUN2GP:
            vsptr->dd = gp1dd;
            break;
        default:
            fprintf(stderr, "get_view_surface: %s is unknown fbtype0, dev);
            return(1);
        }
    /* Now SunCore device driver pointer is set up. */
    if (!devhaswindows || devfnd)
        /*
         * If no window system on device or -d option was specified,
         * tell SunCore which device.  Otherwise, let SunCore figure
         * out the device itself from WINDOW_GFX so the default
         * window will be used if desired.
         */
        strncpy(vsptr->screenname, dev, DEVNAMESIZE);
    return(0);
}

static int chkdevhaswindows(windowfd)
int windowfd;
{
    struct screen windowscreen;

    win_screenget(windowfd, &windowscreen);
    if (strcmp(devchk, windowscreen.scr_fbname) == 0) {
        /*
         * If this window is on the display device we are checking,
         * set the flag TRUE.  Return TRUE to terminate the enumeration.
         */
        devhaswindows = TRUE;
        return(TRUE);
    }
    return(FALSE);
}
```

Windows & Window Based Tools: Beginner's Guide

The following pages are Errata and Addenda for Sun Release 3.0 version of the *Windows & Window Based Tools: Beginner's Guide* — Sun Part Number: 800-1287.

Subwindow resizing

p. 27

After *Resizing a window* in section 3.1, add:

Resizing subwindows

Many tools are composed of more than one window. The windows in such tools are more properly referred to as *subwindows*. For example in `mail-tool` there is a subwindow for message headers, a subwindow with various control buttons and a subwindow for the display of the current message.

You can adjust the size of subwindows in a tool, for example to see more message headers in `mailtool`. This is a window manipulation that cannot be performed from the frame menu; you have to use an *accelerator*. To resize subwindows, move the cursor into the border of the subwindow you want to resize so that the *target circle cursor* appears, hold down the CTRL key, and press the middle mouse button. Depending on where the cursor is on the subwindow border, the cursor will turn into one of the same *constrained* or *unconstrained resizing cursors* you see when resizing tools. Then move the mouse until the subwindow is the desired size. You cannot make a subwindow bigger than the tool it is in.

Advanced subwindow adjustments

The above is the most common resizing operation. There are others:

- Similar to moving the frame itself, you can *move* a subwindow within the frame without resizing it by moving the cursor onto the border between two subwindows, holding down the middle mouse button, and moving the cursor to the desired position. You can use this to change the order of subwindows in a tool — for example putting the scratch window in `textedit` at the bottom instead of the top.
- By holding down the SHIFT key when moving OR resizing subwindows, the other subwindows will not adjust to fill any empty space that results. You have little reason to use this feature except when moving subwindows around to a new layout.

Many tools with multiple subwindows allow you to specify the sizes of the different subwindows in advance, either in `defaultsedit` or through command

line options.

Note: Some tools handle the resizing of subwindows better than others, so be cautious in shrinking or expanding subwindows too much.

`shelltool` arrow keys

p. 62

In section 7.4, add:

By default the keys [R8], [R10], [R12] and, [R14] on the Sun-2 and Sun-3 keyboard act like arrow keys in terminal-based applications such as `vi` unless you use the `setkeys noarrows` option or set *Arrow_Keys* in `defaultsedit`. For more information on changing these keys' function, see section 16.2.

New `input_from_defaults` program

p. 69

In section 8.1, add:

In 3.2 there is a new way to set your keyboard function key preferences that obsoletes `setkeys`. There is a new category, *Input*, in `defaultsedit` that lets you specify the option *Left_Handed*. When you [Save] your changes in `defaultsedit`, the program `input_from_defaults` is run to change the current input settings to your preferences; it automatically determines and sets your keyboard type at the same time. If you specify *Left_Handed* then it also swaps the left and right mouse buttons.

You can run the program `input_from_defaults` to change the settings before you run `suntools`. You can modify the sample lines for your `.login` in section 2.4, *Starting* `suntools` *Automatically Upon Login*, so that `input_from_defaults` is run before you start `suntools`.

Note: When you change the input settings using `setkeys`, `defaultsedit` or `input_from_defaults` the change affects all windows, not just text subwindows, and it takes place immediately, even if someone else is running `suntools`. It remains in effect until one of the programs is rerun or the machine is restarted.

`textedit` menu out of date

p. 70

The text menu shown in Figure 8-4 is out of date. The text menu includes the item `Put then Get` documented in section 9.1.

Keyboard equivalent for *Put* p. 83

Under *Copying a Selection* in section 8.3, change

> The [Put] function key (usually [L6]) stores selected characters on the shelf without deleting them.

to

> The [Put] function key (usually [L6]), or [CTRL-P], stores selected characters on the shelf without deleting them.

Search-and-Replace of text p. 113

At the end of *The Again Operation* in section 9.2 add:

Another common operation that can be accelerated by using the [Again] key is searching for a string and replacing it. One way to do this is to select the string to search for, [Find] (if you selected the string in the same window then this will jump to the *next* instance of the string), [Delete], type replacement string. Then press [Again] and the Find-Delete-*replacement_string* operation will be repeated.

Pressing [Again] will do nothing if `textedit` can't find any more strings. If an unwelcome replacement occurs, press [Undo] and continue with [Again].

Checkpointing in `textedit` p. 115

In section 9.6 add:

You can avoid losing more than a small amount of changes that you make while editing by specifying the `checkpoint` *N* option to `textedit`; see the following errata on *Checkpoint_frequency*.

p. 116

In section 9.8 add:

`Checkpoint_frequency` *integer*

> If *Integer* is non-zero, the file you are editing will periodically be copied to a file with the same name with `%%` on the end. This will occur after every *Integer* edits to the file; each character insertion or text operation counts as an edit. The entire file is copied to this backup file, so this serves as an extra protection against losing critical work. To recover from a crash you need only copy the *filename*`%%`, losing at most the last *Integer*

changes to the file. The creation of this backup file is a separate process from the *filename*`%` file that `textedit` creates when you save a file (see `textedit`'s *Backup Files* in the previous chapter). The default is `0` which means that checkpointing is disabled.

Command line arguments to `textedit` p. 115

In section 9.8, after

> You can specify a number of options to the text facility using `defaultsedit`.

add:

> You can also set many of these options as command line options when you start up `textedit` (and `cmdtool`). This allows you to have different settings from your defaults in particular `textedit` tools. See the `textedit` (1) man page for more information.

List of keyboard equivalents p. 117

Section 9.9, *Summary of Text Facility*, should be modified as follows:

Add after *Bracket a selection*:

Copy primary selection to caret — Choose `Put then Get` from the menu, or use the accelerator CTRL-P.

Go to end of text — CTRL-RETURN moves the insertion point to the end of the text, positioning the text so that the insertion point is visible.

Replace

Get (Usually L8)

with

Get (Usually L8) or CTRL-G

Similarly,
CTRL-P is a keyboard equivalent for Put, CTRL-D is a keyboard equivalent for Delete and CTRL-F is a keyboard equivalent for Find.

Checkpointing in `cmdtool` p.124

At the end of section 10.2, add

You can use the same checkpoint facility in `cmdtool` that exists for `textedit` to safeguard important files. Turn on checkpointing by setting *Checkpoint_frequency* in the *Tty* category in `defaultsedit` to some non-zero value. See the errata section for `textedit` above for more information about checkpointing.

New `defaultsedit` **look** p. 127

In section 11.1 add

`defaultsedit` (1) looks different in 3.2. The different categories of options are available from a single `Category` choice item. Click the left mouse button on `Category` to cycle through the categories available or hold down the right mouse button to see a menu of categories.

There is a new category, *Input* which allows you to tailor keyboard function key usage (previously set using `setkeys`), mouse responsiveness several mouse and keyboard settings. When you [Save] your defaults, the program `input_from_defaults` is run to load your new settings. See the errata for section 8.1 for more information.

Modifying subwindow behavior in `cmdtool` p. 166

After section 16.2 add the following

`cmdtool` reads both `.ttyswrc` and `.textswrc`; the latter overrides the former in cases where the same function key is mapped in both.

FORTRAN Programmer's Guide

The following pages are Errata and Addenda for Sun Release 3.0 version of the *FORTRAN Programmer's Guide* — Sun Part Number: 800-1371.

6.4. Developing and Maintaining FORTRAN Programs — Errata

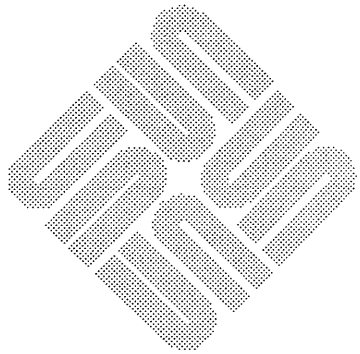
Page 9

Insert the following line at the bottom of the page:

- `f77` permits a maximum of 19 contiguous continuation lines

Page 20

Replace the subroutine named `startupcore.f` with the following:



```
        subroutine startupcore
        include '/usr/include/f77/usercore77.h'
C       make initializing calls to core library

    common /vwsurf/ vsurf
    integer vsurf(VWSURFSIZE), SelectVwsurf

        integer pixwindd, InitializeCore, InitializeVwsurf
C    (Use cgpixwindd instead of pixwindd to run with color)

        external pixwindd
        character*4 envreturn
        character*50 sccsid
    integer loc
        data sccsid/"@(#)startupcore.f  1.1     84/03/05\n"/
    data vsurf /VWSURFSIZE*0/

    vsurf(DDINDEX) = loc(pixwindd)
        if (InitializeCore(BASIC, NOINPUT, TWOD) .ne.0) call exit
        call getenv( "WINDOW_ME", envreturn )
        if (envreturn .eq. "     ") then
            write(0,*)"must run in a window"
            call exit(2)
        endif
        if (InitializeVwsurf( vsurf, FALSE) .ne. 0) call exit(2)
        if (SelectVwsurf(vsurf) .ne. 0) call exit(3)
        call SetWindow( -1.5, 1.5, -2.0, 2.0 )
        call CreateTempSeg()
        return
        end


        subroutine closecore
        include '/usr/include/f77/usercore77.h'
C       make terminating calls to core library
    common /vwsurf/ vsurf
    integer vsurf(VWSURFSIZE)

        call CloseTempSeg()
        call DeselectVwsurf( vsurf )
        call TerminateCore()
        return
        end
```

6.5. Input and Output — Errata

Page 48

Insert the following paragraphs at the end of the page:

FORTRAN unformatted IO is generally not used very much. There are two kinds of unformatted IO: sequential and direct.

Sequential unformatted IO, like sequential formatted IO, is record oriented: each `read` reads exactly one record, each `write` writes exactly one record. But since the UNIX† file system has no records, how does FORTRAN find out where the record boundaries are? The information is represented in the data.

In formatted files, the end-of-record is denoted by a special character, the **newline** character (octal 012). The newline character is not accessible to the FORTRAN program, but is inserted or filtered-out as appropriate, by the run-time system. While it would be nice to have an analogous separator for unformatted records, but there are no characters that can be unambiguously differentiated from unformatted data. For example, the newline character could be confused with an integer value of 10.

Preceeding and following each unformatted sequential record is an `integer*4` byte count. The trailing byte count is used to permit `backspace` to operate on records. How do those counts get there?

FORTRAN inserts the byte counts when writing records to files connected for unformatted sequential IO. The result is that FORTRAN programs cannot use an unformatted sequential `read` to read any data not written by an unformatted sequential `write` operation. So, if the first byte of your data file is "4", an unformatted sequential `read` will interpreted it as the first byte of a count, which thus has to be at least 0x4000000! This is probably not what was intended.

Direct unformatted IO, like sequential unformatted IO, is record oriented. But because you have to specify the record size in the `open` statements `recl` clause, the record length information is not buried in the file data. Since each transfer involves an integral number of records, you have to be a careful:

According to the FORTRAN specification, on input `exactly one` record is read. On input, the number of values required by the input list must be `less than or equal to` the number of values in the record.

As a general rule, avoid using the unformatted sequential `read` unless the file being read was originally written that way. If you want to use unformatted IO, try to use the unformatted direct `read` whenever possible, opening the file with `recl=1` if not all your input lists are the same length.

6.6. The Run Time Environment — Errata

Page 62

Replace the definition of the representation of `real` or `double precision` numbers at the middle of page with the following definition:

A `real` or `double precision` number is represented by the form:

•

† UNIX is a trademark of AT&T Bell Laboratories.

$$(-1)^{sign} * 2^{exponent-bias} * 1.f$$

where *f* is the bits in the fraction.

Replace the definition of the representation of subnormal numbers with the following definition:

The form of a subnormal number is

$$(-1)^{sign} * 2^{1-bias} * 0.f$$

where *f* is the bits in the significand.

6.7. Deviations from the FORTRAN 77 Standard — Errata

Page 87

Replace the last sentence in the second paragraph with the following:

The second part describes areas where this compiler and run time system violate the ANSI standard, either because the compiler or run time system cannot correctly implement the ANSI standard, or the standard is ambiguous. These violations seldom affect working programs and are often not even recognized by the standard validation tests.

Page 89

The section `include` *Statement* should have the following paragraphs appended to it:

If the name referred to by the `include` statement begins with the character '/', it is taken by `f77` to mean the absolute pathname of the include file.

Otherwise, the `f77` looks first for the include file in the directory containing the source code file with the `include` statement. If the file cannot be found there, `f77` looks for it in `/usr/include`.

Note that files included via `#include` may contain `#defines` and the like, while files included with the compiler `include` statement must contain only FORTRAN statements.

Page 91

Remove the subsection *Dummy Procedure Arguments.*

Remove the subsection *Assigned* `goto`.

Remove the subsection *Default files.*

Page 92

Remove the subsection *Exponent representation on Ew.dEe output.*

Pascal Programmer's Guide

The following pages are Errata and Addenda for Sun Release 3.0 version of the *Pascal Programmer's Guide* — Sun Part Number: 800-1376.

Sun Extensions to Berkeley Pascal

Page 89

Insert the following subsection immediately before section 8.3:

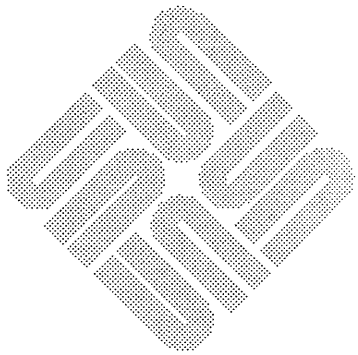
Logical `AND` **Token**

The character "&" can be used in programs to denote a logical `AND`.

Page 90

Append to end of text on page:

- Description: The `pc` preprocessor allows labels greater than 9999. The ISO standard limits labels to the range 0-9999.



System Administration Manual

The following pages are Errata and Addenda for Sun Release 3.0 version of the *System Administration Manual* — Sun Part Number: 800-1323.

6.8. Diag — A Disk Maintenance Program

This chapter describes `diag`, the Sun Microsystems standalone disk utility program. It describes how to start `diag`, how to prepare a new disk for operation, and how to fix problems on a disk.

6.9. Architecture

A disk consists of a stack of spinning platters, each with its own **head.** The heads, which all move together, travel back and forth across the surface of the platters between the outer rim of the stack and the center.

Each platter consists of a series of **tracks,** which are concentric rings that run 360 degrees around the platter. Each track is divided into **sectors,** and each sector is marked with a unique header. Sectors are the basic storage units; each sector contains 512 bytes of usable data. Sectors are also referred to as `blocks`.

A **cylinder** is a vertical stack of tracks; for example, the third track on all the platters in the stack is cylinder 3. Because the heads move together, they are always on the same cylinder.

Cylinder, Head and Sector Numbers

`Diag` understands disk addresses in two forms: cylinder/head/sector (CC/HH/SS), or absolute decimal block numbers (NNNNNN). `Diag` accepts either, but the CC/HH/SS form resembles the physical architecture of the disk more closely. Both forms identify one particular sector.

A CC/HH/SS address in the form CC/00/00 identifies a cylinder boundary; the cylinder starts at sector 0 in track 0. Similarly, each track starts at sector 0, so an address in the form CC/HH/00 identifies a track boundary.

Logical vs. Physical

An address can be logical or physical. A logical address represents a count of usable sectors from the beginning of the disk, and a physical address represents an actual location on the surface of the disk.

When UNIX reports disk errors, it provides the logical block number of the sector in error. It is listed by UNIX as the `abs blk #`. This is generally the actual sector that caused the error. However, due to controller hardware limitations, this number is occasionally approximate.

Partitions and File Systems

Partitions are regions on the disk set up by using the `partition` and `label` commands. They start on cylinder boundaries and are used by UNIX.

`Diag` has built-in partition tables for all standard disk drive configurations plus a couple of alternates; you can select these easily using the `label` command. If these are not adequate, the `partition` command also has a facility for developing custom partitions.

SCSI Interface

SCSI controllers are attached to the system through a host adapter, which presents a bus address where its registers are located. All commands to controllers pass through the host adapter, and thus through the same bus registers, but a particular target controller may be specified.

Controllers communicate with disks over an ST506 or ESDI interface, and with the host adapter over the SCSI bus. Controllers perform all defective sector handling automatically. This insulates software from the physical details of SCSI disks; it sees an idealized device consisting of a continuous stream of sectors.

SMD Interface

The CPU and file system must be more intimately acquainted with the details of the SMD disk; the SMD controllers are connected directly to the system bus. The CPU writes commands directly to the SMD control registers, and the controller carries out specified operations by communicating directly with the disk using the SMD interface. The device driver needs a lot of knowledge about the specific nature of the disk drive, and the filesystem takes advantage of this to arrange data on the disk in a way that will provide fast access for reading, writing, and updating files.

Removing Bad Sectors

All disk surfaces contain some media defects. The sectors where these occur must be removed from service to provide reliable operation.

Practically speaking, new defective sectors should never turn up during operation unless you have problems. All disks start life with a few surface defects, but the disk drive manufacturer and Sun each perform extensive surface analysis and slip or map these. Also, the procedure for preparing a new disk includes a third surface analysis to catch any surface defects caused during transit.

If you encounter surface defects after all that, it's usually caused by damage to the defective sector map, a "head kiss" where a head actually touches the surface of the disk, or an electronic glitch, which can corrupt the data on a sector making it look like a surface defect. These things usually indicate some underlying problem; if they happen once, they are likely to happen again. The disk troubleshooting section later in this chapter provides instructions for finding and fixing these problems

The following sections discuss defect handling on both SMD and SCSI disks.

SMD Bad Sectors

`Diag` commands deal with defective sectors on an SMD-controlled disk by *slipping* or *mapping* them. `Format` and `fix` search a defined area for defective sectors. If they find one, they attempt to slip it and if that fails they map it. The `map` command simply maps sectors on request.

Disks configured for sector slipping contain a spare data sector at the end of each track. When `diag` attempts to slip a sector, it removes the defective sector from service by identifying it with a special header, then it bumps subsequent logical sectors forward until the last sector moves to where the spare was.

If the disk is not configured for sector slipping, or if a track contains a second bad sector, you must rely on mapping. `Diag` writes the defective sector's address in the bad sector map and marks the sector with a header identifying it as a mapped sector. When the access to the mapped sector fails, the bad sector map redirects UNIX to use another sector on the spares track.

Sector slipping improves disk efficiency; it allows the disk to operate without the additional overhead of mapping.

To identify slipped sectors, mapped sectors, and to tell if the disk is configured for sector slipping, use the `rhdr` command, described later in this chapter.

SCSI Bad Sectors — ST506 Controllers

To deal with SCSI disk bad sectors, the ST506 controller marks the bad sector and resumes its sector count on the next sector. It then slips the physical locations of all subsequent sectors towards the end of the disk.

As defects add up, the sectors get skewed across cylinder and track boundaries. To deal with the difference between where a sector should be and where it has slipped to, the controller does a quick scan of the disk when it comes up. It divides the disk into zones and keeps a table of the amount of slippage for each zone. When it does a seek, it adds the amount of slippage for each zone it has to go through, so that it ends up looking somewhere near the desired sector.

Because slipping requires moving all subsequent sectors on the disk, all of diag's strategies for dealing with defective sectors involve reformatting the disk. When it's finished formatting and slipping sectors, it writes a list of the slipped sectors at the end of the disk.

SCSI Bad Sectors — ESDI Controllers

To deal with SCSI disk bad sectors, the ESDI controller uses a technique similar to the SMD controllers. Each track on the disk contains a spare sector. When a defective sector is found, it is marked invalid and all subsequent sectors on that track are bumped forward one sector. When multiple sectors on a track are defective, the controller maps the entire track to a spare at the end of the disk. The main difference between an ESDI controller's defect handling and that of an SMD controller is where the intelligence resides. For an SMD disk, all the mappings are handled in software and `diag` must explicitly write the sector headers to perform a slip. For ESDI disks, the software simply tells the controller which sectors are defective and the controller handles all the the physical details.

6.10. Starting `diag`

`Diag` is a standalone program that lives in `stand/diag`. You must boot it standalone (without UNIX) from the system monitor. After booting, you must then configure `diag` to match your controller/disk configuration. This section provides instructions for both.

User Interface

`Diag` responds to commands typed on the workstation keyboard. It contains a number of subsystems with different command sets. The prompts for each subsystem are different, too; the normal prompt is:

```
diag>
```

When you are in the `format` subsystem, the prompt changes to:

```
format>
```

to remind you where you are.

A couple of tricks make `diag` easier to use:

Restart
: If you exit from `diag` to the monitor, you can restart it with the monitor command:

```
> g4000
```

Command Shortening
: You can shorten most `diag` commands to a single letter, or to the shortest unique string of letters. For example, to enter the `translate` command, just type:

```
diag> t
```

Booting `diag`

`Diag` boots from the PROM monitor's command interpreter. To activate the command interpreter, power-on or reset the system. If UNIX is active, bring it down properly (as described in previous chapters) before resetting the system. When this message appears:

```
Auto-boot in progress
```

abort booting and enter the prom monitor. To do this, type [L1-A] on Sun-2 or Sun-3 keyboards (while holding down `L1`, press the `A` key), or type [BREAK] on non-Sun and Model 100U keyboards. The monitor should respond by interrupting the UNIX bootstrap and displaying its prompt:

```
>
```

To boot `diag` from the monitor, enter:

```
>b path_name
```

where *path_name* is the path to, and the name of the file to be booted. From a formatted, operational disk, use:

```
>b stand/diag
Boot: disk(0,0,0)stand/diag
Load: disk(0,0,0)boot
Boot: disk(0,0,0)stand/diag
Size: 34816+20480+1160 bytes          [varies by version]

where disk =
xy for a Xylogics controller,
sd for a SCSI controller,
ip for an Interphase controller, or
ec for 3Com, ie for Sun ethernet boards or le
  for Lance ethernet boards.
```

Booting `diag` from tape requires loading the boot block from tape, then loading `diag`. Normally `diag` is the fourth file on the tape (file 3 because numbering starts at 0).

```
>b tape
Boot: tape(0,0,0)
Boot: tape(0,0,3)
Size: 34816+20480+1160 bytes      [varies by version]
```

where *tape* is **mt** for 1/2" tape with Tapemaster controller, **xt** for 1/2" tape with Xylogics controller, **ar** for 1/4" Archive tape, and **st** for 1/4" SCSI tape.

When `diag` first starts up, it displays a sign on message:

```
Version 2.2 84/08/10                  [varies with different versions]
Disk Initialization and Diagnosis

When asked if you are sure, respond with 'y' or 'Y'
```

Earlier versions of `diag` may not display the version message. These are outdated; use the most recent version you have.

Configuring *diag*

When `diag` starts, it automatically enters the `diag` command subsystem. It prompts for required hardware-specific configuration information, then returns to the command level. To change the configuration later from the command level, enter the command `diag`.

`Diag` needs detailed information about the disk and controller it is working with. It has information about standard configurations built in, and it has a facility for accepting information about a non-standard disk and controller combination.

The configuration procedure differs, depending on whether the system is being configured for a standard SMD or SCSI interface, or something non-standard.

In the following command descriptions, where the example or explanation applies to both types of disk, it clearly says so. Ignore sections describing an interface type other than yours.

Configuring for Standard Disks

First, `diag` asks what type of disk controller it's dealing with:

```
specify controller:
          0 - Interphase SMD-2180
          1 - Xylogics 440 (prom set 926)[1]
          2 - Xylogics 450/451
          3 - Adaptec ACB 4000 - SCSI/ST506
          4 - Emulex MD21 - SCSI/EDSI
which one? 2
```

This example shows a Xylogics 450/451 (type 2) disk controller.

Next, `diag` asks for the controller's bus address. After you provide an address (see the table below for defaults) it echoes back the address provided:

```
Specify controller bus address in hex: address from table
Device address: address you selected
```

Table 6-2 *Controller/Host Adapter Bus Addresses*

Controller Type	*MULTIBUS ADDRESS*		*VMEbus ADDRESS*	
	1st cntrlr	*2nd cntrlr*	*1st cntrlr*	*2nd cntrlr*
Xylogics	ee40	ee48	ee40	ee48
SCSI (Sun-2)	80000	84000	ee2800	N/A
SCSI (Sun-3)	N/A	N/A	200000	N/A
SCSI (Sun-3/50)			140000	(on board)

If you specify a controller which interfaces to a SCSI disk, `diag` then asks for the controller's SCSI bus unit number:

```
Which target? 0
```

0 is the target number for the first (or only) SCSI disk controller on a host adapter; **1** is the target number for the second.

Next `diag` requests the physical unit number of the disk on the controller:

```
Which unit? 0
```

0 is the correct response for the first (or only) disk drive connected to the

[1] Sun Microsystems no longer officially supports Xylogics 440 controllers.

selected controller; **1** is the correct response for the second, and so on. SMD disks can be set for unit numbers of 0 to 3 while SCSI disks are either unit 0 or unit 1.

Next `diag` displays a menu of the different disks for which it has built-in configuration information, and asks for the disk drive type. If you select one of these (except for `other`), `diag` prints some physical data about that disk, including the number of data cylinders, number of alternate cylinders, number of heads, and number of sectors per track. If you select `other`, it prompts for this information. It then initializes the controller, and for SMD controllers issues a `status` command to the device, then displays results. This works like the `status` command issued to the `diag>` prompt.

The following example shows a Fujitsu M2312K (84–Mbyte unformatted) disk controlled by a Xylogics 450 controller with Revision C PROMs.

```
Specify drive:
          0 - Fujitsu-M2312K
          1 - Fujitsu-M2284/M2322
          2 - Fujitsu-M2351 Eagle
          3 - Fujitsu-M2333
          4 - Fujitsu-M2361 Eagle
          5 - CDC EMD 9720
          6 - Other
which one? 0
ncyl 586 acyl 3 nhead 7 nsect 32
status: ready
drive status: ready
Xylogics PROM Rev 'C'
```

If you have an older Xylogics board with Rev A PROMs, `diag` issues a warning message stating that these boards should be upgraded. They perform certain operations incorrectly, which causes the software to be unable to detect some ECC errors.

Now `diag` has the information it needs about the disk. It returns to the command mode and displays its prompt.

```
diag>
```

If the sequence fails before this point, check the hardware cabling and the information already given. Then use the `diag` command to reenter the data, or reboot `diag`.

Other Disks

If you select the *other* drive type, `diag` asks for information about the drive. Once you have defined the new drive type, `diag` adds this entry to the list of drive types, enabling you to use it over and over. `Diag` allows you to define a total of 4 `other` drive types.

NOTE *Consult the disk drive manual for information about* `other` *disk drives. This manual should be supplied by the disk drive manufacturer, not by Sun.*

The following two examples show how to configure an `other` disk drive for an SMD, then a SCSI controller.

The first example shows questions asked when using a Xylogics 450 controller for an imaginary SMD disk called a "Bogus-M1234". The second example shows the same for an imaginary SCSI ST506 disk called a "Phony-4321".

Suppose that the manufacturer's manual says that the Bogus 1234 has 823 cylinders, 10 heads, and 33 sectors/track, with 600 bytes/sector. For compatibility with Sun supported disks, you must allocate at least two alternate cylinders. These are necessary to store the defect information on the disk. Also, to leave a spare sector per track for slip-sectoring, tell `diag` there is one less data sector per track than there really is.

```
Specify drive:
          0 - Fujitsu-M2312K
          1 - Fujitsu-M2284/M2322
          2 - Fujitsu-M2351 Eagle
          3 - Other
which one? 3
Note: # of data cylinders must be at least 2 less
than the physical number of cylinders.
# of data cylinders? 820                    [total cyls minus alts]
# of alternate cylinders (min 2)? 2         [for bad sec forwarding]
# of bytes/sector (incl overhead)? 600      [for defect calculations]
first head? (usually 0, 2 for Lark fixed) 0
physical partition? (usually 0, 1 for Lark cartridge) 0
# of heads? 10
drive type? 3                               [Xylogics only – 0 to 3]
ASCII identification? Bogus-M1234           [Used for labeling disk]
                                            [same as partition cmd]
# of data sectors/track? 32                [one less for slipping]
interleave factor? 1
ncyl 820 acyl 3 nhead 10 nsect 32 interleave 1
status: ready
drive status: ready
Xylogics PROM Rev 'C'
```

NOTE *On the Xylogics 450/451 controllers, all disks with the same drive type must be the same type of drive. This is straightforward in the above example; the Bogus-M1234 is assigned drive type 3. To add a second* `other` *different from the Bogus-M1234 you would have to assign it a different drive type. If you do this, be sure you do not assign it the drive type of a disk that is or will be connected to that controller. For more on drive type numbers, see the command* `rhdr`.

`Diag` asks different questions for a SCSI controller. The following example shows how to configure a Phony-4321 with 578 cylinders, 5 heads, and 17 sectors per track, for use with the Adaptec ACB 4000 controller. The drive manual recommends using a buffered seek of 2 and write precomp starting at cylinder 0.

```
Specify drive:
          0 - Micropolis-1304
          1 - Micropolis-1325
          2 - Maxtor-1050
          3 - Fujitsu-2243AS
          4 - Vertex-185
          5 - Other
which one? 5
Note: # of data cylinders must be at least 2 less
than the physical number of cylinders.
# of data cylinders? 576                    [total minus alternates]
# of alternate cylinders (min 2)? 2         [bad sector spaces]
buffered seek? (usually 2) 2                [from drive manual]
cyl # to start write precomp? 0             [from drive manual]
# of heads? 5
ASCII identification? Phony-4321            [For labeling disk]
# of data sectors/track? 17                 [sectors per track]
interleave factor? 1
ncyl 576 acyl 2 nhead 5 nsect 17 interleave 1
```

NOTE *The menu for an ESDI disk is slightly different. Because the Emulex controller requires 4 cylinders of each disk for internal use, the number of data cylinders must be at least 6 less than the physical numer of cylinders. Also, the questions concerning buffered seek and write precomp are not asked for ESDI disks, since they don't apply.*

6.11. Preparing a New Disk

This section describes how to use `diag` to prepare new SCSI and SMD disks for operation. This discussion assumes that `diag` is loaded and configured as described earlier.

This process includes formatting the disk, performing surface analysis, then writing a label on the disk. Formatting divides the disk into blocks and sectors, surface analysis checks for and repairs surface defects, and labeling writes important information on the disk, including the disk name and the partition map.

The partition table defines the disk partition boundaries that UNIX uses for file system boundaries. You should have ready the information you will need to build the partition table when you start this procedure. If you are going to use one of `diag`'s built-in partition tables, this is no problem, but if you are going to use any other partition table, see the instructions for the `partition` command later in this chapter.

NOTE *If you are installing a new system, we recommend that you use the instructions in Installing UNIX on the Sun Workstation to prepare your disks.*

`diag` provides 2 different format subsystems; one for SCSI disks and one for SMD disks. This section discusses each separately.

SCSI Disks

The following procedure provides instructions for testing, formatting, and labeling a SCSI disk. Use the following procedure to format and label a disk:

1. Load and boot `diag`.
2. Type **`format`** to access `diag`'s SCSI format subprogram (`diag` knows it's dealing with a SCSI disk from information provided during configuration). It displays its prompt:

```
diag> format
SCSI format.
format>
```

2a. If a new disk is unformatted or the defect list has been corrupted, `diag` prints a message saying it was unable to read the defect list off the disk. If this happens, you must format the disk using the `f` subcommand:

```
diag> format
SCSI format.
no defect list found
format> f
formatting...
```

To see a list of all the SCSI format subcommands, type:

```
format> ?

SCSI Format Subcommands:
   f: format disk
   p: print defect lists
   a: add defect to physical list
   b: bias added defects
   c: clear defect lists
   d: delete defect from lists
   s: surface analysis
   r: reassign logical block
   t: translate logical block # to physical
   q: quit format
```

Before going on, it is worth understanding SCSI defect lists in detail. There are several types of defect lists, defined below.

manufacturer's defect list
: This is the list of defects supplied by the drive's manufacturer. It is always supplied in hardcopy form with each drive. On ESDI disks, this list is also stored on the disk itself, so the controller can read it. These defects are always in physical format (see below).

grown defect list
: This is the list of all defects added by the user because they were not on the manufacturer's defect list. These defects may be in physical or logical

format (see below). On ESDI disks, this list is also stored on the disk, so the controller can read it. The sum of the grown list and the manufacturer's list always equals the total list of known defects.

physical defect list
: This is the list of defects in physical format, including the manufacturer's defect list, and grown defects entered by the `add` command. For ST506 drives, this is the total list of known defects. `Diag` displays this list when given the `print` command. `Diag` keeps a copy of this list on the alternate cylinders.

reassigned block list
: This is the list of defects in logical format. This list is nonzero only on ESDI disks. It contains grown defects entered by the `reassign` command. The sum of the reassigned list and the physical list equals the total list of known defects for ESDI drives. `Diag` displays this list when given the `print` command. `Diag` keeps a copy of this list on the alternate cylinders.

Each time the SCSI `format` subcommand is entered, `diag` reads the physical and reassigned lists off the alternate cylinders into memory. If the `reassign` command is used to add a sector to the reassigned list (ESDI only), the list is immediately updated and the new copy written to disk. However, any other changes made to the lists in memory do not take effect unless you format the disk. Thus, if you change the defect list then leave the `format` subcommand without formatting the drive, those changes are lost forever. The `reassign` command is the only exception to this rule.

If the defect lists on the alternate cylinders are destroyed for some reason, the necessary action depends on the type of drive. For ST506 disks, you will have to reenter all the defects using the `add` command, then reformat the drive. For ESDI disks, it is easier. Since the controller stores copies of the manufacturer's and grown lists on the disk itself, `diag` simply asks the controller for these lists and then rebuilds the lists on the alternate cylinders. The only effect is that defects on the reassigned block list will appear on the physical defect list. This is because the controller stores all defects in physical format. In rare instances, the controller may be unable to find its lists, and the defects may have to be reentered by hand.

3. Check to make sure that the disk defect list written on the disk matches the hardcopy list shipped with your drive. Enter the command `p` to display it on the screen:

NOTE *The location of the hardcopy list depends on the workstation type. It is usually taped to the front or top of the pedestal. If you don't see it or have misplaced it during unpacking, look for a second copy taped to the disk drive housing inside the pedestal; to find it, consult the appropriate hardware manual.* **Be sure to replace this second copy when you have used it.** *Also note that on a some drives, the hardcopy list groups defects by head; the cylinder and bytes from index numbers appear in the CYL and BI columns. On others, the numbers appear in the CYL, H, and BYTE columns respectively.*

```
format> p
Defect list - Physical Format
Defect  Cylinder  Head   Bytes from Index
  N         NN       N          NNNN
etc.
```

4. Make sure this either matches the hardcopy disk defect list. It may have more defects on it, but not fewer.

   a. If they match, continue with step 4.

   b. If the hardcopy list shows defects that are not displayed on the screen list, use the a command to add to the list in memory (it will get written on the disk when you format it):

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

   c. If you cannot read the list from the disk, use the a command as shown above to type in the entire hardcopy list. The only exception to this is with a brand new ESDI drive. Even though the defect list cannot be read off the disk, the controller can get at it. Simply format the drive; the controller automatically uses the built-in defect list. After the format, `diag` extracts the defect lists from the controller and stores them on the alternate cylinders.

   d. After making any changes to the list on the disk, use the p command to display the changes on the screen. Check again to make sure the copy on the screen matches the hardcopy list.

5. When you have verified the defect listing, format the disk with the format (f) command. After you type this command, the system displays a warning, then asks for confirmation:

```
format> format
DISK FORMAT - DESTROYS ALL DISK DATA!
are you sure? y
Formatting...
```

The formatting process takes three minutes or more. At the end, you should see a message:

```
done.
```

For ESDI drives, you may also see a message about reassigning grown defects; this is OK. If you see any other message (`SCSI reset`, for example), the

formatting process did not succeed, and the defect list was not recorded on the disk. **You must format the disk again.**

6. When you have successfully formatted the disk, do a surface analysis using the (s) sub-command. This analyzes the entire disk surface, then displays a list of any defective sectors it found. For a new disk, you should ask for five surface analysis passes:

```
format> s
# of surface analysis passes (5 is usual)? 5
```

Five passes may take an hour or more to complete. When it's done, it displays a message to announce that fact.

7. For ESDI disks, `diag` automatically reassigns any defective sectors it finds. It also adds these sectors to the defect lists immediately, so there is no need to format the disk again. If all the defective sectors were successfully reassigned, you can proceed with the next step. If one of the reassignments failed for any reason, you should reassign that sector by hand, then rerun the surface analysis to make sure the disk is clean. For ST506 disks, `diag` simply reports the bad sectors it found and adds them to the defect list in memory. When it's done, it tells 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**:

   a. Reformat the disk (go back to step 4).

   b. Continue this loop until the surface analysis reports that it found no bad sectors. Once the surface analysis completes without finding any bad sectors, the format is successful.

8. Enter **q** to exit the format subsystem:

```
format> q
diag>
```

A system may have more then one controller, and a controller may have more than one disk. The `format` process described above must be repeated for every controller and every disk.

NOTE *The following 'label' operation writes a partition map on your disk. If you do not have a custom partition made, or you do not intend to use the default, you may continue, but note that you will change the partition map later.*

9. Type **label** to the `diag` prompt.

```
diag> label
```

10. `diag` now asks if you want to use the built-in partition map, then asks for confirmation:

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

11. After labeling the disk, `diag` automatically verifies the label it has just written. The following example shows the verify for a Micropolis 1325:

```
[ This is an example only; do not enter this information. ]
verify label
id: <Micropolis 1325 cyl 1022 alt 2 hd 8 sec 17 interlv 1>
      Partition a: starting cyl=0, # blocks=15912    [#'s vary]
      Partition b: starting cyl=117, # blocks=33456
      Partition c: starting cyl=0, # blocks=138448
      Partition g: starting cyl=363, # blocks=89080
diag>
```

12. If you confirm, `diag` partitions your disk according to the default maps shown below, then exits.

The following table shows the default partitions for Sun-supplied SCSI disks:[2]

Table 6-3 *Default Partition Sizes for SCSI Disk Subsystems*

SCSI Disk	*Raw*	*Partition Sizes (MBytes)*							
		"a"	"b"	"c"	"d"	"e"	"f"	"g"	"h"
Micropolis 1304	50	8.1	8.4	43.1	*unused*	*unused*	*unused*	26.5	*unused*
Micropolis 1325	85	8.1	17.1	70.9	*unused*	*unused*	*unused*	45.6	*unused*
Maxtor XT-1050	50	8.1	8.4	44.4	*unused*	*unused*	*unused*	27.9	*unused*
Fujitsu M2243AS	86	8.1	17.1	70.8	*unused*	*unused*	*unused*	45.6	*unused*
Vertex V185	85	8.1	17.1	70.9	*unused*	*unused*	*unused*	45.5	*unused*
Micropolis 1355	170	8.1	17.1	141.7	*unused*	*unused*	*unused*	116.5	*unused*
Toshiba MK156F	170	8.1	17.1	141.8	*unused*	*unused*	*unused*	116.6	*unused*

[2] Note that the numbers in this table are approximate: formatted capacity depends on the type of controller. Also, note that a 'Megabyte' of disk capacity is defined as one million bytes, and that UNIX file storage capacity is substantially smaller.

SMD Disks

This section provides instructions for formatting, checking, and labeling an SMD disk. It assumes that any slippings and mappings already on the disk should be left intact. If this is not the case, the `sformat` command described at the end of the manual should be used.

NOTE *You must have a partition map ready before beginning this procedure.*

1. Enter the command `format`:

```
diag> format
DISK FORMAT - DESTROYS ALL DISK DATA!
are you sure? y
Formatting
NNN
   ...
   [messages about slippings and mappings]
   ...
Verifying
NNN
total defects N with N original defects from manufacturer
diag>
```

During the formatting and verifying phases, `diag` displays the number of each track it completes. Now continue to step 2.

NOTE *From here on, this procedure assumes you have are either going to use a default partition table or you have written or modified one using the* `partition` *command. The default partitions are designed for standalone use only; if you are installing an NFS server, or have any other reason for not using the default partition, construct or modify a partition with the* `partition` *command, then return here.*

2. Enter the command `label`:

```
diag> label
```

`Diag` now asks if you want to use default partition map, then asks for confirmation before proceeding:

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

NOTE `diag` *remembers if you recently used the partition command; if so, it uses the last partition you accessed in place of 'disk type' above. If you just created a custom partition called 'xyz', it asks:*

```
OK to use logical partition map xyz?
```

3. After labeling the disk, `diag` automatically verifies the label it has just written. For 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
        Partition b: starting cyl=50, # blocks=33440
        Partition c: starting cyl=0, # blocks=262720
        Partition g: starting cyl=155, # blocks=213120
diag>
```

The following table shows the default partitions:[3]

Table 6-4 *Default Partition Sizes for SMD Disk Subsystems*

SMD Disk	*Raw*	*Partition Sizes (MBytes)*							
		"a"	"b"	"c"	"d"	"e"	"f"	"g"	"h"
Fujitsu 2312K (8")	84	8.1	17.1	67.3	*unused*	*unused*	*unused*	42.0	*unused*
Fujitsu 2284 (14")	169	8.1	17.1	134.5	*unused*	*unused*	*unused*	109.1	*unused*
Fujitsu 2322 (8")	168	8.1	17.1	134.5	*unused*	*unused*	*unused*	109.1	*unused*
Fujitsu 2351 Eagle	474	8.1	17.1	395.7	*unused*	*unused*	*unused*	369.8	*unused*
Fujitsu 2333 (8")	337	8.2	17.2	281.6	*unused*	*unused*	*unused*	256.3	*unused*
Fujitsu 2361 Eagle	690	8.2	17.2	576.3	*unused*	*unused*	*unused*	550.9	*unused*
CDC 9720 (8")	347	8.3	17.2	281.9	*unused*	*unused*	*unused*	256.3	*unused*

6.12. Troubleshooting With Diag

This section shows methods for using `diag` to deal with disk problems.

Usually, you find out you have disk problems when you see a UNIX error report. This should provide the file system name, and a disk block number in partition-relative and absolute form. For example:

```
Sd0g read error block xxx abs block nnn
```

To deal with this error, you must discover and repair the cause of the error and repair the surface of the disk.

This section provides the following procedures:

Fixing a Bad Sector (SCSI) — Describes how to use the information in a UNIX error report to add a sector to the bad sector list. (SCSI only.)

[3] Note that the numbers in this table are approximate: formatted capacity depends on the type of controller. Also, note that a 'Megabyte' of disk capacity is defined as one million bytes.

Checking and Fixing a Bad Sector (SMD) — Describes how to go to the sector indicated by a UNIX error report, determine correct action, and repair the defect. This procedure provides a reasonable chance of salvaging most of the data on the defective sector.

Electronic Problems — Describes how to set up `diag` to test the swapping area of the disk, so you can perform electronic troubleshooting while using `diag` to check the disk functionality without destroying critical data on the disk.

Checking and Fixing a Bad Sector (SCSI)

Use these procedures to add a bad sector reported by UNIX to the bad sector list on the disk.

ST506 Controllers

1. Obtain the absolute block number (*nnn*) from the UNIX error message.
2. Take full dumps of your entire system.

NOTE *Because of the nature of the ST506 interface, repairing any surface defect requires reformatting the disk, which erases all the data on it. To ensure the safety of your data, take a full dump before starting any ST506 disk repair.*

3. Boot and configure `diag` as described earlier in this chapter.
4. Do a `read` of the area to obtain the exact address of the failed block. Start the `read` a few sectors before the one reported to be sure you cover the bad sector (NNN = nnn-20 in the example):

```
diag> read
starting block? NNN
# of blocks? 40
increment? 1
# of blocks per transfer? 1
NNN/NN/NN

And if it fails...
NNN/NN/NN Read failed, cyl=NNN, head = NN,
sector = NN scb: N check
[A few lines of other information]
```

5. Enter the `format` subsystem:

```
diag> format
SCSI format.
format>
```

6. Use the translate command to obtain bytes from index:

```
format> t
logical block number?  NNN/NN/NN
cyl NNN  head NN bfi NNNN (physical)
```

7. Use the format subsystem `add` command to add this data to the list:

```
format> a
cylinder? NNN
head? NN
bytes from index? NNNN  [use data from above!]
```

8. Now format the disk:

```
format> format
DISK FORMAT - DESTROYS ALL DISK DATA!
are you sure? y
Formatting... done              [Takes a while]
format>
```

9. Restore the system from the dump tape made earlier.

ESDI Controllers

1. Obtain the absolute block number (*nnn*) from the UNIX error message.
2. Boot and configure `diag` as described earlier in this chapter.
3. Do a `read` of the area to obtain the exact address of the failed block. Start the `read` a few sectors before the one reported to be sure you cover the bad sector (NNN = nnn-20 in the example):

```
diag> read
starting block? NNN
# of blocks? 40
increment? 1
# of blocks per transfer? 1
NNN/NN/NN

And if it fails...
NNN/NN/NN Read failed, cyl=NNN, head = NN,
sector = NN scb: N check
[A few lines of other information]
```

4. Enter the `format` subsystem:

```
diag> format
SCSI format.
format>
```

5. Use the subcommand `reassign` to mark the sector defective:

```
format> r
logical block address? NNN
logical block NNN (0x HHH) reassigned
format>
```

Checking and Fixing a Bad Sector (SMD)

To repair a bad sector on an SMD disk, use the following procedure:

1. Obtain the absolute block number (*nnn*) from the UNIX error message.
2. Boot and configure `diag` as described earlier.

NOTE *It's a good idea to toggle the 'errors' flag so you can see retrys. Sometimes it is the only way to spot the error. To toggle the flag, enter:*

```
diag> e.
```

3. Do a `read` of the area to obtain the exact address of the failed block. Start the `read` a few sectors before the sector reported, to be sure you cover the bad sector (NNN = nnn-20 in the example):

```
diag> read
read
Starting block? NNN
number of blocks? 40
Increment? 1
# of blocks per transfer? 1
CC/HH/SS diag>
```

Or, if you have an error:

```
Read failed #6, CRC/hard ECC error, cyl=nn head=nn sector=dd
```

Note: above message is example only. If you have 'errors' flag set for retrys, a similar message may appear several times.

4. If your disk is configured for slipping, attempt to slip the bad sector:

```
diag> slip
slip sector
slipping may be removed only by complete format of the disk
cylinder number? NN
track number? NNN
logical sector to be slipped? NN
Attempt to preserve data? y
OK to attempt slip of logical sector CC/HH/SS? y
diag>
```

5. If the `slip` does not succeed, the following buffer shuffle gives you a fairly good chance to save your data:

a. Use the `map` command as if to *map* the defective sector. When it asks if you want to save the data, say yes:

```
Attempt to save data? y
```

b. When it asks if you want to actually write in the mapping, say no:

```
OK to map this sector? n
```

c. Now do a `write` to the defective sector. Note that because `diag` still has the "saved" data in its buffers, it will restore the original data to the bad sector.

CAUTION **To avoid losing data, be sure to write to only the one sector that you used the `map` command on.**

```
diag> write
write
starting block? NNN
# of blocks? 1
increment? 1
# of blocks per transfer? 1
CC/HH/SS diag>
```

6. Now repeat the `read` from above:

```
diag> read
read
Starting block? NNN
number of blocks? 40
Increment? 1
# of blocks per transfer? 1
CC/HH/NN diag>
```

7. If the sector checks out OK, the problem is solved. You may want to repeat the `read` a few times to make sure the sector is not marginal. If the read still reports an error, repeat the `map` done earlier, only when it asks for permission to map the sector, answer **`yes`**.

Electronic Problems

Many disk errors are caused by problems with the disk cables or the electronics. These problems tend to generate multiple disk error messages at random locations.

The following procedure shows how to set up `diag` to "bang" on the swap area of the disk. This is where the most expendable data lives; destroying the data here should cause minimum harm. This procedure does not describe how to troubleshoot the electronics; it only describes how to set up `diag` so that the

electronics troubleshooting causes the least damage.

For electronics troubleshooting instructions, see the appropriate field service manual.

1. Start up and configure `diag` for the disk with the problem.
2. Enter the `verify` command. For a Micropolis 1304, the display goes:

```
diag> v
verify label
Micropolis 1304 cyl = 824, alt 5, hd 6, sec 17>
partition a: starting cyl # 0      # blocks 15884
partition b: starting cyl # 156    # blocks 16442
partition d: starting cyl # 0      # blocks 84150
parititon g: starting cyl # 317    # blocks 51856
```

On almost all disks, the swap space is *partition b*. To make `diag` do reads within partition b, request a `read` using a cylinder within *partition b*. For example, enter the following:

CAUTION **To avoid damaging data, make sure you really hit the swap space. Be sure a) you don't leave the slash off the cylinder number, b) the starting cylinder is within partition b, and c) the number of blocks does not lead into the next partition (g in this example).**

```
diag> r                    [This is an example only]
read
starting block? 156/              [Don't forget the "/"!]
# of blocks? 16442          [Or fewer blocks to be safe]
increment ? 1
# of blocks per transfer ? 1
NNN/NN/NN
```

NOTE *In this example,* `diag` *will do reads of the entire swap space block by block. If you wish to make it read the same block over and over, use an increment of 0.*

6.13. Command List

The following list shows the `diag` subsystems, commands, and subcommands. Note that subsystems provide access to subcommands, while commands initiate action from the top layer of `diag`. Subprograms are routines called by subsystems.

For convenience, the commands are divided into several categories. These are:

Toggle flags and options
Miscellaneous commands
Perform some test
Do something complicated and interactive to the disk.

Toggle Flags and Options

All the flags and option toggles work similarly; the option is either ON or OFF, and calling the following commands cause it to switch states and display its new value:

`error` — When ON, displays messages for every retry; when OFF, displays a message only after all retries are done. (Most operations retry 3 times; formatting retries 1 time). (default = OFF).

`info` — When ON, provides verbose messages for every operation performed (default = OFF).

`time` — When ON, displays timing messages for formatting, read, and write operations (default = OFF).

`slipmsgs` — When ON, provides before and after dumps of track headers when a sector is slipped (default = OFF).

`formatmsgs` — When ON, if a corrected error occurs during disk formatting, a message is displayed; when OFF, corrected formatting errors are not reported (default = OFF).

`mapcheck` — When ON, if an error occurs during a `read, write, position` or `test` command with an SMD disk, it reads the map table from the disk, and checks to see if any sector in the range tested has been mapped. If so, it reports this fact. Since `diag` reports errors when trying to read a mapped sector, the mapcheck message may explain an otherwise mysterious error report (default = ON).

Miscellaneous Commands

Use the following commands to help you use `diag`:

`help` or `?` — Display current list of commands available.

`quit` — Exit `diag`.

`clear` — Clear drive faults.

`status` — Fetch and display current controller and drive status.

`diag` — Reset configuration information (described earlier in this manual).

`translate` — Take a disk address and display it in CC/HH/SS, decimal, and hex, translated for the currently configured disk.

addition and subtraction — Entering a + or − on the command line causes a prompt asking you for two numbers (to be added or subtracted). `Diag` provides the result in decimal, hex, and cylinder/head/sector form, translated for the currently configured disk.

`version` — Displays all the *sccs* identifiers in the program.

`verify` — Reads the labels on the disk, prints out the partition map, and if necessary, asks if you want to restore the primary label.

Tests

The following commands perform non-interactive tests:

`position` — This non-destructive test checks the ability to seek and read sectors. It selects and reads single sectors at random, continuing until the user types ^C. If it encounters an error, if `mapcheck` is *on*, it checks to see if the sector is mapped; if so, it reports that fact.

`test` — This **destructive** test writes, then reads groups of sectors continuously, testing for ability to seek, write, and read. It selects sectors at random, and tests a block of sectors starting there. It prompts for the size of the block, and it continues until the user enters ^C. If it finds an error, and if `mapcheck` is ON, it checks the map to see if any sectors in the group are mapped. If so, it reports that fact.

`seek` — This non-destructive test can do an hourglass seek over all cylinders, then report the time it took. It can also continuously seek between two given blocks.

`read/write` — This test checks for ability to read and write data. Write is **destructive** but read is not.

`dmatest` (Xylogics 450/451 controller only) — This test checks the Xylogics controllers using BUFLOAD and BUFDUMP commands. If `abortdma` is ON, it exits when it finds an error; otherwise it continues until the user types ^C.

Complicated, Interactive Commands

The following commands provide significant interaction with the disk. They are described in detail, later.

`map` (SMD disks only). This command enables you to read the current list of mapped sectors, and to (optionally) add a new sector to that list.

`fix` (SMD disks only). This command reformats and verifies a given range of the disk.

`slip` (Xylogics 450/451 only). This command allows you to manually slip individual sectors on a Xylogics-controlled disk with slip-sectoring enabled. It prompts to see if you want to attempt to save the data in the sector being slipped.

`rhdr` (read headers — Xylogics 450/451 only). This is among the most informative commands; it displays individual sector headers sequentially. Use it to help identify slipped or mapped sectors, to show whether the disk is configured for sector slipping, and to help spot anomalies in sector headers.

`label` (write label on disk) — This command writes a label on the disk.

`partition` (set partition table) — This command allows you to set the partition table boundaries.

`scan` — This **destructive** command does repeated sector scans over a specified range of the disk. On SMD disks, it can automatically map or slip any bad sectors it finds. Use this command after formatting the disk, but before installing UNIX, to check for bad sectors that didn't show up on other tests.

`whdr` (Xylogics 450/451 only) — This command reads in existing sector headers starting at a specified location, then loops while asking if you want to change a header. **SPECIAL USES ONLY.**

`format` — `diag` actually provides a SCSI format subsystem with its own `format` subcommand, and an SMD subsystem with its own `format` subcommand. Format subsystems are where you go to perform formatting related operations, and the format subcommands actually cause the format to occur.

`sformat` — For SMD only. A special formatting subprogram for manipulating the defect list on SMD disks. It allows you to enter the manufacturer's defect list manually, if the list has been corrupted. It also allows you to format the disk using only the original manufacturer's defects. This is useful if sectors were errantly slipped or mapped, and you wish to restore them to normal.

`format` Command

Formatting a disk describes the process of dividing it into sectors so UNIX can use it. `diag` provides separate format subsystems for both SMD and SCSI disks; these are invoked by entering the command `format` to the `diag` prompt. `diag` selects the proper subsystem based on configuration information.

Once within either format subsystem, the prompt changes to:

```
format>
```

Both format subsystems provide `format` commands; these actually perform the format operation to the disk. For instructions to use the `format` commands, see the earlier part of this chapter.

`map` Command

The `map` command (SMD disks only) displays the current map table and allows you to add a new mapping.

CAUTION **This command destroys disk data if you add a new mapping.** Backup disk data before proceeding.

After you enter `map`, it displays the current map table, then prompts for additional information. When it asks if you want to add a mapping, if you enter `n`, it exits without changing anything. If you enter `y` to add a mapping, it asks if you want to attempt to preserve the data. If you answer `y`, it writes the data to the alternate sector before the mapping is actually done. A typical session goes:

```
diag> map
Current mapping:
Sector CC/HH/SS mapped to CC/HH/SS
Sector CC/HH/SS mapped to CC/HH/SS
Do you wish to add a mapping? y
mapping may be removed only by complete format of the disk
cylinder to be mapped? 704
track to be mapped? 9
sector to be mapped? 7
Attempt to preserve data? y
Data transfer successful!
OK to map 704/9/7? y
mapped sector 704/7/7 to 822/8/31
```

After it maps the sector, map adds the new sector to the map, marks the old sector bad, and rewrites the new map table.

If a read error occurs during attempt to preserve data, map reports the transfer unsuccessful. This may be only partially true; with fixed ECC errors, the data is still intact. With a hard CRC error, some data may survive, but with other errors the data is usually lost.

You can map a sector on a disk set up for slip sectoring, but slipping a bad sector is preferable to mapping it.

fix Command

The fix command formats and verifies user-specified sections of SMD disks. Use it to verify a section of the disk without doing the entire thing.

CAUTION **This command destroys disk data. Backup disk data before proceeding.**

Fix requires a starting and ending track address. It also requires a number of surface analysis passes. After it obtains this information, it asks for permission to continue.

A typical session might go:

```
diag> fix
fix -- DESTROYS SOME DISK DATA
Warning! use 'format' command when fixing the whole disk.
formats a range of tracks
enter track number as 'cyl/track'
starting track? 15/20
ending track? 15/30
# of surface analysis passes (5 recommended)? 5
OK to format from 15/20/0 to 15/29/31? y
CC/HH
diag>
```

CC/HH represents the current track number. It increments as the test proceeds.

If you answer `y`, it proceeds like a format within the specified boundaries, except that it prints cylinder/head numbers instead of just cylinder numbers. If it encounters a previously slipped sector, it reslips the sector after the track is formatted, and displays a message announcing this fact. It reports mapped sectors and adds them to the map if they are not already there.

Before it exits, it writes the new bad sector table on the disk then updates the mapped sector headings. If it is interrupted, it does not write the new map or update the mapped sector headers.

`slip` Command

The `slip` command allows you to manually slip a sector on an SMD disk connected to a Xylogics controller, provided the disk and the sector are eligible for sector slipping.

CAUTION **This command destroys disk data. Backup disk data before proceeding.**

`slip` asks if you want to save data; if you answer `y`, it stores the data from the entire track in a buffer, and attempts to rewrite the data after the slip. The data usually survives ECC errors; other errors may cause damage.

`rhdr` Command

The command `rhdr` (read headers) works with Xylogics controllers only. It displays the sector headers for consecutive tracks.

A display similar to the following results:

```
diag> rhdr
read track header
starting track (dd/dd): 00

0/0
sec 0  8000  8100  8200  8300  8400  8500  8600  8700
sec 8  8800  8900  8A00  8B00  8C00  8D00  8E00  8F00
sec 16 9000  9100  9200  9300  9400  9500  9600  9700
sec 24 9800  9900  9A00  9B00  9C00  9D00  9E00  9F00
stop? <ret>
0/1
sec 0  9F01  8001  8101  8201  8301  8401  8501  8601
sec 8  8701  8801  8901  8A01  8B01  8C01  8D01  8E01
sec 16 8F01  9001  9101  9201  9301  9401  9501  9601
sec 24 9700  9801  9901  9A01  9B01  9C01  9D01  9E01
stop? y
```

Reading headers can help locate anomalies in sector headers, and locate spare, mapped, slipped and runt sectors. This in turn can help identify a disk with slip sectoring. The following headers identify unusual sectors:

FFFFFFFF — Mapped sector. These headings identify mapped sectors. UNIX redirects accesses to these sectors to spare sectors as specified by the map. On a disk with slip sectoring, mapping occurs only after a second sector on a track goes bad (very rarely).

FEFEFEFE — Slipped sector. This header marks a place where a logical sector was slipped from.

DDDDDDDD — Spare data sector. Each track on a slip sectored disk starts life with a spare data sector; if a bad sector on that track is slipped, this spare gets used up.

EEEEEEEE — Runt sector. This is the designation given to the extra space at the end of a track; it is usually too short for a data sector but too long to ignore.

Use `rhdr` to help identify mapped or slipped sectors, or to discover if a disk is setup for slip sectoring. Also, if any error message identifies a track or sector as the source of a problem, use `rhdr` to check the headers in that area for consistency.

Understanding a normal header number requires translating it from hex to binary and reading the contents of the fields. The bits are:

ss00 0ccc cccc cccc ttss ssss hhhh hhhh

where:

h = head number
s = sector number (note s bits in two locations!)
t = drive type
c = cylinder number
0 = unused bits, always set to 0

The head number is straightforward; it is two hex numbers.

The sector number is divided into two fields.

The drive type is an identifier for the Xylogics controller. These are hardwired into `diag` except in case of *other* disk types. The usual assignments for these bits are:

00	Fujitsu-M2351 Eagle
01	Fujitsu-M2312K and CDC EMD 9720
10	Fujitsu-M2284/2322
11	Fujitsu-M2361 Eagle and Fujitsu-M2333

The cylinder number is also straightforward; three hex numbers where the most significant hex digit is <8 because its high-order bit is 0.

When `diag` displays numbers, it strips off leading 0s. In the above header display, the first header with all zeroes attached would be 00008000.

The following example shows the decoding of the last header number shown in the `rhdr` display above:

9E01 - as shown

00009E01 - append leading zeroes

`0000 0000 0000 0000 1001 1110 0000 0001` - translate to bir

`ss00 0ccc cccc cccc ttss ssss hhhh hhhh` - show field val

This shows it is cylinder 0, head 1, sector 0x1E, on a drive type 2 (10 binary -

Fujitsu-M2284/2322).

`label` Command

The `label` command writes a new label on the disk. It asks if you want to use the 'built in' (default) partition map for your disk, and displays a copy of what it has done.

NOTE *If you write an incorrect label on a disk, UNIX may not be able to use a filesystem. However, if you rewrite a correct label it should correct the problem.*

A typical session goes:

```
diag> label
label this disk
OK to use logical partition map disk_type? y
Are you sure you want to write? y
```

After it writes the label, it displays the label it has just written. For example, if you had a Fujitsu M2322, it would display:

```
verify label
id: <fujitsu-M2322 cyl 821 alt 2 hd 10 sec 32 interleave 1>
    Partition a: starting cyl=0, # blocks=15884
    Partition b: starting cyl=50, # blocks=33440
    Partition c: starting cyl=0, # blocks=262720
    Partition g: starting cyl=155, # blocks=213120
diag>
```

Note that the numbers in the above display differ depending on disk type.

To use a label other than the 'built in' defaults, see *partition*, and *Installing UNIX on the Sun Workstation.*

`partition` Command

The `partition` command selects a table to use when labeling the disk. `Diag` maintains default partition tables for all standard disks; it asks you to select a disk, then prints the default partition table. Then it asks you if you want to modify this table. A typical session goes:

```
diag> partition
Select partition table
   0 - Fujitsu-M2312K
   1 - Fujitsu-M2312K Old Type
   2 - Fujitsu-M2284/2322
   3 - Fujitsu-M2284/2322 Old Type
   4 - Fujitsu-M2351 Eagle
   5 - Fujitsu-M2351 Eagle Old Type
   6 - Fujitsu-M2333
   7 - Fujitsu-M2361 Eagle
   8 - CDC EMD 9720
   9 - Other
Which One? 3
Do you wish to modify this table? y
```

```
Partition a: starting cyl=0, # blocks=15884
Change this partition? n
Partition b: starting cyl=50, # blocks=33440
Change this partition? n
Partition c: starting cyl=0, # blocks=262720
Change this partition? n
Partition d: starting cyl=0, # blocks=0
Change this partition? n
Partition e: starting cyl=0, # blocks=0
Change this partition? n
Partition f: starting cyl=0, # blocks=0
Change this partition? n
Partition g: starting cyl=155, # blocks=213120
Change this partition? y
starting cylinder? 155
# of blocks? 600/0/0
Partition h: starting cyl=0, # blocks=0
Change this partition? y
starting cylinder? 755
# of blocks 66/0/0
Verify partition table 'Fujitsu-M2284/2322':
        Partition a: starting cyl=0, # blocks=15884
        Partition b: starting cyl=50, # blocks=33440
        Partition c: starting cyl=0, # blocks=262720
        Partition d: starting cyl=0, # blocks=0
        Partition e: starting cyl=0, # blocks=0
        Partition f: starting cyl=0, # blocks=0
        Partition g: starting cyl=155, # blocks=192000
        Partition h: starting cyl=755, # blocks=21120
OK to use this partition table? y
Use the label command to write out the partition table.
```

Note that if you select 'other', it asks you to name the partition table, and then it goes through the above sequence, except that all values start at 0.

scan Command

The `scan` command performs repeated sector scans over a specified range of the disk. It is typically used to find additional disk errors after the disk is formatted but before UNIX is installed.

CAUTION **This command destroys disk data. Backup disk data before proceeding.**

`scan` looks for new bad sectors and doesn't look at mapped sectors. Unlike `fix`, when it does a mapping, it writes the new mapping table to the disk immediately. It runs continuously until interrupted.

`scan` includes a number of options, all of which it prompts for. These are:

scan entire disk? — If you answer `y`, it scans the entire disk; if you answer `n`, it asks for beginning and ending addresses. If the area to scan includes primary and secondary label areas, it displays a message to this effect.

NOTE *If you overwrite the disk label, you will have to re-label the disk before use.*

use random bit patterns? — y causes it to use random patterns (better for a small intensive disk scans); n causes it to use 5 standard patterns.

perform corrections when defects are found?— If `diag` is configured for a Xylogics controller, `scan` asks if it should do corrections to bad sectors. If y, it tries to slip bad sectors and if it can't, it tries to map them. If n, it only reports newly found bad sectors. If the controller is not a Xylogics, it displays a message that it cannot fix defective sectors.

Using `scan` without fixing bad sectors is handy for tracking cable problems, or for other cases where you don't want to fix a sector every time you find an error. It is also useful for SCSI surface analysis.

A typical session might go:

```
diag> scan
scan - continuous scan for defective sectors
DESTROYS DISK DATA
scan entire disk? n
starting block? 700
ending block? 702
use random bit patterns? y
perform corrections when errors are found? y
OK to scan from 2/1/25 to 2/1/26? y
type control-C to quit
[pass 1 - bit pattern #1: 0xnnnnnnnn]
2/1/n
```

Scan continues until the user aborts with `control C`. Then it prints:

```
Command aborted
diag>
```

`whdr` Command

The `whdr` command (Xylogics 450/451 only) changes sector headers on request. This requires intimate knowledge of sector headers and should only be necessary under extreme circumstances.

CAUTION **This command destroys disk data. Backup disk data before proceeding.**

`sformat` Command

The `sformat` command is a special purpose formatting subprogram for SMD disks. It should only be used when the disk's defect list has been destroyed or contains incorrect information.

CAUTION **This command destroys disk data. Back-up disk data before proceeding.**

When you enter the `sformat` subprogram, you can choose from the following commands:

```
diag> sformat
specify format:
   1 - list defect list saved on disk
   2 - format only with original disk defects
   3 - manual entry for defect list
   4 - delete manual defect entry
   5 - change manual defect entry
   6 - exit sformat
which one?
```

`list defect list saved on disk`
This command reads the defect list off the disk and displays it. The disk must be formatted for this command to succeed.

`format only with original disk defects`
This command is similar to the normal format command, except that it erases any defects that were added by `slip` and `map`. This command is useful only if you have errantly slipped or mapped sectors and wish to restore them to normal.

`manual entry for defect list`
This command allows you to enter the manufacturer's defect list manually from the hard-copy supplied with every disk. Any existing defect list is overwritten by the one entered. After the list has been entered, the disk must be formatted with the normal `format` command before the changes are used.

`delete manual defect entry`
This command can be used to remove a defect that was incorrectly entered by hand. After deleting the defect, the disk must be formatted with the normal `format` command before the changes are used.

`change manual defect entry`
This command can be used to change a defect that was incorrectly entered by hand. After the defect has been changed, the disk must be formatted with the normal `format` command before the changes are used.

Assembly Language Reference Manual

The following pages are Errata and Addenda for Sun Release 3.0 version of the *Assembly Language Reference Manual* — Sun Part Number: 800-1372.

Preface — Errata

Replace the first sentence of paragraph three of the preface with the following sentence:

> This manual describes the syntax and usage of the `as` assembler for the Motorola MC68010 and MC68020 microprocessors, the MC68881 floating-point coprocessor, and Sun's Floating-Point Accelerator (FPA).

Introduction — Errata

Page 3

> Replace the heading before paragraph five, "`-m68010` or `-10`", with "`-mc68010`".

Page 3

> Replace the heading before paragraph four, "`-m68020` or `-20`", with "`-mc68020`".

Assembler Directives — Errata

Page 37

> Append the following text to the end of the second paragraph: "`stabn` build various types of symbol table entries."

Instructions and Addressing Modes — Errata

Page 42

Replace the definition of the *d* notation with the following:

d refers to a displacement, which is a constant expression in `as`. In 68020 mode, a length specifier (`:L`, described below) may be appended to the displacement. Any forward or external references *require* the length specifier to be `:l`. All other references permit either `:l` or `:w`.

`L` refers to the index register's length. This may be either long (`l`) or word (`w`). If the only value permitted by a particular addressing mode or category is `l` or `w`, then `L` will be replaced by the appropriate value in the table notation.

`s` refers to a scale factor that may be used to multiply the index register's length. The scale factor may have a value of 1, 2, 4, or 8.

The table notation of two or three items separated by colons, such as *ri*`:L:s`, indicate items that may be optional. In that particular case, *you may not* specify `:s` unless you have specified `:L`, which you may not specify unless you have specified *ri*. The items in the list must appear in the order given in the notation of the tables that follow.

Page 43

Replace Table 6-1 with the following table:

Table 6-5 *Addressing Modes*

Mode	*Notation*	*Example*
Register	`an,dn,sp,pc,cc,sr,usp`	`movw a3,d2`
Register Deferred	`an@`	`movw a3@,d2`
Register List	*ri*`-`*rj* `or` *ri*`/`*rj*	`movem a0-a4, a6@-`
FPA register	`fpa`*i*	`fpmoves fpa1,d2`
Floating-Point Register (MC68881 only)	`fpi`	`fmoves fp1,a3@(24)`
Postincrement	`an@+`	`movw a3@+,d2`
Predecrement	`an@-`	`movw a3@-,d2`
Displacement	`an@(`*d*`)`	`movw a3@(24),d2`
Word Index	`an@(`*d*`, `*ri*`:w)`	`movw a3@(16, d2:w),d3`
Long Index	`an@(`*d*`, `*ri*`:l)`	`movw a3@(16, d2:l),d3`
Absolute Short	*xxx*`:w`	`movw 14:w,d2`
Absolute Long	*xxx*`:l`	`movw 14:l,d2`
PC Displacement	`pc@(`*d*`)`	`movw pc@(20),d3`
PC Word Index	`pc@(`*d*`, `*ri*`:w)`	`movw pc@(14, d2:w),d3`
PC Long Index	`pc@(`*d*`, `*ri*`:l)`	`movw pc@(14, d2:l),d3`
PC-Memory Indirect Pre-Indexed (68020)	`pc@(d:L,`*ri*`:L:s)@(d:L)`	`movl pc@(2:w,d4:w:4)@(14:l),d3`
PC-Memory Indirect Post-Indexed (68020)	`pc@(d:L)@(d:L,`*ri*`:L:s)`	`movl pc@(d:l)@(3:w,d2:l:4),d3`
Memory Indirect Pre-Indexed (68020)	`an@(`*d'*`:L,`*ri*`:L:s)@(d:L)`	`movl a1@(d:l,d2:l:4)@(14:w)`
Memory Indirect Post-Indexed (68020)	`an@(`*d*`:L)@(d':L,`*ri*`:L:s)`	`movl a2@(2:w)@(14:w,d4:w:2)`
Normal	*identifier*	`movw widget,d3`
Immediate	`#`*xxx*	`movw #27+3,d3`

Page 45

Replace Table 6-2 with the following:

Table 6-6 *Addressing Categories*

Addressing Mode	*Assembler Syntax*	*Data*	*Memory*	*Control*	*Alterable*	*68020 Only*
Register Direct	`an, dn, sp, pc, cc, sr, usp`	X			X	
A Register Indirect	`an@`	X	X	X	X	
A Register Indirect with Displacement	`an@(d:L)`	X	X	X	X	X
A Register Indirect with Word Index	`an@(d:L, ri:w:s)`	X	X	X	X	X
A Register Indirect with Long Index	`an@(d:L, ri:l:s)`	X	X	X	X	X
A Register Indirect with Post Increment	`an@+`	X	X		X	
A Register Indirect with Pre Decrement	`an@-`	X	X		X	
A Register Indirect with Displacement	`an@(d)`	X	X	X	X	
A Register Indirect with Word Index	`an@(d,ri:w)`	X	X	X	X	
A Register Indirect with Long Index	`an@(d,ri:l)`	X	X	X	X	
Memory Indirect Post-Indexed	`an@(d:L)@(d:L,ri:L:s)`	X	X	X	X	X
Memory Indirect Pre-Indexed	`an@(d':L,ri:L:s)@(d:L)`	X	X	X	X	X
Absolute Short	`xxx:w`	X	X	X	X	
Absolute Long	`xxx:l`	X	X	X	X	
PC-relative	`pc@(d)`	X	X	X		
PC-Indirect with Displacement	`pc@(d:L)`	X	X	X		X
PC-relative with Word Index	`pc@(d,ri:w)`	X	X	X		
PC-Indirect with	`pc@(d:L,ri:w:s)`	X	X	X		X

Table 6-6 *Addressing Categories— Continued*

Addressing Mode	*Assembler Syntax*	*Data*	*Memory*	*Control*	*Alterable*	*68020 Only*
Word Index						
PC-relative with Long Index	`pc@(`*d*`,`*ri*`:l)`	X	X	X		
PC-Indirect with Long Index	`pc@(`*d*`:L,`*ri*`:l:s)`	X	X	X		X
PC-Memory Indirect Post-Indexed	`pc@(d:L)@(d':L,`*ri*`:L:s)`	X	X	X	X	X
PC-Memory Indirect Pre-Indexed	`pc@(d':L,`*ri*`:L:s)@(d:L)`	X	X	X	X	X
Immediate Data	*#nnn*	X	X			

The current version of `as` doesn't support base suppression.

Error Codes — Errata

Page 52

Insert the following before the description of the *Stab storage exceeded* error message.

Register out of range

In the FPA's dot product, matrix move and transpose instructions when the register specified does not fall within the specified range, then Register out of range error is reported. Note that for most instructions where one operand is an effective address, then register range is 0 to 15. If all operands are FPA registers, then register range is 0 to 31. For constant RAM registers, the range is 0 to 511. This type of error would probably also cause the Invalid operand error to be reported.

Page 53

Insert the following after the description of the *Undefined symbol* error message.

Unqualified forward reference

The displacement field in a 68020 based/indexed address mode contains an unqualified forward reference. Note that the displacement in a based/indexed address mode for 68020 instruction set can contain forward or external reference ONLY if the length specifier is present. The length specifier should be `:l` (long). This type of error would probably also cause Multiply defined symbol (Phase error).

List of `as` Opcodes — Errata

Page 57

Replace the first paragraph with the following paragraph:

This appendix is a list of the instruction mnemonics accepted by `as`, grouped alphabetically. The list is divided into two tables, the first covers the MC680x0 processor's instuctions, the second covers the MC68881 floating-point processor's instructions. (For more information about floating-point programming, see *Floating-Point*

Programmer's Guide for the Sun Workstation.)

Page 57

Replace the bulleted paragraph beginning "An instruction of the form..." with the following:

- An instruction of the form `addX` in the assembly language syntax column means that the instruction is coded as `addb`, `addw`, or `addl`, *etc.*

Pages 58-72

Replace Table B-1 with the following table:

Table 6-7 *List of MC680x0 Instruction Codes*

Mnemonic	*Operation Name*	*Syntax*	*Processor*
abcd	add decimal with extend	abcd dy,dx abcd ay@-,aX@-	

addb
addw
addl
	add binary	addX *ea*,d*n*
		addX d*n*,*ea*
		addX *ea*,a*n* (except addb)
		addX #*data*,*ea*	

addqb addqw addql	add quick	addqX #*data*,*ea*	
addxb addxw addxl	add extended	addxX dy,dX addxX ay@-,aX@-	
andb andw andl	logical and	andX *ea*,d*n* andX d*n*,*ea* andX #*data*,d*n*	
aslb aslw asll	arithmetic shift left	aslX dX,dy aslX #*data*,dy aslX *ea*	
asrb asrw asrl	arithmetic shift right	asrX dX,dy asrX #*data*,dy asrX *ea*	
bcc bccl bccs	branch conditionally	bccX *label*	68020
bchg	test a bit and change	bchg d*n*,*ea* bchg #*data*,*ea*	
bclr	test a bit and clear	bclr d*n*,*ea* bclr #*data*,*ea*	
bkpt	breakpoint	bkpt #*data*	68020
bset	test a bit and set	bset d*n*,*ea*	

Table 6-7 *List of MC680x0 Instruction Codes— Continued*

Mnemonic	*Operation Name*	*Syntax*	*Processor*
		bset #*data*, *ea*	
btst	test a bit	btst d*n*, *ea* btst #*data*, *ea*	
bfchg bfclr	test a bit field and change test a bit field and clear	bfchg *ea{offset:width}* bfclr *ea{offset:width}*	68020 68020
bfexts	extract a bit field signed	bfexts *ea{offset:width}*, d*n*	68020
bfextu	extract a bit field unsigned	bfextu *ea{offset:width}*, d*n*	68020
bfffo	find first one in bit field	bfffo *ea{offset:width}*, d*n*	68020
bfins	insert a bit field	bfins d*n*, *ea{offset:width}*	68020
bfset	test a bit field and set	bfset *ea{offset:width}*	68020
bftst	test a bit field	bftst *ea{offset:width}*	68020
bcs bcsl bcss	branch carry set	bcs*X* *ea*	68020
beq beql beqs	branch on equal	beq*X* *ea*	68020
bge bgel bges	branch greater or equal	bge*X* *ea*	68020
bgt bgtl bgts	branch greater than	bgt*X* *ea*	68020
bhi bhil bhis	branch higher	bhi*X* *ea*	68020
ble blel bles	branch less than or equal	ble*X* *ea*	68020
bls blsl	branch lower or same	bls*X* *ea*	68020
blt bltl blts	branch less than	blt*X* *ea*	
bmi bmil bmis	branch minus	bmi*X* *ea*	

Table 6-7 *List of MC680x0 Instruction Codes— Continued*

Mnemonic	*Operation Name*	*Syntax*	*Processor*
bne bnel bnes	branch not equal	bne*X* *ea*	 68020
bpl bpll bpls	branch positive	bpl*X* *ea*	 68020
bra bral bras	branch always	bra*X* *label*	 68020
bsr bsrl bsrs	subroutine branch	bsr*X* *label*	 68020
bvc bvcl bvcs	branch overflow clear	bvc*X* *ea*	 68020
bvs bvsl bvss	branch overflow set	bvs*X* *ea* bvsl	 68020
callm	call module	callm #*data*, *ea*	68020
cas2b cas2l cas2w	compare & swap with operand	cas2*X* dc1:dc2,du1:du2,(rn1):(rn2)	68020 68020 68020
casb casl casw	compare & swap with operand	cas*X* dc,du, *ea*	68020 68020 68020
chkb chkw chkl	check register against bounds	chk*X* *ea*,dn	68020 68020 68020
chk2b chk2l chk2w	check register against bounds	chk2*X* *ea*,rn	68020 68020 68020
clrb clrw clrl	clear an operand	clr*X* *ea*	
cmp2b cmp2l cmp2w	compare register against bounds	cmp2*x* *ea* rn	68020 68020 68020
cmpmb cmpmw	compare memory	cmpm*X* ay@+,a*X*@+	

Table 6-7 *List of MC680x0 Instruction Codes— Continued*

Mnemonic	*Operation Name*	*Syntax*	*Processor*
cmpml			
cmpb cmpw cmpl	arithmetic compare	cmp*X* *ea*, d*n* cmp*X* #*data*, *ea*	
dbcc	decrement & branch on carry clear	dbcc d*n*, *label*	
dbcs	" on carry set	dbcs d*n*, *label*	
dbeq	" on equal	dbeq d*n*, *label*	
dbf	" on false	dbf d*n*, *label*	
dbge	" on greater than or equal	dbge d*n*, *label*	
dbgt	" on greater than	dbgt d*n*, *label*	
dbhi	" on high	dbhi d*n*, *label*	
dble	" on less than or equal	dble d*n*, *label*	
dbls	" on low or same	dbls d*n*, *label*	
dblt	" on less than	dblt d*n*, *label*	
dbmi	" on minus	dbmi d*n*, *label*	
dbne	" on not equal	dbne d*n*, *label*	
dbpl	" on plus	dbpl d*n*, *label*	
dbra	" always (same as dbf)	dbra d*n*, *label*	
dbt	" on True	dbt d*n*, *label*	
dbvc	" on overflow clear	dbvc d*n*, *label*	
dbvs	" on overflow set	dbvs d*n*, *label*	
divs	signed divide	divs *ea*, d*n*	
divsl		divs*X* *ea*, d*n*	68020
divsll		divs*X* *ea*, d*q*	68020
		divs*X* *ea*, d*r*:d*q*	68020
divu	unsigned divide	divu *ea*, d*n*	
divul		divu*X* *ea*, d*n*	68020
divuw		divu*X* *ea*, d*n*	68020
		divu*X* *ea*, d*q*	68020
		divu*X* *ea*, d*r*:d*q*	68020
divull		divull *ea*, d*r*:d*q*	68020
eorb	logical exclusive or	eor*X* d*n*, *ea*	
eorw		eor*X* #*data*, *ea*	
eorl		eorb #*data*, *cc*	
		eorw #*data*, *sr*	
exg	exchange registers	exg r*x*, r*y*	
extbl	sign extend	extbl d*n*	68020
extw		ext*X* d*n*	
extl			
jmp	jump	jmp *ea*	
jsr	jump to subroutine	jsr *ea*	
jcc	jump carry clear	jcc *ea*	

Table 6-7 *List of MC680x0 Instruction Codes— Continued*

Mnemonic	*Operation Name*	*Syntax*	*Processor*
jcs	jump on carry	jcs *ea*	
jeq	jump on equal	jeq *ea*	
jge	jump greater or equal	jge *ea*	
jgt	jump greater than	jgt *ea*	
jhi	jump higher	jhi *ea*	
jle	jump less than or equal	jle *ea*	
jls	jump lower or same	jls *ea*	
jlt	jump less than	jlt *ea*	
jmi	jump minus	jmi *ea*	
jne	jump not equal	jne *ea*	
jpl	jump positive	jpl *ea*	
jra	jump always	jra *ea*	
jbsr	jump to subroutine	jbsr *ea*	
jvc	jump no overflow	jvc *ea*	
jvs	jump on overflow	jvs *ea*	
lea	load effective address	lea *ea*, a*n*	
link	link and allocate	link a*n*, #*disp*	
linkl		linkl a*n*, #*disp*	68020
lslb	logical shift left	lsl*X* d*x*, dy	
lslw		lsl*X* #*data*, dy	
lsll		lsl*X* *ea*	
lsrb	logical shift right	lsr*X* d*x*, dy	
lsrw		lsr*X* #*data*, dy	
lsrl		lsr*X* *ea*	
movb	move data	mov*X* *ea*, *ea*	
movl			
movw		mov*X* #*data*, d*n*	
movw	move from condition code register	movw *cc*, *ea*	
movw	move from status register	movw sr, *ea*	
movc	move to/from control register	movc r*n*, *cr*	
		movc *cr*, r*n*	
moveml	move multiple registers	movem*X* #*mask*, *ea*	
movemw		movem*X* *ea*, #*mask*	
		movem*X* *ea*, *reglist*	
		movem*X* *reglist*, *ea*	
movepl	move peripheral	movep*X* d*n*, a*n*@ (*d*)	
movepw		movep*X* a*n*@d, *n* (*d*)	
moveq	move quick	moveq #*data*, d*n*	
movsb	move to/from address space	movs*X* r*n*, *ea*	
movsw		movs*X* *ea*, r*n*	
movsl			

Table 6-7 *List of MC680x0 Instruction Codes— Continued*

Mnemonic	*Operation Name*	*Syntax*	*Processor*
muls mulslw mulsll	signed multiply	muls *ea*, d*n* muls*X* *ea*, d*l* muls*X* *ea*, d*h*:d*l*	 68020 68020
mulu mulul	unsigned multiply	mulu *ea*, d*n* mulu*X* *ea*, d*l* mulu*X* *ea*, d*h*:d*l*	 68020 68020
nbcd	negate decimal with extend	nbcd *ea*	
negb negw negl	negate binary	neg*X* *ea*	
negxb negxw negxl	negate binary with extend	negx*X* *ea*	
nop	no operation	nop	
notb notw notl	logical complement	not*X* *ea*	
orb orw orl	inclusive or	or*X* *ea*, d*n* or*X* d*n*, *ea* or #*data*, *ea* orb #*data*, *cc* orw #*data*, *sr*	
pack	pack	pack a*X*@-, ay@-, #*data* pack d*X*, dy, #*data*	68020 68020
pea	push effective address	pea *ea*	
reset	reset device	reset	
rolb rolw roll	rotate left rotate left	rol*X* dx, dy rol*X* #*data*, dy rol*X* *ea*	
rorb rorw rorl	rotate right	ror*X* dx, dy ror*X* #*data*, dy ror*X* *ea*	
roxlb roxlw roxll	rotate left with extend	roxl*X* dx, dy roxl*X* #*data*, dy roxl*X* *ea*	
roxrb roxrw roxrl	rotate right with extend	roxr*X* dx, dy roxr*X* #*data*, dy roxr*X* *ea*	
rtd	return and deallocate parameters	rtd #*data*	

Table 6-7 *List of MC680x0 Instruction Codes— Continued*

Mnemonic	*Operation Name*	*Syntax*	*Processor*
rte rtm rtr rts	return from exception return from module return and restore codes return from subroutine	rte rtm *rn* rtr rts rts #*n*	 68020
sbcd	subtract decimal with extend	sbcd dy,dx sbcd ay@-,a*X*@-	
stop	halt machine	stop #*xxx*	
subb subw subl	arithmetic subtract	sub*X* *ea*,d*n* sub*X* d*n*,*ea* sub*X* *ea*,a*n* sub*X* #*data*,*ea*	
st sf shi sls scc scs sne seq svc svs spl smi sge slt sgt sle	set all ones set all zeros set high set lower or same set carry clear set carry set set not equal set equal set no overflow set on overflow set plus set minus set greater or equal set less than set greater than set less than or equal	st *ea* sf *ea* shi *ea* sls *ea* scc *ea* scs *ea* sne *ea* seq *ea* svc *ea* svs *ea* spl *ea* smi *ea* sge *ea* slt *ea* sgt *ea* sle *ea*	
subqb subqw subql	subtract quick subtract quick	subq*X* #*data*,*ea*	
subxb subxw subxl	subtract extended	subx*X* dy,dx subx*X* ay@-,a*X*@-	
swap	swap register halves	swap d*n*	
tas	test operand then set	tas *ea*	
trap	trap	trap #*vector*	
trapcc trapccl trapccw	trap on carry clear	trapcc*X* trapcc*X* #*data*	68020 68020 68020
trapcs	trap on carry set	trapcsx	68020

Table 6-7 *List of MC680x0 Instruction Codes— Continued*

Mnemonic	*Operation Name*	*Syntax*	*Processor*
trapcsl trapcsw		trapcs*X* #*data*	68020 68020
trapeq trapeql trapeqw	trap on equal	trapeq*X* trapeq*X* #*data*	68020 68020 68020
trapf trapfl trapfw	trap on never true	trapf*X* trapf*X* #*data*	68020 68020 68020
trapge trapgel trapgew	trap on greater or equal	trapge*X* trapge*X* #*data*	68020 68020 68020
trapgt trapgtl trapgtw	trap on greater	trapgt*X* trapgt*X* #*data*	68020 68020 68020
traphi traphil traphiw	trap on hi	traphi*X* traphi*X*x #*data*	68020 68020 68020
traple traplel traplew	trap on less or equal	traple*X* traple*X*x #*data*	68020 68020 68020
trapls traplsl traplsw	trap on low or same	trapls*X* traplsx #*data*	68020 68020 68020
traplt trapltl trapltw	trap on less than	traplt*X* trapltx#*data*	68020 68020 68020
trapmi trapmil trapmiw	trap on minus trap on minus	trapmi*X* trapmi*X* #*data*	68020 68020 68020
trapne trapnel trapnew	trap on not equal	trapne*X* trapne*X* #*data*	68020 68020 68020
trappl trappll trapplw	trap on plus	trappl trapplx #*data*	68020 68020 68020
trapt traptl traptw	trap on always true	trapt traptx #*data*	68020 68020 68020
trapv	trap on overflow	trapv	

Table 6-7 *List of MC680x0 Instruction Codes— Continued*

Mnemonic	*Operation Name*	*Syntax*	*Processor*
trapvc trapvcl trapvcw	trap on overflow clear	trapvc trapvcx #*data*	68020 68020 68020
trapvs trapvsl trapvsw	trap on overflow set	trapvs trapvsx #*data*	68020 68020 68020
tstb tstw tstl	test operand	tst*X* *ea*	
unlk	unlink	unlk a*n*	
unpk	unpack bcd	unpk a*X*@-,ay@-,#*data* unpk d*X*,dy,#*data*	68020 68020

FPA Assembler Syntax — (New appendix)

Page 79

Append the following appendix at the end of the manual:

This appendix describes the Floating-Point Accelerator (FPA) support extensions to `as` included in Sun software release 3.1 and later.

The extensions to `as` are described in general, with discussions of two-, three-, and four-operand instruction examples. Some instructions covered separately don't follow the formats described at the beginning of the appendix. The appendix includes restrictions and potential errors, followed by a summary of supported floating-point instructions.

6.14. Instruction Syntax

The general format for floating-point instructions is

```
fpopt@A      operands
```

where

`fp` indicates an FPA instruction.

op is the opcode name.

t is the operand type, either single (s) or double (d).

The *@A* part of the instruction is optional. When present, *A* specifies the address register which contains the base address for the FPA and can be in the range 0..7. If this form is used, a previous instruction must load the FPA address (0xe0000000) into the specified address register.

If *@A* is not present, then absolute long addressing is used to refer to the FPA. This form is more efficient for short routines.

Depending on the instruction, there may be from zero to four operands specified. The operands can be any of the following forms:

Any MC68020 effective address, with the exception that absolute short addresses are not allowed for double-precision values.

If either of the data register or the address register is used to hold a double-precision value, then the value will be in a register pair and both registers, separated by a colon, must be specified in the instruction. For example:

```
fpaddd   d0:d1, fpa0
```

The only exception to this rule is the `fpltod` instruction (convert integer to double-precision value).

In some instructions (command register type) it is possible to specify that the register is in constant RAM. The syntax used for this case is `%n`, where `n` is a register number in the range 0 to 511.

6.15. Register Syntax

The 32 floating-point data registers are designated `fpa0, fpa1, ..., fpa31`. The supported control registers are:

Hardware	*Software*
MODE3_0	fpamode
WSTATUS	fpastatus

6.16. Operand Types

`as` supports three floating-point operand types:

- `s` for single-precision operands.
- `d` for double-precision operands.
- `l` for 32-bit integer operands, used for integer to floating-point conversions.

6.17. Two-Operand Instructions

Opcodes such as add, subtract, multiply, divide, negate, absolute value, square root, conversion from integer to floating-point, conversion from single to double (and vice versa) are all represented as:

```
fpopt  X, fpan
```

where *t*= `s` or `d`, and *X* is any valid MC68020 effective address for an operand or is an FPA data register.

If *X* is an FPA register which is in the constant RAM, then it can be in the range 0 to 511. If it is not in constant RAM, then it is one of the 32 FPA data registers. When *X* is an FPA register, then `fpa`*n* is one of the 32 floating-point data registers. If *X* is an effective address, then `fpa`*n* is one of the FPA registers in the range 0 to 15. The following are examples of such instructions:

```
                Instruction                          Computes

fpnegs      <effective address>, fpa1
fpsqrd      <effective address>, fpa2
fpsubs      fpa1, fpa2                          fpa2 ← fpa2 - fpa1
fprsubs     fpa1, fpa2                          fpa2 ← fpa1 - fpa2
fpdivs      d0, fpa2                            fpa2 ← fpa2 / d0
fprdivs     d0, fpa2                            fpa2 ← d0 / fpa2
```

In the above examples `fprsubs` and `fprdivs` are the reverse subtract and reverse divide operators, respectively.

The opcodes for `sine`, `cosine`, `atan`, `e^x`, `e^x -1`, `ln(x)`, `ln(1+x)`, `sqrt(x)`, and `sincos(x)` are all supported as command register type instructions:

```
fpopt  fpax, fpan
```

where *t* = `s` or `d`.

`fpax` is either a floating-point register or a register in the constant RAM (which is specified as `%`number). For the `sincos` instruction, the destination operand is actually a register pair:

```
fpsincost  fpax, fpac:fpas
```

where `fpac` is the cosine's destination and `fpas` is the sine's destination.

6.18. Three-Operand Instructions

The opcodes +, -, *, / are supported in extended and command register forms as

```
fpop3t    X, fpam, fpan
```

where *t* = `s` or `d` and *X* is an <effective address>for an extended instruction or a floating-point register for a command register type of instruction.

In the *command register form*, *X* and `fpam` can indicate a register number in the constant RAM. That is, they can either be in the range 0 to 511 or in the range 0 to 31. In the *extended instruction form*, `fpam` and `fpan` must be in the range 0 to 15. In the above format the position of *X* and `fpam` can be exchanged for the commutative operators add and multiply (the result of the operation remains the same).

For example,

```
fpa2 ← <effective address> + fpa1
```

can be represented by either of the following forms:

```
fpadd3s  <effective address>, fpa1, fpa2
fpadd3s  fpa1, <effective address>, fpa2
```

The same rule applies to subtract and divide operations. However, they are not commutative, so different answers result from each order. For example,

```
fpa2 ← fpa1 - <effective address>
```

must be coded as:

```
fpsub3s  <effective address>, fpa1, fpa2
```

whereas

```
fpa2 ← <effective address> - fpa1
```

must be coded as:

```
fpsub3s  fpa1, <effective address>, fpa2
```

Following the same format,

```
fpa3 ← fpa2 - fpa1
```

must be coded as:

```
fpsub3s  fpa1, fpa2, fpa3
```

6.19. Four-Operand Instructions

In the extended and command register formats there are pivot instructions of the form:

```
fpopt  X, fpax, fpay, fpan
```

where `fpa`*n* is the destination floating-point data register and *t* = `s` or `d`, and *X* is an effective address or a floating-point register.

In the extended form, the positions of *X* and `fpa`*y* can be exchanged for both single- and double-precision types of instructions. In single-precision extended form, it is possible for two of the four operands to be effective addresses. This is in general either the first and third or the second and third operands.

In the command register form, `fpa`*x* and `fpa`*y* can be replaced by `%`*x* and `%`*y* indicating register numbers *x* and *y* in the constant RAM.

For four-operand instructions, `fpa`*x*, `fpa`*y* and `fpa`*n* can each be in the range 0 to 15, when *X* is an effective address. If *X* is an FPA register, then *X* and

fpa*n* must be in the range 0 to 31 and fpa*x* and fpa*y* can either be in the range 0 to 511 (designating a location in constant RAM) or else in the range 0 to 31.

These pivot instructions are rather complicated and will be dealt with completely. The following shows the forms of each operation, the assembly code equivalent to each form, a generalization of the assembly instruction and the sequence of operations equivalent to the pivot instruction.

Instruction		Meaning
fpma{s,d}	<effective address>, reg2, reg3, reg1	reg1 ← reg3 + (reg2 * operand)
fpma{s,d}	reg2, reg3, <effective address>, reg1	reg1 ← operand + (reg3 * reg2)
fpma{s,d}	reg4, reg2, reg3, reg1	reg1 ← reg3 + (reg2 * reg4)
fpmas	<ea1>, reg2, <ea2>, reg1	reg1 ← operand2 + (reg2 * operand1)

The fpma instruction, where m stands for multiply, and a stands for add, can be generalized as

```
fpmat X, fpax, fpay, fpan
```

where *t* is s or d, and *X* is an <effective address>or one of the floating-point data registers. In the extended type of instruction, the positions of *X* and fpa*y* can be exchanged. Also, for single precision either the first and third operands or the second and third operands can be effective addresses. Note that, for example,

```
fpmas   d0, fpa1, fpa2, fpa3
```

is equivalent to the following sequence of instructions

```
fpmul3s     d0, fpa1, temp
fpadd3s     temp, fpa2, temp
fpmoves     temp, fpa3
```

where temp is a temporary register.

Instruction		Meaning
fpms{s,d}	<effective address>, reg2, reg3, reg1	reg1 ← reg3 - (reg2 * operand)
fpms{s,d}	reg2, reg3, <effective address>, reg1	reg1 ← operand - (reg3 * reg2)
fpms{s,d}	reg4, reg2, reg3, reg1	reg1 ← reg3 - (reg2 * reg4)
fpmss	<ea1>, reg2, <ea2>, reg1	reg1 ← operand2 - (reg2 * operand1)

The fpms instruction, where m stands for multiply, and s stands for subtract, can be generalized as

```
fpmst   X, fpax, fpay, fpan
```

where *t* is `s` or `d`, and *X* is an <effective address> or one of the floating-point data registers. In the extended type of instruction, the positions of *X* and `fpay` can be exchanged. Also, in single-precision two-memory instructions, either the first and third operands or the second and third operands can be effective addresses. Note that, for example,

```
fpmss  fpa1, fpa2, d0, fpa3
```

is equivalent to the following sequence of instructions

```
fpmul3s     fpa1, fpa2, temp
fpsub3s     temp, d0, temp
fpmoves     temp, fpa3
```

The `fpmr` instruction, where `m` stands for multiply, and `r` stands for reverse subtract, can be generalized as

```
fpmrt    X, fpax, fpay, fpan
```

where *t* is `s` or `d`, and *X* is an <effective address> or one of the floating-point data registers. In the extended type of instruction, the positions of *X* and `fpay` can be exchanged.

Instruction		Meaning
`fpmr{s,d}`	`<effective address>, reg2, reg3, reg1`	reg1 ← (-reg3) + (reg2 * operand)
`fpmr{s,d}`	`reg2, reg3, <effective address>, reg1`	reg1 ← (-operand) + (reg3 * reg2)
`fpmr{s,d}`	`reg4, reg2, reg3, reg1`	reg1 ← (-reg3) + (reg2 * reg4)
`fpmrs`	`<ea1>, reg2, <ea2>, reg1`	reg1 ← (-operand2) + (reg2 * operand1)

In single-precision extended form either the first and third operands or the second and third operands can be effective addresses. Note that, for example,

```
fpmrs   d0, fpa1, fpa2, fpa3
```

is equivalent to the following sequence of instructions:

```
fpmul3s     d0, fpa1, temp
fpsub3s     fpa2, temp, temp
fpmoves     temp, fpa3
```

The `fpam` instruction, where `a` stands for add, and `m` stands for multiply, can be generalized as

```
fpamt    X, fpax, fpay, fpan
```

where *t* is `s` or `d`, and *X* is an <effective address> or one of the floating-point

data registers. In the extended type of instruction, the positions of *X* and `fpay` can be exchanged.

Instruction	Meaning
`fpam{s,d}  <effective address>, reg2, reg3, reg1`	reg1 ← reg3 * (reg2 + operand)
`fpam{s,d}  reg2, reg3, <effective address>, reg1`	reg1 ← operand * (reg3 + reg2)
`fpam{s,d}  reg4, reg2, reg3, reg1`	reg1 ← reg3 * (reg2 + reg4)
`fpams  <ea1>, reg2, <ea2>, reg1`	reg1 ← operand2 * (reg2 + operand1)

In single-precision two-memory instructions, either the first and third operands or the second and third operands can be effective addresses. Note that, for example,

```
fpams    fpa1, fpa2, fpa3, fpa4
```

is equivalent to the following sequence of instructions:

```
fpadd3s     fpa1, fpa2, temp
fpmul3s     temp, fpa3, temp
fpmoves     temp, fpa4
```

The `fpsm` instruction, where `s` stands for subtract, and `m` stands for multiply, can be generalized as

```
fpsmt    X, fpax, fpay, fpan
```

where *t* is `s` or `d`, and *X* is an effective address or one of the floating-point data registers. In the extended type of instruction, the positions of *X* and `fpay` can be exchanged. The special cases for single-precision instructions are that either the first and third operands or the second and third operands can be effective addresses.

Instruction	Meaning
`fpsm{s,d}  <effective address>, reg2, reg3, reg1`	reg1 ← reg3 * (reg2 - operand)
`fpsm{s,d}  reg2, reg3, <effective address>, reg1`	reg1 ← operand * (reg3 - reg2)
`fpsm{s,d}  reg4, reg2, reg3, reg1`	reg1 ← reg3 * (reg2 - reg4)
`fpsm{s,d}  reg2, <effective address>, reg3, reg1`	reg1 ← reg3 * (-reg2 + operand)
`fpsm{s,d}  reg2, reg4, reg3, reg1`	reg1 ← reg3 * (-reg2 + reg4)
`fpsms  <ea1>, reg2, <ea2>, reg1`	reg1 ← operand2 * (reg2 - operand1)
`fpsms  reg2, <ea1>,  <ea2>, reg1`	reg1 ← operand2 * (-reg2 + operand1)

Note that, for example,

```
fpsms  d0, fpa1, fpa2, fpa3
```

is equivalent to the following sequence of instructions:

```
fpsub3s  d0, fpa1, temp
fpmul3s  temp, fpa2, temp
fpmoves  temp, fpa3
```

6.20. Other Instructions

Other special instructions are listed below. In each of them the last operand is also the destination, except for `tst`, `cmp` and `mcmp` where `fpastatus` is the implied destination. *X* is either an effective address or an FPA data register and *t* is either `s` or `d` for all instructions except `fpmove`*t*, where *t* can be `s`, `d`, or `l`.

Table 6-8 *Other Instructions*

Mnemonic	*Syntax*	*Operation Name*
fpnop		nop
fptst*t*	X	operand compare with zero
fpcmp*t* fpmcmp*t*	X, fpa*m* X, fpa*m*	register *m* compare with operand register *m* compare magnitude with operand
fpmove*t* fpmove2*t* fpmove3*t* fpmove4*t*	fpa*m*, fpa*n* fpa*m*, fpa*n* fpa*m*, fpa*n* fpa*m*, fpa*n*	move floating-point registers 2x2 matrix move 3x3 matrix move 4x4 matrix move
fpdot2*t* fpdot3*t* fpdot4*t*	fpa*x*, fpa*y*, fpa*n* fpa*x*, fpa*y*, fpa*n* fpa*x*, fpa*y*, fpa*n*	fpa*n* ← fpa*x**fpa*y* + (fpa*x*+*1*) * (fpa*y*+*1*) fpa*n* ← fpa*x**fpa*y* + (fpa*x*+*1*) * (fpa*y*+*1*) + (fpa*x*+*2*) * (fpa*y*+*2*) fpa*n* ← fpa*x**fpa*y* + (fpa*x*+*1*)*(fpa*y*+*1*) + (fpa*x*+*2*)*(fpa*y*+*2*) + (fpa*x*+*3*)*(fpa*y*+*3*)
fptran2*t* fptran3*t* fptran4*t*	fpa*m*, fpa*n* fpa*m*, fpa*n* fpa*m*, fpa*n*	transpose 2x2 matrix transpose 3x3 matrix transpose 4x4 matrix
fpmove fpmove fpmove fpmove fpmove*t* fpmove*t*	fpamode, <ea> <ea>, fpamode fpastatus, <ea> <ea>, fpastatus fpa*m*, <ea> <ea>, fpa*n*	read mode register write to mode register read status register write to status register read a floating-point data register write to a floating-point data register

6.21. Restrictions and Errors

In double-precision instructions, when absolute short addressing or a single data or address register is used, `as` reports an invalid operand error.

For the dot product and matrix move and transpose instructions, when the register specified does not fall within the specified range, `as` reports a register out of range error.

For most instructions where one operand is an effective address, the register range is 0 to 15. If all operands are FPA registers, then the register range is 0 to 31. For constant RAM registers, the range is 0 to 511. `as` reports an invalid operand error when any of these registers are not within the permitted range.

6.22. Instruction Set Summary

In the following table, `X` is any valid MC68020 effective address (the form `(xxx):w` is not allowed for double) or FPA register. In some three- or four-address instructions the position of the `X` and one of the FPA register can be exchanged. This is shown in the fourth column of the following table.

Table 6-9 *Floating-Point Instructions*

Instruction	*Syntax*	*Operation*	*Alternative*
fpnegs	X, fpa*n*	negate single	
fpnegd	X, fpa*n*	negate double	
fpabss	X, fpa*n*	absolute value single	
fpabsd	X, fpa*n*	absolute value double	
fpltos	X, fpa*n*	convert integer to single	
fpltod	X, fpa*n*	convert integer to double	
fpstol	X, fpa*n*	convert single to integer	
fpdtol	X, fpa*n*	convert double to integer	
fpstod	X, fpa*n*	convert single to double	
fpdtos	X, fpa*n*	convert double to single	
fpsqrs	X, fpa*n*	square single	
fpsqrd	X, fpa*n*	square double	
fpadds	X, fpa*n*	add single	
fpadd3s	X, fpa*m*, fpa*n*	add single	fpa*m*, X, fpa*n*
fpaddd	X, fpa*n*	add double	
fpadd3d	X, fpa*m*, fpa*n*	add double	fpa*m*, X, fpa*n*
fpsubs	X, fpa*n*	subtract single	
fpsub3s	X, fpa*m*, fpa*n*	subtract single	fpa*m*, X, fpa*n*
fprsubs	<ea>, fpa*n*	reverse subtract single	
fpsubd	X, fpa*n*	subtract double	
fpsub3d	X, fpa*m*, fpa*n*	subtract double	fpa*m*, X, fpa*n*
fprsubd	<ea>, fpa*n*	reverse subtract double	

Table 6-9 *Floating-Point Instructions— Continued*

Instruction	*Syntax*	*Operation*	*Alternative*
fpmuls	X, fpa*n*	multiply single	
fpmul3s	X, fpa*m*, fpa*n*	multiply single	fpa*m*, X, fpa*n*
fpmuld	X, fpa*n*	multiply double	
fpmul3d	X, fpa*m*, fpa*n*	multiply double	fpa*m*, X, fpa*n*
fpdivs	X, fpa*n*	divide single	
fpdiv3s	X, fpa*m*, fpa*n*	divide single	fpa*m*, X, fpa*n*
fprdivs	<ea>, fpa*n*	reverse divide single	
fpdivd	X, fpa*n*	divide double	
fpdiv3d	X, fpa*m*, fpa*n*	divide double	fpa*m*, X, fpa*n*
fprdivd	<ea>, fpa*n*	reverse divide double	
fpnop		nop	
fptsts	X	single compare with 0	
fptstd	X	double compare with 0	
fpcmps	X, fpa*m*	single compare	
fpcmpd	X, fpa*m*	double compare	
fpmcmps	X, fpa*m*	single magnitude compare	
fpmcmpd	X, fpa*m*	double magnitude compare	
fpsins	fpa*x*, fpa*n*	sine single	
fpsind	fpa*x*, fpa*n*	sine double	
fpcoss	fpa*x*, fpa*n*	cosine single	
fpcosd	fpa*x*, fpa*n*	cosine double	
fpatans	fpa*x*, fpa*n*	atan single	
fpatand	fpa*x*, fpa*n*	atan double	
fpetoxs	fpa*x*, fpa*n*	e^x single	
fpetoxd	fpa*x*, fpa*n*	e^x double	
fpetoxm1s	fpa*x*, fpa*n*	e^x-1 single	
fpetoxm1d	fpa*x*, fpa*n*	e^x-1 double	
fplogns	fpa*x*, fpa*n*	ln(x) single	
fplognd	fpa*x*, fpa*n*	ln(x) double	
fplognp1s	fpa*x*, fpa*n*	ln(1+x) single	
fplognp1d	fpa*x*, fpa*n*	ln(1+x) double	
fpsincoss	fpa*x*, fpac:fpa*s*	fpac ← cosine(x), fpa*s* ← sine(x)	
fpsincosd	fpa*x*, fpac:fpa*s*	fpac ← cosine(x), fpa*s* ← sine(x)	
fpmas	X, fpa*x*, fpay, fpa*n*	fpa*n* ← (fpa*x* * X) + fpay	fpa*x*, X, fpay, fpa*n* fpay, fpa*x*, X, fpa*n* X, fpa*x*, X, fpa*n* fpa*x*, X, X, fpa*n*
fpmad	X, fpa*x*, fpay, fpa*n*	fpa*n* ← (fpa*x* * X) + fpay	fpa*x*, X, fpay, fpa*n*

Table 6-9 *Floating-Point Instructions— Continued*

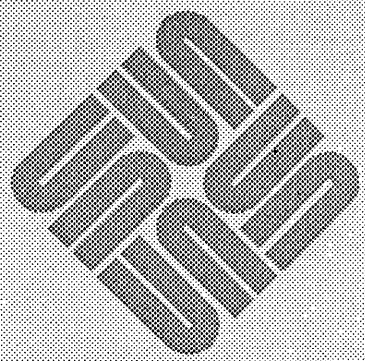
Instruction	*Syntax*	*Operation*	*Alternative*
fpmss	X, fpa*x*, fpay, fpa*n*	fpa*n* ← fpay - (fpa*x* * X)	fpay, fpa*x*, X, fpa*n* fpa*x*, X, fpay, fpa*n* fpay, fpa*x*, X, fpa*n* X, fpa*x*, X, fpa*n* fpa*x*, X, X, fpa*n*
fpmsd	X, fpa*x*, fpay, fpa*n*	fpa*n* ← fpay - (fpa*x* * X)	fpa*x*, X, fpay, fpa*n* fpay, fpa*x*, X, fpa*n*
fpmrs	X, fpa*x*, fpay, fpa*n*	fpa*n* ← (fpa*x* * X) - fpay	fpa*x*, X, fpay, fpa*n* fpay, fpa*x*, X, fpa*n* X, fpa*x*, X, fpa*n* fpa*x*, X, X, fpa*n*
fpmrd	X, fpa*x*, fpay, fpa*n*	fpa*n* ← (fpa*x* * X) - fpay	fpa*x*, X, fpay, fpa*n* fpay, fpa*x*, X, fpa*n*
fpams	X, fpa*x*, fpay, fpa*n*	fpa*n* ← (fpa*x* + X) * fpay	fpa*x*, X, fpay, fpa*n* fpay, fpa*x*, X, fpa*n* X, fpa*x*, X, fpa*n* fpa*x*, X, X, fpa*n*
fpamd	X, fpa*x*, fpay, fpa*n*	fpa*n* ← (fpa*x* + X) * fpay	fpa*x*, X, fpay, fpa*n* fpay, fpa*x*, X, fpa*n*
fpsms	X, fpa*x*, fpay, fpa*n*	fpa*n* ← (fpa*x* - X) * fpay	fpa*x*, X, fpay, fpa*n* fpay, fpa*x*, X, fpa*n* X, fpa*x*, X, fpa*n* fpa*x*, X, X, fpa*n*
fpsmd	X, fpa*x*, fpay, fpa*n*	fpa*n* ← (fpa*x* - X) * fpay	fpa*x*, X, fpay, fpa*n* fpay, fpa*x*, X, fpa*n*
fpmoves	<ea>, fpa*n*	write to a register, single	
fpmoved	<ea>, fpa*n*	write to a register, double	
fpmovel	<ea>, fpa*n*	write to a register, integer	
fpmoves	fpa*m*, <ea>	read a register, single	
fpmoved	fpa*m*, <ea>	read a register, double	
fpmove2s	fpa*m*, fpa*n*	2x2 matrix move, single	
fpmove2d	fpa*m*, fpa*n*	2x2 matrix move, double	
fpmove3s	fpa*m*, fpa*n*	3x3 matrix move, single	
fpmove3d	fpa*m*, fpa*n*	3x3 matrix move, double	
fpmove4s	fpa*m*, fpa*n*	4x4 matrix move, single	
fpmove4d	fpa*m*, fpa*n*	4x4 matrix move, double	

Table 6-9 *Floating-Point Instructions— Continued*

Instruction	*Syntax*	*Operation*	*Alternative*
fpdot2s	fpa*x*, fpa*y*, fpa*n*	fpa*n* ← fpa*x**fpa*y* + (fpa*x+1*) * (fpa*y+1*)	
fpdot2d	fpa*x*, fpa*y*, fpa*n*	fpa*n* ← fpa*x**fpa*y* + (fpa*x+1*) * (fpa*y+1*)	
fpdot3s	fpa*x*, fpa*y*, fpa*n*	fpa*n* ← fpa*x**fpa*y* + (fpa*x+1*) * (fpa*y+1*) + (fpa*x+2*) * (fpa*y+2*)	
fpdot3d	fpa*x*, fpa*y*, fpa*n*	fpa*n* ← fpa*x**fpa*y* + (fpa*x+1*) * (fpa*y+1*) + (fpa*x+2*) * (fpa*y+2*)	
fpdot4s	fpa*x*, fpa*y*, fpa*n*	fpa*n* ← fpa*x**fpa*y* + (fpa*x+1*)*(fpa*y+1*) + (fpa*x+2*)*(fpa*y+2*) + (fpa*x+3*)*(fpa*y+3*)	
fpdot4d	fpa*x*, fpa*y*, fpa*n*	fpa*n* ← fpa*x**fpa*y* + (fpa*x+1*)*(fpa*y+1*) + (fpa*x+2*)*(fpa*y+2*) + (fpa*x+3*)*(fpa*y+3*)	
fptran2s	fpa*m*, fpa*n*	transpose 2x2 matrix, single	
fptran2d	fpa*m*, fpa*n*	transpose 2x2 matrix, double	
fptran3s	fpa*m*, fpa*n*	transpose 3x3 matrix, single	
fptran3d	fpa*m*, fpa*n*	transpose 3x3 matrix, double	
fptran4s	fpa*m*, fpa*n*	transpose 4x4 matrix, single	
fptran4d	fpa*m*, fpa*n*	transpose 4x4 matrix, double	
fpmove	fpamode, <ea>	read the mode register	
fpmove	<ea>, fpamode	write on mode register	
fpmove	fpastatus, <ea>	read the status register	
fpmove	<ea>, fpastatus	write to status register	

A

Insert Pages for 3.0 Commands Reference Manual



A

Insert Pages for 3.0 Commands Reference Manual

NAME

whereis – locate source, binary, and/or manual for program

SYNOPSIS

whereis [**–sbm**] [**–u**] [**–BMS** dir ... **–f**] *filename* ...

DESCRIPTION

whereis locates source/binary and manuals sections for specified files. The supplied names are first stripped of leading pathname components and any (single) trailing extension of the form *.ext*, for example, *.c*. Prefixes of *s.* resulting from use of source code control are also dealt with. *whereis* then attempts to locate the desired program in a list of standard places:

/bin
/usr/bin
/usr/5bin
/usr/games
/usr/hosts
/usr/include
/usr/local
/usr/etc
/usr/lib
/usr/man
/usr/src
/usr/ucb

OPTIONS

–b Search only for binaries.

–s Search only for sources.

–m Search only for manual sections.

–u Search for unusual entries. A file is said to be unusual if it does not have one entry of each requested type. Thus **whereis –m –u *** asks for those files in the current directory which have no documentation.

–B Change or otherwise limit the places where *whereis* searches for binaries.

–M Change or otherwise limit the places where *whereis* searches for manual sections.

–S Change or otherwise limit the places where *whereis* searches for sources.

–f Terminates the last directory list and signals the start of file names, and *must* be used when any of the **–B**, **–M**, or **–S** options are used.

EXAMPLE

Find all files in */usr/bin* which are not documented in */usr/man/man1* with source in */usr/src/cmd*:

```
angel% cd /usr/ucb
angel% whereis –u –M /usr/man/man1 –S /usr/src/cmd –f *
```

FILES

/usr/src/*
/usr/{doc,man}/*
/lib, /etc, /usr/{lib,bin,ucb,old,new,local}

BUGS

Since *whereis* uses *chdir*(2) to run faster, pathnames given with the **–M**, **–S**, or **–B** must be full; that is, they must begin with a '/'.

NAME

which – locate a program file, including any aliases or paths

SYNOPSIS

which [*command*] ...

DESCRIPTION

For each *command* argument given, *which* looks up the pathname of the file used to execute that *command.* If there is an alias set for *command* (*csh* only), *which* displays its value. Otherwise, *which* searches for the pathname along your search path.

DIAGNOSTICS

A diagnostic is given for names which are aliased to more than a single word, or if an executable file with the argument *command* was not found in the path.

NAME

listen – listen for connections on a socket

SYNOPSIS

listen(s, backlog)
int s, backlog;

DESCRIPTION

To accept connections, a socket is first created with *socket*(2), a backlog for incoming connections is specified with *listen*(2) and then the connections are accepted with *accept*(2). The *listen* call applies only to sockets of type SOCK_STREAM or SOCK_SEQPACKET.

The *backlog* parameter defines the maximum length the queue of pending connections may grow to. If a connection request arrives with the queue full the client will receive an error with an indication of ECONNREFUSED.

RETURN VALUE

A 0 return value indicates success; –1 indicates an error.

ERRORS

The call fails if:

EBADF	The argument *s* is not a valid descriptor.
ENOTSOCK	The argument *s* is not a socket.
EOPNOTSUPP	The socket is not of a type that supports the operation *listen*.

SEE ALSO

accept(2), connect(2), socket(2)

BUGS

The *backlog* is currently limited (silently) to 5.

NAME

lseek, tell – move read/write pointer

SYNOPSIS

```
#include <sys/file.h>
#define L_SET     0     /* set the seek pointer */
#define L_INCR    1     /* increment the seek pointer */
#define L_XTND    2     /* extend the file size */

pos = lseek(d, offset, whence)
long pos;
int d;
long offset;
int whence;
```

DESCRIPTION

The descriptor *d* refers to a file or device open for reading and/or writing. *lseek* sets the file pointer of *d* as follows:

If *whence* is L_SET, the pointer is set to *offset* bytes.

If *whence* is L_INCR, the pointer is set to its current location plus *offset*.

If *whence* is L_XTND, the pointer is set to the size of the file plus *offset*.

Upon successful completion, the resulting pointer location as measured in bytes from beginning of the file is returned. Some devices are incapable of seeking. The value of the pointer associated with such a device is undefined.

The obsolete function *tell(fildes)* is identical to *lseek(fildes, 0L, L_INCR)*.

NOTES

Seeking far beyond the end of a file, then writing, creates a gap or "hole", which occupies no physical space and reads as zeros.

RETURN VALUE

Upon successful completion, a non-negative (long) integer, the current file pointer value, is returned. Otherwise, a value of −1 is returned and *errno* is set to indicate the error.

ERRORS

lseek will fail and the file pointer will remain unchanged if:

EBADF	*Fildes* is not an open file descriptor.
ESPIPE	*Fildes* is associated with a pipe or a socket.
EINVAL	*whence* is not a proper value.

SEE ALSO

dup(2), open(2V)

NAME

mount – keep track of remotely mounted filesystems

SYNOPSIS

#include <rpcsvc/mount.h>

RPC INFO

program number:

MOUNTPROG

xdr routines:

```
xdr_exportbody(xdrs, ex)
        XDR *xdrs;
        struct exports *ex;
xdr_exports(xdrs, ex);
        XDR *xdrs;
        struct exports **ex;
xdr_fhandle(xdrs, fh);
        XDR *xdrs;
        fhandle_t *fp;
xdr_fhstatus(xdrs, fhs);
        XDR *xdrs;
        struct fhstatus *fhs;
xdr_groups(xdrs, gr);
        XDR *xdrs;
        struct groups *gr;
xdr_mountbody(xdrs, ml)
        XDR *xdrs;
        struct mountlist *ml;
xdr_mountlist(xdrs, ml);
        XDR *xdrs;
        struct mountlist **ml;
xdr_path(xdrs, path);
        XDR *xdrs;
        char **path;
```

procs:

MOUNTPROC_MNT
: argument of xdr_path, returns fhstatus.
  Requires unix authentication.

MOUNTPROC_DUMP
: no args, returns struct mountlist

MOUNTPROC_UMNT
: argument of xdr_path, no results.
  requires unix authentication.

MOUNTPROC_UMNTALL
: no arguments, no results.
  requires unix authentication.
  umounts all remote mounts of sender.

MOUNTPROC_EXPORT
MOUNTPROC_EXPORTALL
: no args, returns struct exports

versions:

MOUNTVERS_ORIG

structures:

```
struct mountlist {          /* what is mounted */
        char *ml_name;
        char *ml_path;
        struct mountlist *ml_nxt;
};
struct fhstatus {
        int fhs_status;
        fhandle_t fhs_fh;
};
/*
 * List of exported directories
 * An export entry with ex_groups
 * NULL indicates an entry which is exported to the world.
 */
struct exports {
        dev_t           ex_dev;         /* dev of directory */
        char            *ex_name;       /* name of directory */
        struct groups   *ex_groups;     /* groups allowed to mount this entry */
        struct exports  *ex_next;
};
struct groups {
        char            *g_name;
        struct groups   *g_next;
};
```

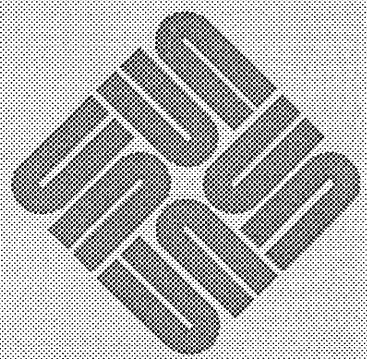
SEE ALSO

mount(8), showmount(8), mountd(8C),

NFS Protocol Spec, in *Networking on the Sun Workstation.*

B

Files to be Saved



B

Files to be Saved

1. Standalone Systems

```
/.login
/.cshrc
/.rhosts
/etc/passwd
/etc/exports
/etc/printcap
/etc/fstab
/etc/hosts.equiv
/etc/dumpdates
/etc/dump
/etc/hosts
/etc/rc.local
/etc/rc.boot
/etc/ttytype
/etc/ttys
/usr/lib/crontab
/usr/lib/sendmail.cf
```

2. Servers

```
/.login
/.cshrc
/.rhosts
/etc/passwd
/etc/exports
/etc/printcap
/etc/fstab
/etc/hosts.equiv
/etc/dumpdates
/etc/dump
/etc/hosts
/etc/nd.local
/etc/rc.boot
/etc/rc.local
/etc/ttytype
/etc/ttys
/usr/hosts/MAKEHOSTS
/private/usr/lib/crontab
```

```
/private/usr/lib/sendmail.cf
```

3. Diskless Clients

```
/.login
/.cshrc
/.rhosts
/etc/passwd
/etc/printcap
/etc/fstab
/etc/hosts.equiv
/etc/hosts
/etc/rc.local
/etc/rc.boot
/private/usr/lib/crontab
/private/usr/lib/sendmail.cf
```

In addition to the files listed above, the following files will be saved if your system is a YP master or a non YP machine.

4. YP Masters And Non YP Machines

```
/etc/ethers
/etc/netgroup
/etc/services
/etc/protocols
/etc/servers
/etc/group
/etc/networks
/etc/rpc
/usr/lib/aliases
/usr/etc/yp/domain/* (for yp masters only)
```

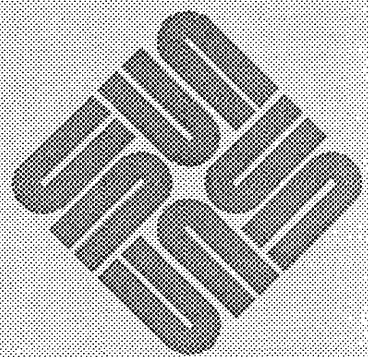
If your system is a YP slave server, all the YP maps under `/usr/etc/yp/domain` will be saved.

5. YP Slave Servers

```
/usr/etc/yp/domain/*
```

C

Optional Software for Release 3.2



C

Optional Software for Release 3.2

The following is the optional software available in the 3.2 release.

NETWORKING:

```
/usr/bin/traffic
/usr/bin/ypcat
/usr/bin/ypmatch
/usr/bin/yppasswd
/usr/bin/ypwhich
/usr/etc/etherfind
/usr/etc/in.ftpd
/usr/etc/in.rexecd
/usr/etc/in.rwhod
/usr/etc/in.telnetd
/usr/etc/in.tftpd
/usr/etc/in.timed
/usr/etc/nfsstat
/usr/etc/ping
/usr/etc/rarpd
/usr/etc/route
/usr/etc/rpc.etherd
/usr/etc/rpc.mountd
/usr/etc/rpc.rquotad
/usr/etc/rpc.rusersd
/usr/etc/rpc.rwalld
/usr/etc/rpc.sprayd
/usr/etc/rpc.yppasswdd
/usr/etc/rpcinfo
/usr/etc/rwall
/usr/etc/showmount
/usr/etc/spray
/usr/etc/yp/
/usr/etc/yp/makedbm
/usr/etc/yp/yppush
/usr/etc/yp/ypset
/usr/etc/yp/ypxfr
/usr/etc/yp/yppoll
/usr/etc/yp/stdhosts
/usr/etc/yp/ypinit
/usr/etc/yp/ypxfr_1perday
/usr/etc/yp/ypxfr_2perday
```

```
/usr/etc/yp/ypxfr_1perhour
/usr/etc/yp/revnetgroup
/usr/etc/yp/Makefile
/usr/etc/ypserv
/usr/ucb/ftp
/usr/ucb/netstat
/usr/ucb/rcp
/usr/ucb/rdate
/usr/ucb/rdist
/usr/ucb/rlogin
/usr/ucb/rsh
/usr/ucb/rup
/usr/ucb/ruptime
/usr/ucb/rusers
/usr/ucb/rwho
/usr/ucb/telnet
/usr/ucb/tftp
```

DEBUGGING

```
/usr/lib/adb/adbgen1
/usr/lib/adb/adbgen3
/usr/lib/adb/adbgen4
/usr/lib/adb/adbgen
/usr/lib/adb/buf
/usr/lib/adb/callout
/usr/lib/adb/callout.nxt
/usr/lib/adb/cblock
/usr/lib/adb/cblock.nxt
/usr/lib/adb/clist
/usr/lib/adb/dino
/usr/lib/adb/dir
/usr/lib/adb/dir.nxt
/usr/lib/adb/file
/usr/lib/adb/filsys
/usr/lib/adb/ifnet
/usr/lib/adb/inode
/usr/lib/adb/inpcb
/usr/lib/adb/iovec
/usr/lib/adb/ipreass
/usr/lib/adb/ipreass.nxt
/usr/lib/adb/mact
/usr/lib/adb/mact.nxt
/usr/lib/adb/mbstat
/usr/lib/adb/mbuf
/usr/lib/adb/mbuf.nxt
/usr/lib/adb/mbufs
/usr/lib/adb/mbufs.nxt
/usr/lib/adb/mount
/usr/lib/adb/pcb
/usr/lib/adb/proc
/usr/lib/adb/protosw
/usr/lib/adb/rawcb
```

```
/usr/lib/adb/rtentry
/usr/lib/adb/rusage
/usr/lib/adb/setproc
/usr/lib/adb/setproc.done
/usr/lib/adb/setproc.nop
/usr/lib/adb/setproc.nxt
/usr/lib/adb/socket
/usr/lib/adb/stat
/usr/lib/adb/tcpcb
/usr/lib/adb/tcpip
/usr/lib/adb/tcpreass
/usr/lib/adb/tcpreass.nxt
/usr/lib/adb/text
/usr/lib/adb/traceall
/usr/lib/adb/traceall.nxt
/usr/lib/adb/tty
/usr/lib/adb/u
/usr/lib/adb/ucred
/usr/lib/adb/uio
/usr/lib/adb/vnode
/usr/lib/adb/vtimes
/usr/lib/adb/adbsub.o
/usr/bin/dbxtool
```

SUNTOOLS_USERS

```
/usr/bin/adjacentscreens
/usr/bin/align_equals
/usr/bin/capitalize
/usr/bin/clear_functions
/usr/bin/clock
/usr/bin/clocktool
/usr/bin/cmdtool
/usr/bin/coretool
/usr/bin/defaults_from_input
/usr/bin/defaults_to_indentpro
/usr/bin/defaults_to_mailrc
/usr/bin/defaultsedit
/usr/bin/fontedit
/usr/bin/get_selection
/usr/bin/gfxtool
/usr/bin/iconedit
/usr/bin/indentpro_to_defaults
/usr/bin/input_from_defaults
/usr/bin/insert_brackets
/usr/bin/lockscreen
/usr/bin/lockscreen_default
/usr/bin/mailrc_to_defaults
/usr/bin/mailtool
/usr/bin/overview
/usr/bin/perfmeter
/usr/bin/perfmon
/usr/bin/scrolldefaults
```

```
/usr/bin/selection_svc
/usr/bin/setkeys
/usr/bin/shelltool
/usr/bin/shift_lines
/usr/bin/stty_from_defaults
/usr/bin/suntools
/usr/bin/swin
/usr/bin/switcher
/usr/bin/tektool
/usr/bin/textedit
/usr/bin/othertools
/usr/bin/toolplaces
/usr/bin/traffic
/usr/demo/bouncedemo
/usr/demo/canvas_demo
/usr/demo/cursor_demo
/usr/demo/framedemo
/usr/demo/jumpdemo
/usr/demo/spheresdemo
/usr/lib/defaults/Defaults.d
/usr/lib/defaults/Indent.d
/usr/lib/defaults/Input.d
/usr/lib/defaults/Mail.d
/usr/lib/defaults/Menu.d
/usr/lib/defaults/Scrollbar.d
/usr/lib/defaults/SunView.d
/usr/lib/defaults/Text.d
/usr/lib/defaults/Tty.d
/usr/lib/fonts/fixedwidthfonts/README
/usr/lib/fonts/fixedwidthfonts/apl.r.10
/usr/lib/fonts/fixedwidthfonts/cmr.b.14
/usr/lib/fonts/fixedwidthfonts/cmr.b.8
/usr/lib/fonts/fixedwidthfonts/cmr.r.14
/usr/lib/fonts/fixedwidthfonts/cmr.r.8
/usr/lib/fonts/fixedwidthfonts/cour.b.10
/usr/lib/fonts/fixedwidthfonts/cour.b.12
/usr/lib/fonts/fixedwidthfonts/cour.b.14
/usr/lib/fonts/fixedwidthfonts/cour.r.10
/usr/lib/fonts/fixedwidthfonts/cour.r.12
/usr/lib/fonts/fixedwidthfonts/cour.r.14
/usr/lib/fonts/fixedwidthfonts/gacha.b.8
/usr/lib/fonts/fixedwidthfonts/gacha.b.7
/usr/lib/fonts/fixedwidthfonts/gacha.r.7
/usr/lib/fonts/fixedwidthfonts/gacha.r.8
/usr/lib/fonts/fixedwidthfonts/gallant.r.10
/usr/lib/fonts/fixedwidthfonts/gallant.r.19
/usr/lib/fonts/fixedwidthfonts/sail.r.6
/usr/lib/fonts/fixedwidthfonts/screen.b.12
/usr/lib/fonts/fixedwidthfonts/screen.b.14
/usr/lib/fonts/fixedwidthfonts/screen.r.11
/usr/lib/fonts/fixedwidthfonts/screen.r.12
/usr/lib/fonts/fixedwidthfonts/screen.r.13
/usr/lib/fonts/fixedwidthfonts/screen.r.14
```

```
/usr/lib/fonts/fixedwidthfonts/screen.r.7
/usr/lib/fonts/fixedwidthfonts/serif.r.10
/usr/lib/fonts/fixedwidthfonts/serif.r.11
/usr/lib/fonts/fixedwidthfonts/serif.r.12
/usr/lib/fonts/fixedwidthfonts/serif.r.14
/usr/lib/fonts/fixedwidthfonts/serif.r.16
/usr/lib/fonts/tekfonts/tekfont0
/usr/lib/fonts/tekfonts/tekfont1
/usr/lib/fonts/tekfonts/tekfont2
/usr/lib/fonts/tekfonts/tekfont3
/usr/lib/view_surface
/usr/lib/.rootmenu
/usr/lib/.suntools
/usr/lib/.textswrc
```

SUNTOOLS_PROGRAMMERS

```
/usr/include/images/*
/usr/include/suntool/*
/usr/include/sunwindow/*
/usr/lib/libsuntool.a
/usr/lib/libsunwindow.a
/usr/lib/libtoolmerge.a
/usr/src/sun/suntool/*
```

SUNTOOLS_SOURCE

```
/usr/src/sun/suntool/mailtool/main.o
/usr/src/sun/suntool/mailtool/tool.o
/usr/src/sun/suntool/mailtool/selection.o
/usr/src/sun/suntool/mailtool/cmds.o
/usr/src/sun/suntool/mailtool/mail.o
/usr/src/sun/suntool/mailtool/subr.o
/usr/src/sun/suntool/mailtool/vars.o
/usr/src/sun/suntool/iconedit/iconedit_canvas.c
/usr/src/sun/suntool/iconedit/iconedit_main.c
/usr/src/sun/suntool/iconedit/iconedit_mpr.c
/usr/src/sun/suntool/iconedit/iconedit_panel.c
/usr/src/sun/suntool/iconedit/iconedit_browse.c
/usr/src/sun/suntool/iconedit/iconedit.h
/usr/src/sun/suntool/iconedit/Makefile
/usr/src/sun/suntool/Makefile
/usr/src/sun/suntool/toolmerge.c
/usr/src/sun/suntool/suntools.c
/usr/src/sun/suntool/suntools_menu.c
/usr/src/sun/suntool/selection_svc.c
/usr/src/sun/suntool/textedit.c
/usr/src/sun/suntool/view_surface.c
/usr/src/sun/suntool/gfxtool.c
/usr/src/sun/suntool/shelltool.c
/usr/src/sun/suntool/cmdtool.c
/usr/src/sun/suntool/clock.c
/usr/src/sun/suntool/toolplaces.c
/usr/src/sun/suntool/overview.c
```

```
/usr/src/sun/suntool/tektool.c
/usr/src/sun/suntool/canvas_demo.c
/usr/src/sun/suntool/cursor_demo.c
/usr/src/sun/suntool/jumpdemo.c
/usr/src/sun/suntool/spheresdemo.c
/usr/src/sun/suntool/bouncedemo.c
/usr/src/sun/suntool/framedemo.c
/usr/src/sun/suntool/perfmeter.c
/usr/src/sun/suntool/meter.c
/usr/src/sun/suntool/clockhands.c
/usr/src/sun/suntool/clockhands.rom.c
/usr/src/sun/suntool/swin.c
/usr/src/sun/suntool/get_view_surface.c
/usr/src/sun/suntool/switcher.c
/usr/src/sun/suntool/meter.h
/usr/src/sun/suntool/clockhands.h
/usr/src/sun/suntool/basetools.h
/usr/src/sun/suntool/othertools.h
```

TEXT_PROCESSING

```
/usr/bin/addbib
/usr/bin/deroff
/usr/bin/eqn
/usr/bin/indxbib
/usr/bin/lookbib
/usr/bin/neqn
/usr/bin/nroff
/usr/bin/ptx
/usr/bin/refer
/usr/bin/roffbib
/usr/bin/sortbib
/usr/bin/tbl
/usr/bin/troff
/usr/lib/me/acm.me
/usr/lib/me/chars.me
/usr/lib/me/deltext.me
/usr/lib/me/eqn.me
/usr/lib/me/float.me
/usr/lib/me/footnote.me
/usr/lib/me/index.me
/usr/lib/me/local.me
/usr/lib/me/null.me
/usr/lib/me/refer.me
/usr/lib/me/revisions
/usr/lib/me/sh.me
/usr/lib/me/tbl.me
/usr/lib/me/thesis.me
/usr/lib/ms/ms.acc
/usr/lib/ms/ms.cov
/usr/lib/ms/ms.eqn
/usr/lib/ms/ms.ref
/usr/lib/ms/ms.tbl
```

```
/usr/lib/ms/ms.ths
/usr/lib/ms/ms.toc
/usr/lib/refer/hunt
/usr/lib/refer/inv
/usr/lib/refer/mkey
/usr/lib/tmac/tmac.a
/usr/lib/tmac/tmac.an
/usr/lib/tmac/tmac.bib
/usr/lib/tmac/tmac.cp
/usr/lib/tmac/tmac.e
/usr/lib/tmac/tmac.imagen
/usr/lib/tmac/tmac.os
/usr/lib/tmac/tmac.r
/usr/lib/tmac/tmac.s
/usr/lib/tmac/tmac.scover
/usr/lib/tmac/tmac.sdisp
/usr/lib/tmac/tmac.skeep
/usr/lib/tmac/tmac.srefs
/usr/lib/tmac/tmac.sun
/usr/lib/tmac/tmac.vcat
/usr/lib/tmac/tmac.vgrind
/usr/lib/vfontedpr
/usr/lib/vgrindefs
/usr/ucb/vgrind
```

SETUP

```
/usr/etc/setup.files/setup.cards
/usr/etc/setup.files/setuphardware.file
/usr/etc/setup.files/xtr_root
/usr/etc/setup.files/xtr_rootarch
/usr/etc/setup.files/xtr_usrarch
/usr/etc/setup.files/xtr_symlinks
/usr/etc/setup.files/xtr_standalone
/usr/etc/setup.files/xtr_standpub
/usr/etc/setup.files/xtr_client
/usr/etc/setup.files/copy_client
/usr/etc/setup.files/fix_client
/usr/etc/setup.files/fix_hostname
/usr/etc/setup.files/fix_domainname
/usr/etc/setup.files/fix_rc.boot
/usr/etc/setup.files/fix_servers
/usr/etc/setup.files/rootmenu
/usr/etc/setup.files/setup.tty
/usr/etc/setup.files/setup.window
/usr/etc/setup.files/setup.config
/usr/etc/setup
```

STAND_DIAG

```
/usr/stand/extract_diags
```

FORTRAN

```
/usr/bin/f77
/usr/bin/ratfor
/usr/include/f77/usercore77.h
/usr/include/f77/cgidefs77.h
/usr/lib/cg
/usr/lib/f1
/usr/lib/f77pass1
/usr/lib/iropt
/usr/lib/libF77.a
/usr/lib/libI77.a
/usr/lib/libU77.a
/usr/lib/libcgi77.a
/usr/lib/libcore77.a
/usr/ucb/fpr
```

USR_DIAG

```
/usr/diag/loopback
/usr/diag/vid.120.pat
/usr/diag/sysdiag/.cshrc
/usr/diag/sysdiag/.login
/usr/diag/sysdiag/.suntools
/usr/diag/sysdiag/.suntools-ipc
/usr/diag/sysdiag/devtest
/usr/diag/sysdiag/dcptest
/usr/diag/sysdiag/skyprobe
/usr/diag/sysdiag/probe
/usr/diag/sysdiag/vmem
/usr/diag/sysdiag/pmem
/usr/diag/sysdiag/disk
/usr/diag/sysdiag/gpmtest
/usr/diag/sysdiag/sptest
/usr/diag/sysdiag/reply
/usr/diag/sysdiag/c160
/usr/diag/sysdiag/ipctest
/usr/diag/sysdiag/ffpusr
/usr/diag/sysdiag/softfp
/usr/diag/sysdiag/gpmtest.all.2p
/usr/diag/sysdiag/gpmtest.allbutgb.2p
/usr/diag/sysdiag/gpmtest.fifo_vme.2p
/usr/diag/sysdiag/gpmtest.fifo_vme_dec.2p
/usr/diag/sysdiag/gpmtest.fpalu.2p
/usr/diag/sysdiag/gpmtest.fpmult.2p
/usr/diag/sysdiag/memtop
/usr/diag/sysdiag/gpmtest.fprega.2p
/usr/diag/sysdiag/gpmtest.fpregb.2p
/usr/diag/sysdiag/gpmtest.gbnorm.2p
/usr/diag/sysdiag/gpmtest.gbrmw.2p
/usr/diag/sysdiag/gpmtest.int_flag.2p
/usr/diag/sysdiag/gpmtest.pp_29116.2p
/usr/diag/sysdiag/gpmtest.ppfifo.2p
/usr/diag/sysdiag/gpmtest.ppprom.2p
```

```
/usr/diag/sysdiag/gpmtest.scrpad.2p
/usr/diag/sysdiag/gpmtest.shmem.2p
/usr/diag/sysdiag/gpmtest.vme_byte.2p
/usr/diag/sysdiag/gpmtest.vme_read.2p
/usr/diag/sysdiag/gpmtest.vme_read_byte.2p
/usr/diag/sysdiag/gpmtest.vp_29116.2p
/usr/diag/sysdiag/gpmtest.vpprom.2p
/usr/diag/sysdiag/gpmtest.xoperand.2p
/usr/diag/sysdiag/gpmtest.yoperand.2p
/usr/diag/sysdiag/disktop
/usr/diag/sysdiag/dev
/usr/diag/sysdiag/tapetop
/usr/diag/sysdiag/devtop
/usr/diag/sysdiag/setterm
/usr/diag/sysdiag/endt
/usr/diag/sysdiag/sysdiag
/usr/diag/sysdiag/sysdiag.help
/usr/diag/sysdiag/nextlog
/usr/diag/sysdiag/ipctop
/usr/diag/sysdiag/options
```

GRAPHICS

```
/usr/bin/tek
/usr/bin/t4013
/usr/bin/t300
/usr/bin/t300s
/usr/bin/t450
/usr/bin/aedplot
/usr/bin/bgplot
/usr/bin/crtplot
/usr/bin/dumbplot
/usr/bin/gigiplot
/usr/bin/hpplot
/usr/bin/vplot
/usr/bin/plot
/usr/include/cgicbind.h
/usr/include/cgiconstants.h
/usr/include/cgidefs.h
/usr/include/cgipw.h
/usr/include/usercore.h
/usr/lib/libcgi.a
/usr/lib/libcgi77.a
/usr/lib/libcore.a
/usr/lib/libcore77.a
/usr/lib/libcorepas.a
/usr/lib/libcoresky.a
/usr/lib/libf77plot.a
/usr/lib/libplot.a
/usr/lib/lib300.a
/usr/lib/lib300s.a
/usr/lib/lib4013.a
/usr/lib/lib4014.a
```

```
/usr/lib/lib450.a
/usr/lib/libvt0.a
/usr/lib/libplotaed.a
/usr/lib/libplotbg.a
/usr/lib/libplotdumb.a
/usr/lib/libplotgigi.a
/usr/lib/libplot2648.a
/usr/lib/libplot7221.a
/usr/lib/libplotimagen.a
```

PASCAL

```
/usr/lib/how_pc
/usr/lib/how_pi
/usr/lib/how_pix
/usr/lib/how_pxp
/usr/lib/libpc.a
/usr/lib/libpc_p.a
/usr/lib/f1
/usr/lib/pc0
/usr/lib/pc2.il
/usr/lib/pc3
/usr/lib/pc3.5strings
/usr/lib/pcexterns.o
/usr/lib/pi3.5strings
/usr/lib/px_header
/usr/ucb/pc
/usr/ucb/pi
/usr/ucb/pix
/usr/ucb/pmerge
/usr/ucb/px
/usr/ucb/pxp
/usr/ucb/pxref
```

PROFILED

```
/usr/lib/libF77_p.a
/usr/lib/libI77_p.a
/usr/lib/libU77_p.a
/usr/lib/libc_p.a
/usr/lib/libcurses_p.a
/usr/lib/libm_p.a
/usr/lib/libpc_p.a
/usr/lib/libpfc_p.a
/usr/lib/libtermcap_p.a
/usr/lib/libtermlib_p.a
```

UUCP

```
/usr/bin/uucp
/usr/bin/uulog
/usr/bin/uuname
/usr/bin/uusend
/usr/bin/uustat
```

```
/usr/bin/uux
/usr/lib/uucp/uuxqt
/usr/lib/uucp/uucico
/usr/lib/uucp/uuclean
/usr/lib/uucp/uusub
/usr/lib/uucp/L-devices
/usr/lib/uucp/L-dialcodes
/usr/lib/uucp/L.cmds
/usr/lib/uucp/L.sys
/usr/lib/uucp/SEQF
/usr/lib/uucp/USERFILE
/usr/lib/uucp/uuck
/usr/lib/uucp/uucp.day
/usr/lib/uucp/uucp.hour
/usr/lib/uucp/uucp.night
/usr/lib/uucp/uucp.noon
/usr/lib/uucp/uucp.week
/usr/lib/uucp/uupoll
/usr/spool/uucppublic/.hushlogin
/usr/spool/uucp/C.//
/usr/spool/uucp/D.//
/usr/spool/uucp/D.noname//
/usr/spool/uucp/LOGFILE
/usr/spool/uucp/OLD//
/usr/spool/uucp/SYSLOG
```

SYSTEM_V

```
/usr/5include/*
/usr/5lib/*
/usr/5bin/*
/usr/bin/cflow
/usr/bin/csplit
/usr/bin/ctrace
/usr/bin/cut
/usr/bin/cxref
/usr/bin/diffmk
/usr/bin/dirname
/usr/bin/getopt
/usr/bin/id
/usr/bin/logname
/usr/bin/nl
/usr/bin/pack
/usr/bin/paste
/usr/bin/pcat
/usr/bin/sdiff
/usr/bin/unpack linked to ./bin/pcat
/usr/bin/xargs
/usr/etc/devnm
/usr/etc/grpck
/usr/etc/link
/usr/etc/pwck
/usr/etc/unlink
```

```
/usr/lib/ctrace/
/usr/lib/ctrace/runtime.c
/usr/lib/dag
/usr/lib/flip
/usr/lib/lpfx
/usr/lib/nmf
/usr/lib/xpass
```

MAN

```
/usr/man/man1/*
/usr/man/man2/*
/usr/man/man3/*
/usr/man/man4/*
/usr/man/man5/*
/usr/man/man6/*
/usr/man/man7/*
/usr/man/man8/*
/usr/man/manl/*
```

DEMO

```
/usr/demo/cursor_demo
/usr/demo/framedemo
/usr/demo/spheresdemo
/usr/demo/canvas_demo
/usr/demo/jumpdemo
/usr/demo/bouncedemo
/usr/demo/MAPS/
/usr/demo/MAPS/map.1
/usr/demo/MAPS/map.10
/usr/demo/MAPS/map.2
/usr/demo/MAPS/map.3
/usr/demo/MAPS/map.4
/usr/demo/MAPS/map.5
/usr/demo/MAPS/map.6
/usr/demo/MAPS/map.7
/usr/demo/MAPS/map.8
/usr/demo/MAPS/map.9
/usr/demo/DATA/bottle.dat
/usr/demo/DATA/egg.dat
/usr/demo/DATA/glass.dat
/usr/demo/DATA/icosa.dat
/usr/demo/DATA/mtxs.rotobj
/usr/demo/DATA/pyramid.dat
/usr/demo/DATA/rings.vecs
/usr/demo/DATA/shuttle.vecs
/usr/demo/DATA/socbal.dat
/usr/demo/DATA/space.dat
/usr/demo/DATA/string.vecs
/usr/demo/DATA/testmol
/usr/demo/DATA/vw.vecs
/usr/demo/COLORPIX/colorimage.1
/usr/demo/COLORPIX/colorimage.2
```

```
/usr/demo/SRC/draw.c
/usr/demo/SRC/shaded.c
/usr/demo/SRC/showmap.c
/usr/demo/SRC/stringart.c
/usr/demo/SRC/suncube.c
/usr/demo/SRC/cframedemo.c
/usr/demo/SRC/show.c
/usr/demo/SRC/maze.c
/usr/demo/SRC/rotobj.c
/usr/demo/SRC/flight.c
/usr/demo/SRC/goban.c
/usr/demo/SRC/goboard.c
/usr/demo/SRC/gopanel.c
/usr/demo/SRC/goservice.c
/usr/demo/SRC/gocapture.c
/usr/demo/SRC/gocircle.c
/usr/demo/SRC/goprint.c
/usr/demo/SRC/demolib.h
/usr/demo/SRC/flight_dat.h
/usr/demo/SRC/goban.h
/usr/demo/SRC/goban.icon
/usr/demo/SRC/Makefile
/usr/demo/globeframes/frame.0
/usr/demo/globeframes/frame.1
/usr/demo/globeframes/frame.10
/usr/demo/globeframes/frame.11
/usr/demo/globeframes/frame.12
/usr/demo/globeframes/frame.13
/usr/demo/globeframes/frame.14
/usr/demo/globeframes/frame.15
/usr/demo/globeframes/frame.16
/usr/demo/globeframes/frame.17
/usr/demo/globeframes/frame.18
/usr/demo/globeframes/frame.19
/usr/demo/globeframes/frame.2
/usr/demo/globeframes/frame.20
/usr/demo/globeframes/frame.21
/usr/demo/globeframes/frame.22
/usr/demo/globeframes/frame.23
/usr/demo/globeframes/frame.24
/usr/demo/globeframes/frame.25
/usr/demo/globeframes/frame.26
/usr/demo/globeframes/frame.27
/usr/demo/globeframes/frame.28
/usr/demo/globeframes/frame.29
/usr/demo/globeframes/frame.3
/usr/demo/globeframes/frame.30
/usr/demo/globeframes/frame.4
/usr/demo/globeframes/frame.5
/usr/demo/globeframes/frame.6
/usr/demo/globeframes/frame.7
/usr/demo/globeframes/frame.8
/usr/demo/globeframes/frame.9
```

```
/usr/demo/READ_ME
/usr/demo/Readme.goban
/usr/demo/maze
```

GAMES

```
/usr/games/lib/chess.book
/usr/games/lib/crib.instr
/usr/games/lib/fortunes.dat
/usr/games/lib/hackdir/
/usr/games/lib/hackdir/data
/usr/games/lib/hackdir/help
/usr/games/lib/hackdir/hh
/usr/games/lib/hackdir/rumors
/usr/games/lib/hackdir/perm
/usr/games/lib/hackdir/record
/usr/games/lib/cards.pck
/usr/games/lib/backrules
/usr/games/lib/cfscores
/usr/games/adventure
/usr/games/backgammon
/usr/games/teachgammon
/usr/games/boggle
/usr/games/bogdict
/usr/games/chess
/usr/games/cribbage
/usr/games/fortune
/usr/games/hack
/usr/games/hangman
/usr/games/chesstool
/usr/games/gammontool
/usr/games/boggletool
/usr/games/canfieldtool
/usr/games/life
/usr/games/gammonscore
/usr/games/boggledict
/usr/games/bcd
/usr/games/arithmetic
/usr/games/btlgammon
/usr/games/banner
/usr/games/bj
/usr/games/cfscores
/usr/games/factor
/usr/games/fish
/usr/games/number
/usr/games/random
/usr/games/wump
/usr/games/canfield
/usr/games/primes
```

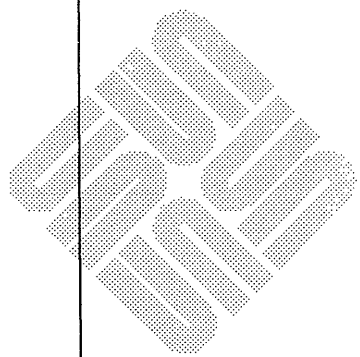
VTROFF

```
/usr/bin/vplot
/usr/lib/rvcat
```

```
/usr/lib/rvsort
/usr/lib/vcat
/usr/lib/vdmp
/usr/lib/vfontinfo
/usr/lib/vfw
/usr/lib/vpf
/usr/lib/vpfW
/usr/lib/vplotf
/usr/lib/vpltdmp
/usr/lib/vpsf
/usr/lib/vpsfW
/usr/lib/vsort
/usr/lib/vswap
/usr/lib/vwidth
/usr/ucb/vtroff
/usr/lib/vfont/*
```

Revision History

Revision	Date	Comments
01 α	11 April 1986	Preliminary Review.
02 β	15 May 1986	Beta Review
03 β2	18 June 1986	Second Beta
04	22 June 1986	Pilot Ship
05	22 August 1986	Third Beta Ship under temporary PN 800-1599



Notes